Thèse n° 9526

A Network Calculus Analysis of Asynchronous Mechanisms in Time-Sensitive Networks

Présentée le 17 mars 2023

Faculté informatique et communications
Laboratoire pour les communications informatiques et leurs applications 2
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Ehsan MOHAMMADPOUR

Acceptée sur proposition du jury

Prof. V. Kuncak, président du jury
Prof. J.-Y. Le Boudec, directeur de thèse
Prof. A. Rizk, rapporteur
Prof. A. Mifdaoui, rapporteuse
Prof. P. Thiran, rapporteur

École
polytechnique
fédérale
de Lausanne

2023

The more you know, the more you know you don't know.
— *Aristotle*

To my parents…

Acknowledgements

First of all, I would like to express my sincere gratitude to my thesis director, Prof. Jean-Yves Le Boudec, for his guidance throughout my PhD journey. His great mind and deep knowledge of the domain, together with his humble manner and unique teaching style, gave me a great opportunity to grow at a rapid pace. It was my great honor to do my PhD research, to complete my thesis, and to graduate under his supervision.

Since my first day at EPFL, I had the opportunity to work closely with Dr. Eleni Stai. With her great personality and her positive and constructive attitude, she was a good colleague and remains a nice friend. I appreciate her for all her scientific and mental support. I also thank my PhD committee members, Prof. Viktor Kunčak, Prof. Amr Rizk, Prof. Ahlem Mifdaoui, and Prof. Patrick Thiran, for their insightful feedback and inspiring discussions. I am grateful to Prof. Ola Svensson and Dr. Tony Przygienda for being my mentors throughout this journey.

My life in LCA2 was immensely facilitated by the help of the best administrative assistants: Patricia Hjelt helped me a lot to settle down in Lausanne upon my arrival and to organize my work trips around the world. Angela Devenoge patiently and perfectly resolved my life-in-Lausanne questions. Holly B. Cogliati filled in my always-missing *articles*, and *hollified* my papers. The system administrators, Marc-André Lüthi and Yves Lopes, were the *default gateway* when it came to technical IT issues. I appreciate the effort of all these people during all these years.

During my time in LCA2, I had the pleasure to work with nice colleagues in LCA2: Roman, Maaz, Wajeb, Marguerite, Cong, Hossein, Ludovic, Stéphan, and Plouton. We shared enjoyable moments together as teaching assistants in organizing homework and exercises for TCP/IP and Performance Evaluation courses. During this time, Roman was a caring and supporting friend who was there for me during the ups and downs of my PhD career. I am grateful for their companionship. Outside LCA2, I had the honor to serve as a representative of EDIC PhD students, along with Sandra, for four years. It was amazing to collaborate with the EDIC committee and to attend EDOC meetings, as well as being involved in extra curriculum activities such as IC Openhouse and the events organized by the EPIC association. I thank them all for such a unique experience.

I would like to thank my friends in Switzerland, Ahmad, Mohammad and Bahareh, Ramin and Armita, Ali, Salman, Atena and Morteza, Amirhossein and Homa, Chris, Isabella, and Melanie, with whom I experienced plenty of skiing, volleyball, outings, and trips. Also, I would like to thank my old friends outside of Switzerland, Hossein and Mohammadreza, with whom we organized regular phone calls, as these moment were a huge support during my PhD times.

I am very lucky to have all these friends around me.

There were many other people who contributed to my well-being during my PhD experience, but I cannot name them all as the list would be too long. If their names are not mentioned above, I hope they know that I know who they are. I thank them all for coming into my life.

Last but not least, my heart goes to my family who believed in me and my abilities. My parents provided constant support of any kind during my studies, especially my PhD journey. My brothers provided the necessary mental support and guidance for me to overcome the challenges, and my sweet sister was always emotionally available for me in the tough times. I thank them all for being there for their unconditional love.

Lausanne, November 17, 2022 E. M.

Abstract

Time-sensitive networks provide worst-case guarantees for applications in domains such as the automobile, automation, avionics, and the space industries. A violation of these guarantees can cause considerable financial loss and serious damage to human lives. To avoid this, it is crucial to provide a deterministic analysis of time-sensitive networks. Our analysis is based on network calculus, a framework for computing bounds on delay and backlog by using bit-level arrival-curve and service-curve characterizations. In this thesis, we focus on the analysis of time-sensitive networks to address four key requirements, specifically, bounded worst-case delay and delay jitter (defined as the difference between worst-case and best-case delays), zero congestion loss, and in-order packet delivery.

In time-sensitive networks, source flows are constrained by the number of packets. A common approach for obtaining a delay bound is to derive a bit-level arrival curve from a packet-level arrival curve, and then to use network calculus. However, such a method is not tight: we show that better bounds can be obtained by directly exploiting the arrival curves expressed at the packet level. By exploiting the information on the packet transmission rate, our analysis method also leads to better bounds when flows are constrained with bit-level arrival curves.

Second, we focus on the computation of delay and backlog bounds for an asynchronous configuration of IEEE Time-Sensitive Networking with a Credit-Based Shaper (CBS) and Asynchronous Traffic Shaping (ATS). ATS is an implementation of the interleaved regulator that regulates traffic in the network before admitting it into a CBS buffer, thus avoiding burstiness cascades. Due to the interleaved regulator, traffic is regulated at every switch, which allows for the computation of explicit delay and backlog bounds. Furthermore, we obtain a per-flow bound for the response time of CBSs by deriving novel results on service curves and credit bounds for CBSs. We also compute a per-flow bound on the response time of the interleaved regulator. Using the above results, we compute bounds on the per-class backlogs. Then, we use the newly computed delay bounds, along with recent results on interleaved regulators from the literature, to derive tight end-to-end delay bounds; then, we show that these derived bounds are less than the sums of per-switch delay bounds.

Third, we analyze the effects of re-sequencing buffers, used to provide in-order packet delivery in time-sensitive networks. To provide worst-case guarantees, up until recently, there has not been a precise understanding of per-flow reordering metrics or of the dimensioning of re-sequencing buffers. We show that a previously proposed per-flow metric, called reordering late time offset (RTO), determines the timeout value. If the network is lossless, another previously defined metric, called reordering byte offset (RBO), determines the required buffer. If packet losses cannot be ignored, the required buffer can be larger than RBO, and depends on the jitter, on the arrival curve of the flow at its source, and on the timeout. Then, we develop a calculus for computing the RTO for a flow path. Our method uses a novel relation with the jitter and the arrival curve, together with a decomposition of the path into non-order-preserving and order-preserving elements. We also analyze the effect of re-sequencing buffers on the worst-case delay, the jitter, and the propagation of arrival curves. We show in particular that, in a lossless (but non-order-preserving) network, re-sequencing is “for free”, in other words, it does not increase worst-case delay or jitter, whereas in a lossy network, re-sequencing increases the worst-case delay and jitter. We apply the analysis to evaluate the performance effect of placing re-sequencing buffers at intermediate points and illustrate the results on two industrial test-cases.

Finally, we focus on the analysis of dampers as asynchronous mechanisms for reducing delay jitter by delaying packets for the amount written in packet headers. Dampers have the potential to solve the burstiness cascade problem of time-sensitive networks; this can be done in a scalable way, as dampers can be stateless. Dampers exist in several variants: Some apply to only the earliest-deadline-first schedulers, whereas others can be associated with any packet schedulers; some enforce FIFO ordering, whereas some others do not. Existing analyses of dampers are specific to some implementations and some network configurations; also, they assume ideal, non-realistic clocks. We provide a taxonomy of all existing dampers in general network settings and analyze their timing properties in presence of non-ideal clocks. We give, in particular, formulas for computing residual jitter bounds of networks with dampers of any kind. We show that non-FIFO dampers can cause reordering due to clock non-idealities and that the combination of FIFO dampers with non-FIFO network elements can very negatively affect the performance bounds. Our results can be used to analyze timing properties and burstiness increases in any time-sensitive networks, as we illustrate on an industrial case-study.

Keywords: Network calculus, Time-Sensitive Networking, Deterministic Networking, Asynchronous mechanisms, Delay bound, Zero congestion-loss, Jitter, In-order packet delivery, Burstiness cascade, Packet-level constraint, Bit-level constraint, Interleaved Regulator, Damper, Credit-based shaper, Re-sequencing buffer, Non-ideal clocks

Résumé

Les réseaux temps-réel fournissent des garanties pire-cas aux applications dans les domaines tels que l'asservissement, l'avionique, les industries automobiles et spatiales. Le non-respect de ces garanties peut entraîner une perte financière considérable ou mettre en péril des vies humaines. Pour éviter cela, il est crucial de réaliser une analyse déterministe des réseaux temps-réel. Notre analyse est basée sur le calcul réseau, une théorie pour obtenir des bornes de latence et de backlog grâce aux représentations que sont les courbes d'arrivée par bits et les courbes de service. Dans cette thèse, nous nous focalisons sur l'analyse des réseaux temps-réel pour évaluer quatre exigences clés : un délai pire-cas borné, une gigue bornée (définie comme étant la différence entre le délai pire-cas et le délai minimal), aucune perte par congestion et une préservation de l'ordre des paquets.

Dans les réseaux temps-réel, les flux sont contraints à leur source en termes de nombre de paquets. Une approche classique pour obtenir des bornes de délai est alors de construire une courbe d'arrivée par bits grâce à la courbe d'arrivée par paquets et d'utiliser le calcul réseau par la suite. Cependant, cette technique n'est pas précise : nous montrons que de meilleures bornes peuvent être obtenues en exploitant directement les courbes d'arrivée exprimées au niveau des paquets. En utilisant l'information sur le taux de transmission des paquets, notre analyse améliore également les bornes lorsque les flux sont contraints par des courbes d'arrivée par bit.

Ensuite, nous nous intéressons au calcul de bornes de délai et de backlog pour une configuration asynchrone de IEEE *Time-Sensitive Networking* avec *Credit-Based Shaper* (CBS, lissage par crédit) et *Asynchronous Traffic Shaping* (ATS, lisseur de trafic asynchrone). ATS est une implémentation du régulateur entrelacé (*interleaved regulator*) qui lisse le trafic dans le réseau avant de l'admettre dans un buffer CBS, évitant ainsi les cascades de burst. Grâce au régulateur entrelacé, le trafic est lissé à chaque commutateur, ce qui permet le calcul de bornes de délai et de backlog explicites. De plus, nous obtenons une borne pour chaque flux sur le temps de réponse des CBSs grâce à des résultats originaux sur les courbes de service et les bornes de crédit des CBSs. Nous calculons aussi une borne sur le temps de réponse du régulateur entrelacé pour chaque flux. En utilisant ces résultats, nous obtenons les bornes de backlog pour chaque classe. Ensuite, nous combinons ces nouvelles bornes de délai avec des résultats récents de la littérature sur les régulateurs entrelacés pour obtenir des bornes précises sur la latence de bout en bout. Nous montrons également que ces bornes sont inférieures à celles obtenues en sommant les bornes de délai obtenues pour chaque commutateur.

Puis nous analysons les effets des buffers de re-séquencement utilisés dans les réseaux temps-réel pour garantir la distribution des paquets dans l'ordre. Jusqu'à récemment, il n'y avait pas de compréhension précise – ni sur les métriques de ré-ordonnancement, ni sur le dimensionnement des buffers de re-séquencement – qui permette l'obtention de garanties pire-cas. Nous montrons que le *reordering late time offset* (RTO), une métrique par flux définie dans la littérature, fournit la valeur de timeout. Si le réseau est sans pertes, une autre métrique de la littérature, le *reordering byte offset* (RBO) fournit la taille requise du buffer. Si les pertes de paquets ne peuvent pas être négligées, la taille du buffer requise est plus grande que le RBO et dépend de la gigue, de la courbe d'arrivée du flux à sa source et du timeout. Ensuite, nous développons une théorie de calcul du RTO pour le chemin d'un flux. Notre méthode utilise une nouvelle relation avec la gigue et la courbe d'arrivée, ainsi qu'une décomposition du chemin en segments, selon qu'ils préservent ou non l'ordre de paquets. Nous analysons également l'effet des buffers de re-séquencement sur le délai pire-cas, la gigue et la propagation des courbes d'arrivée. Nous montrons en particulier que le re-séquencement est "gratuit" dans les réseaux sans pertes de paquets, c'est à dire qu'il n'augmente pas les bornes de délai pire-cas ou de gigue. En revanche, dans un réseau avec pertes, le re-séquencement augmente le délai pire-cas et la gigue. Nous appliquons notre théorie pour évaluer l'effet sur les performances d'un placement des buffers de re-séquencement aux points intermédiaires du réseau et nous illustrons nos résultats sur deux études de cas industrielles.

Enfin, nous nous intéressons à l'analyse des amortisseurs (*dampers*). Ce sont des mécanismes asynchrones qui réduisent la gigue en retardant les paquets selon une durée inscrite dans leur en-tête. Les amortisseurs ont le potentiel de résoudre le problème de cascade des bursts dans les réseaux temps-réels. Ils peuvent passer à l'échelle car il ne conservent pas d'état pour les flux. Il existe plusieurs variantes d'amortisseurs. Certains n'existent qu'en combinaison avec un ordonnanceur de type *earliest deadline first* (EDF, plus proche date limite en premier), tandis que d'autres peuvent être associés avec n'importe quel ordonnanceur de paquet. Certains assure la conservation de l'ordre des paquets tandis que d'autres ne l'assurent pas. Les analyses existantes des amortisseurs sont limitées à certaines implémentations et configurations de réseau. Elles supposent en outre des horloges idéales donc non-réalistes. Nous fournissons une taxonomie des amortisseurs existants dans des réseaux aux paramètres généraux et nous analysons leurs propriétés temporelles en présence d'horloges non-idéales. En particulier, nous donnons les formules pour calculer la gigue résiduelle dans les réseaux avec amortisseurs, quelque soit le type de ces derniers. Nous montrons que les amortisseurs non-FIFO peuvent amener à un ré-ordonnancement des paquets en raison des imperfections des horloges et que l'association d'amortisseurs non-FIFO avec des éléments de réseau FIFO peut affecter très négativement les bornes de performance. Nos résultats peuvent être utilisés pour analyser les propriétés temporelles et l'augmentation de burst dans n'importe quel réseau temps-réel, ce que nous illustrons avec un cas industriel.

Mots-clés : Calcul réseau, Réseaux temps-réel, Réseaux déterministes, Mécanismes asynchrones, Bornes de délai, Absence de perte par congestion, Gigue, Distribution des paquets dans l'ordre, Cascade de bursts, Contraintes de niveau paquet, Contrainte de niveau bit, Amortisseur, Lissage par crédit, Buffer de ré-séquencement, Horloges imparfaites

Contents

Acknowledgements i

Abstract (English and Français) iii

List of Abbreviations xiii

List of Figures xv

List of Tables xvii

1 Introduction 1
- 1.1 Background and Motivation 2
- 1.2 Challenges 3
  - 1.2.1 Worst-Case Delay 4
  - 1.2.2 Zero Congestion-Loss 4
  - 1.2.3 Worst-Case Jitter 4
  - 1.2.4 In-Order Packet Delivery 5
  - 1.2.5 Seamless Redundancy 5
- 1.3 Contributions 6
- 1.4 Roadmap 7

2 Time-Sensitive Networks 11
- 2.1 Traffic Specification 12
- 2.2 Scheduling Mechanisms 13
  - 2.2.1 Strict Priority 13
  - 2.2.2 Fair Queuing 14
  - 2.2.3 Credit-Based Mechanisms 15
  - 2.2.4 TSN and DetNet 16
- 2.3 Flow Regulation and Dampers 17
- 2.4 Conclusion 18

3 Technical Background 19
- 3.1 Mathematical Foundation 19
  - 3.1.1 Pseudo-inverse Functions 19
  - 3.1.2 Min-plus Algebra 20

3.1.3 Additional Definitions 22
3.2 Network Calculus 23
3.2.1 Flow Model 23
3.2.2 Arrival and Service Curves 24
3.2.3 Local Analysis 26
3.2.4 End-to-End Analysis 29
3.3 Conclusion 31
3.4 Notation 32

4 Novel Network Calculus Delay Bounds in Time-Sensitive Networks 33
4.1 Related Works 34
4.2 System Model 35
4.2.1 Bit-level Arrival Curve 36
4.2.2 g-regulation 36
4.2.3 Packet-level Arrival Curve 37
4.3 Relations among Flow Regulations 40
4.4 Delay Bounds for Packet-level Arrival Curves 41
4.5 Improvements on Existing Network-Calculus Delay-Bounds 44
4.6 Comparison of the Different Bounds 46
4.7 Numerical Illustration 47
4.7.1 Flows with Packet-level Arrival Curve 47
4.7.2 Flows with g-regulation 48
4.7.3 Flows with Bit-level Arrival curve 48
4.8 Conclusion 49
4.9 Notation 50

5 Delay and Backlog bounds in TSN with Interleaved Regulators 51
5.1 System Model 52
5.1.1 Architecture of a TSN node 52
5.1.2 Operation of CBS 53
5.1.3 Flow Regulation 54
5.1.4 Other Notations and Definitions 55
5.2 Delay Bounds in TSN 56
5.2.1 Novel Credit Bound for CBS 56
5.2.2 Service Curve Offered by CBFS to AVB flows 58
5.2.3 Upper Bound on the Response Time in CBFS 59
5.2.4 Bound on the Response Time in the Interleaved Regulator 59
5.2.5 Upper Bound on the End-to-End Delay 60
5.3 Backlog Bounds 60
5.3.1 Backlog Bound on Interleaved Regulator 60
5.3.2 Backlog Bound on Class-Based FIFO System 61
5.4 Case Study 61
5.4.1 TSN Network Setup and Flows 62

5.4.2 Computation of Theoretical Bounds 63
5.4.3 Numerical Example of Tightness 63
5.4.4 Sub-additivity of End-to-End Delay Bound 64
5.4.5 Tightness of End-to-End Delay Bound 64
5.5 Conclusion 65
5.6 Notation 66

6 Packet Reordering 67
6.1 Related Work 69
6.2 Background Information 70
6.2.1 Network Assumptions 70
6.2.2 Packet Reordering Metrics 70
6.2.3 Re-sequencing Buffer 72
6.2.4 Assumptions on Arrival Curves 72
6.2.5 Network Calculus Results in Non-FIFO Networks 73
6.3 Properties of the Re-sequencing Buffer 73
6.3.1 Input-output Characterization of the Re-sequencing Buffer 74
6.3.2 Optimal Dimensioning of the Parameters of the Re-sequencing Buffer . 75
6.3.3 Effect of Re-sequencing on Worst-case Delay, Jitter and Arrival Curve . . 76
6.4 Computing RTO and RBO 78
6.4.1 RTO and RBO for Network Elements 78
6.4.2 Concatenation Results 80
6.5 Application to Performance of Intermediate Re-sequencing 83
6.5.1 Methodology 83
6.5.2 Case Study 1: Automotive Network 85
6.5.3 Case study 2: Orion network 87
6.6 Conclusion 88
6.7 Notation 89

7 Analysis of Dampers in Time-Sensitive Networks with Non-ideal Clocks 91
7.1 Related Works 92
7.2 System Model 93
7.2.1 Terminology 93
7.2.2 Assumptions on the Clocks 96
7.3 Taxonomy of Dampers 97
7.3.1 Dampers without FIFO constraint 97
7.3.2 Dampers with FIFO constraint 99
7.3.3 Damper Header Computation 101
7.4 Delay Analysis of Dampers without FIFO constraints 103
7.5 Packet Reordering in Dampers without FIFO Constraints 109
7.6 Analysis of Dampers with FIFO Constraints 112
7.7 Numerical Evaluation 115
7.8 Conclusion 118

7.9 Notation 118

8 Conclusion and Future Work 119

A Appendix (Chapter 4) 123
A.1 Technical Prerequisites 123
A.2 Proofs 126
  A.2.1 Proof of Proposition 4.1 126
  A.2.2 Proof of Proposition 4.2 128
  A.2.3 Proof of Lemma 4.1 129
  A.2.4 Proof of Lemma 4.2 131
  A.2.5 Proof of Theorem 4.2 132
  A.2.6 Proof of Theorem 4.3 138
  A.2.7 Proof of Theorem 4.5 141
  A.2.8 Proof of Proposition 4.3 142
  A.2.9 Proof of Proposition 4.4 143
A.3 Example of non c-Lipschitz service curve 145

B Appendix (Chapter 5) 147
B.1 Proof of Theorem 5.1 147
B.2 Proof of Theorem 5.2 151
B.3 Proof of Theorem 5.3 152

C Appendix (Chapter 6) 153
C.1 Re-sequencing Buffer Operation 153
C.2 Proofs 156
  C.2.1 Proof of Lemma 6.1 156
  C.2.2 Proof of Lemma 6.2 157
  C.2.3 Proof of Theorem 6.1 157
  C.2.4 Proof of Theorem 6.2 160
  C.2.5 Proof of Theorem 6.3 161
  C.2.6 Proof of Theorem 6.4 165
  C.2.7 Proof of Theorem 6.5 166
  C.2.8 Proof of Theorem 6.6 167
  C.2.9 Proof of Theorem 6.7 170
C.3 Details of Computations for Case Study 1 in Section 6.5.2 173

D Appendix (Chapter 7) 179
D.1 Lemmas for Section 7.3 179
D.2 Proof of Theorem 7.1 180
D.3 Proof of Theorem 7.2 186
D.4 Proof of Theorem 7.3 187
D.5 Proof of Theorem 7.4 187

D.6 Proof of Theorem 7.5 190
D.7 Proof of Theorem 7.6 192

Bibliography **208**

List of Publications **209**

Curriculum Vitae **211**

List of Abbreviations

ATS	Asynchronous Traffic Shaping
AVB	Audio Video Bridging
BDS	Bounded-Delay System
CBS	Credit-based Shaper
CDT	Control Data Traffic
DetNet	Deterministic Networking
DHU	Damper Header Update
DRR	Deficit Round-Robin
FOPLEQ	Flow-Order Preserving Latency-Equalizer
HoL	Head-of-Line
IR	Interleaved Regulator
IWRR	Interleaved Weighted Round-Robin
JCS	Jitter-Compensated System
LRQ	Length Rate Quotient
PE router	Provider-Edge router
RBO	Reordering Byte Offset
RCSP	Rate-Control Static Priority
RGCQ	Rotated Gate-Control-Queues
RTO	Reordering Late Time Offset
TAI	Temps Atomique International (international atomic time)
TFA	Total-Flow Analysis
TSN	Time-Sensitive Networking
WRR	Weighted Round-Robin

List of Figures

3.1 Illustration of pseudo-inverse functions. 20

3.2 The horizontal and vertical deviations from w' to w. 22

3.3 A c-Lipschitz function w (left, DRR service curve in [1]) and a non c-Lipschitz function w' (FIFO residual service curve in [2, Section 7.3.1]). The slope of the function is within $[-c, c]$ at any point in time for w. The function w' is not continuous at time θ. 22

3.4 Token-bucket (r, b) and staircase (τ, b) arrival curves. 25

3.5 Rate-latency service curve with rate R and latency T (left), and a DRR service-curve connected to a physical link with rate c [1] (right). 26

4.1 The considered FIFO system. 36

4.2 The two interpretations of TSN and DetNet interval. 38

5.1 Architecture of the considered TSN node output port. 53

5.2 Illustration of the queuing policy in interleaved regulators (IR) by TSN switches for four flows of class A. 54

5.3 Timing Model in TSN 55

5.4 A practical TSN network used for the case study. 62

5.5 Cumulative data input and output curves for the CBFS of H_1 and interleaved regulator of switch 1, with respect to Fig. (5.4). 62

5.6 TSN network for tightness of end-to-end delay bound. 64

6.1 Notation used in Section 6.3. 74

6.2 Notation for the sequence of network elements used in Section 6.4.2. S_s and S_e are respectively the first and the last non order-preserving elements in the sequence. α and α_s are arrival curves at the entrance of S_1 and S_s respectively. 81

6.3 A prototypical scenario used to analyze the performance impact of intermediate re-sequencing. Potential placements of re-sequencing buffers are at points 1, 2, or 3. 84

6.4 Double star automotive network [3]. 86

6.5 The Orion crew exploration vehicle network. 87

6.6 The end-to-end delay bounds (left) and jitter bounds (right) for the flows with in-order delivery requirement for the two strategies, i.e., placing re-sequencing buffers at the destinations or at every switch. In the lossless condition, the delay and jitter bounds do not depend on the strategy. 88

7.1 A local-area time-sensitive network example. 95
7.2 A large-scale time-sensitive network example with no jitter removal at backbone network. 95
7.3 A large-scale time-sensitive network example with jitter removal at backbone network. 96
7.4 Structure of a block whose delay and jitter bounds are computed in Theorem 7.1. 104
7.5 The notation used in Theorem 7.2. 108
7.6 Packet reordering scenario when two packets enter back-to-back to a JCS. . . . 109
7.7 A scenario that two packets become back-to-back after an RCSP and reordering occurs in the downstream RCSP. 111
7.8 The notation used in Theorem 7.3 and Theorem 7.4. 113
7.9 The notation used in Theorem 7.5 and Theorem 7.6. 114
7.10 The end-to-end jitter bounds for RCSP and RGCQ with TE time-stamping using basic jitter computation and using theorems 7.1 and 7.2. 116
7.11 The end-to-end delay bounds (left) and jitter bounds (right) for full deployment of dampers and in the absence of dampers. For FOPLEQ and HoL dampers, we assume that all the elements are FIFO. 117
7.12 The end-to-end jitter bounds for FOPLEQ and head-of-line dampers when switching fabrics are FIFO or not. 117

A.1 The execution trace used in the proof of Theorem 4.2. The delay of the packet with length $L_1^{\max}$ that arrives at time t' is Δ^{pkt}. 134

C.1 Notation for the sequence of network elements used in Theorem 6.7. 171

D.1 The notations used in the proof of Theorem 7.3. 187
D.2 The notations used in the proof of Theorem 7.4. 187
D.3 The notations used in the proof of Theorem 7.5. 190

List of Tables

4.1 Comparison of the delay upper-bound for packet-level arrival-curve in the example of Section 4.4. 43

4.2 Flow information for Section 4.7.1. 48

6.1 Re-sequencing buffer optimal timeout value and increase on end-to-end jitter and delay upper bound (with respect to the baseline with no re-sequencing buffer) for the four placement strategies in Section 6.5.1. T_i is the timeout of re-sequencing buffer placed at point i. We see that placing re-sequencing buffers at 1 and 2 provides better end-to-end delay and jitter comparing to the placement at 1 and 3. 85

6.2 Bounds on end-to-end jitter and worst-case delay for case study 1 with the four placement strategies in Section 6.5.1, under both lossless and lossy network conditions, followed by timeout value and size of the re-sequencing buffers. . . 86

7.1 Minimum values of the sum of delay bounds for the JCSs within a block such that clock synchronization improves the delay and jitter bounds in Theorem 7.1. 107

C.1 Arrival curve propagation from point 1 to point 9 under lossless network condition. Arrival curve at point i is $\alpha_i(t) = \min(6.4e3\ t + b_i, 125e6\ t + M_i)$, where b_i and M_i are in bytes and shown in the table. Since under lossless network condition, the re-sequencing buffers do not increase delay and jitter bounds, the arrival curves are the same for the four placement strategies. 173

C.2 Arrival curve propagation from point 1 to point 9 under lossy network condition. Arrival curve at point i is $\alpha_i(t) = \min(6.4e3\ t + b_i, 125e6\ t + M_i)$, where b_i and M_i are in bytes and shown in the table. 174

1 Introduction

Make everything as simple as possible, but not simpler.
— Albert Einstein

Classic communication networks were designed to provide statistical guarantees in terms of average delay, bandwidth, and packet loss to the end users for the applications such as internet surfing, file transfer, and e-mail services. In contrast, a set of other applications requires deterministic guarantees in the services offered to them: For example, the transmission of safety-relevant control messages in automotive networks requires a worst-case delay of 1 ms [4], the transmission of event-based control messages in industrial environment must be done within 10 ms [5], parametric data for fly-by-wire system in avionics networks must be delivered within 2 ms [6], and fault detection (e.g., occurrence of short circuits) in power-line equipment should be taken care of within 100 ms [7]. In these applications, a violation of the delay constraints or a loss of messages can cause severe damage to the human (e.g., operators and passengers) and equipment (e.g., electrical systems and machines); we refer to the networks that provide such guarantees as *time-sensitive networks*[1].

Time-sensitive networks aim primarily to provide deterministic guarantees on delay upper-bounds, in contrary to classic networks that focus on average delays. Moreover, due to the criticality of the applications, time-sensitive networks also guarantee zero congestion-loss by setting enough buffer spaces at the intermediate systems. However, feedback that slows down the flow of messages to avoid congestion is not an option for the applications in such networks. Today, with the emergence of applications, such as the tactile Internet, industrial Internet of Things [8], electricity distribution[9], and Industry 4.0 [10], the need for worst-case performance guarantees is on the rise. Hence, it is crucial to analyze time-sensitive networks by using proper methods and tools in order to achieve deterministic guarantees as required in such networks.

---

[1] The term "time-sensitive network" is not equal to IEEE Time-Sensitive Networking (TSN), but it does cover any network implementing IEEE TSN.

1.1 Background and Motivation

Traditionally, time-sensitive networks were built over field buses such as controller area network (CAN) [11] and FlexRay [12]. These technologies have limitations in terms of scalability and bandwidth (to the order of a few Mbps). To overcome these limitations, the most promising solutions were provided as extensions over Ethernet as a well-established technology [13]; extensions such as time-triggered Ethernet (TTEthernet), Ethernet for control automation technology (EtherCAT), process field net (PROFINET) and Avionics Full-Duplex Switched Ethernet (AFDX) [14, 15, 16]. Ethernet was standardized well in IEEE802.3 [17]. However, it was not initially designed to provide worst-case guarantees. To this end, the Institute of Electronics and Electrical Engineers (IEEE) took a step toward the standardization of the requirements (such as flow behavior and quality of service) and mechanisms (including scheduling, shaping, and reservation) to provide performance guarantees for multimedia applications; this was published as IEEE802.1BA Audio Video Bridging (AVB) [18].

Later, due to the increasing need for the standardization of mechanisms for applications with requirements stricter than the guarantees provided by IEEE802.1BA AVB, e.g., control-data traffic, the IEEE 802.1 Time Sensitive Networking (TSN) task group [19] is now in the process of providing standardization for a wider range of applications, including control-data traffic and multimedia streams over Link Layer (L2) of Internet protocol suite [20]. In fact, IEEE TSN aims to unify the requirements and mechanisms of the existing Ethernet-based solutions for time-sensitive networks with interconnections of switches and end systems. While IEEE TSN mainly considers local area networks (LANs), the Deterministic Networking (DetNet) working group of Internet Engineering Task Force (IETF) [21] intends to enable time-sensitive networks on a large scale by incorporating Internet Layer (L3) and by using IP packets; some applications of such networks are electrical utilities, building-automation systems, and industrial machine-to-machine[7].

An analysis of time-sensitive networks can be made by measurement or analytical approaches. The measurement can be done through a real network or by simulation. Even though measuring the metrics over a real network brings realistic measurements, it still requires a complex setup and access to the network, and this might not be always feasible. To avoid these complications, an alternative is to use software simulators and to implement an abstraction of the systems in a network, e.g., using NeSTiNg [22]. The main drawback of this method is the difficulty of detecting rare events (e.g., with probability of 10e-7). In contrast, analytical methods use mathematical modeling of the systems in a network with a certain level of abstraction; they can be probabilistic or deterministic. The probabilistic analytical methods, such as stochastic network calculus [23, 24], perform stochastic modeling of the systems and provide probabilistic measurements (e.g., probability of missing a deadline is 10e-6). Whereas, deterministic analytical methods model the systems based on their worst-case behaviors and obtain bounds on the metrics of interest. As a result, if the bounds are tolerable by the applications in time-sensitive networks, the performance guarantees are achieved; otherwise, a configuration change or extra resource allocation will be necessary to make it possible. In this thesis, we follow this approach by using deterministic network calculus [25, 26]; hence, we can provide proven performance guarantees within time-sensitive networks because, otherwise, it could lead to high financial losses, as well as dangers for humans and the environment. In the remainder of this document, by "network calculus" we mean "deterministic network calculus".

Network calculus is a mathematical framework with a set of tools and results for computing delay and backlog bounds by the means of arrival envelopes and service abstractions of the systems. The fundamentals were established by Parekh and Gallager [27] and Rene Cruz [28, 29, 30], then it was extended by [31]. Le Boudec and Thiran [25], as well as Chang [26], introduced the concepts of arrival and service curves that are the main building blocks of the currently-known network calculus: A service curve is an abstraction of a system, and an arrival curve is a bound on the amount of input traffic to this system within any duration of time. The network calculus tool-set in [25] enables us to compute delay and backlog bounds for a system by using the arrival and service curves. Moreover, the theory presented in [25] eases the analysis of systems in sequence by the means of service-curve concatenation, shaping and arrival-curve propagation. This makes network calculus a proper choice to be used in time-sensitive networks. Over several years, a plentiful of research papers and tools were published based on network calculus; the textbook by Bouillard et. al. [2] is the most recent collection of methods and results in this domain.

1.2 Challenges

According to the published and draft documents on time-sensitive networks [32, 33, 34, 35, 36, 37], the performance guarantees in such networks are defined in terms of worst-case delay, zero congestion-loss, jitter, as well as in-order packet delivery and seamless redundancy. Packet losses in networked systems could stem from buffer overflow (as a result of network congestion), transmission failure, network-element malfunction, etc. In time-sensitive networks, the goal is to avoid congestion losses by a proper configuration of buffer sizes and to recover from other possible network failures by the use of redundancy methods.

To this end, the documents on time-sensitive networks specify a number of scheduling mechanisms, redundancy methods, and packet ordering functions [35, 37]. In the analysis of time-sensitive networks, we first need a proper abstraction of these scheduling mechanisms in isolation, and their input traffic, by means of service and arrival curves; then, we can obtain delay and backlog bounds for a network element in isolation using network calculus. Second, we need to extend the analysis for an interconnection of such mechanisms to obtain end-to-end bounds. Finally, we need to capture the effect of redundancy methods and packet ordering functions on the other metrics of interest in time-sensitive networks. Hereafter, we discuss the challenges and the open problems in the analysis of the aforementioned metrics in time-sensitive networks.

1.2.1 Worst-Case Delay

The “worst-case delay" of a flow is the maximum delay of all non-lost packets sent by the flow during its lifetime; it is one of the most common requirement across time-sensitive applications. To this end, the documents in time-sensitive networks specify a number of per-class queuing and scheduling mechanisms such as credit-based shaper [35, Annex L] and strict priority. Because the mechanisms are per-class, one key issue in this context is knowing how to deal with the burstiness cascade: individual flows that share a resource dedicated to a class can see their burstiness increase, which can in turn cause increased burstiness to other flows downstream of this resource; as a consequence, this can lead to increased delay bound at the downstream systems. Another key issue in this context is cyclic dependency: some flows partially share their paths with other flows and form a cycle. The presence of the burstiness cascade in a network with cyclic dependencies can make the computation of end-to-end delay bounds hard and can lead to network instability [38, 39].

In this thesis, we focus on the analysis of two mechanisms, interleaved regulators and dampers. They break cyclic dependencies and avoid the burstiness-cascade issue by recreating arrival curve of flows according to their sources.

1.2.2 Zero Congestion-Loss

A congestion occurs when the input rate to an output queue exceeds the output rate for a sufficient amount of time; this in turn causes the queue to be overflowed and packets to be discarded. Time-sensitive networks seek to avoid such phenomenon as one of its goals called “zero congestion-loss". To provide such a guarantee, it is necessary to compute an upper bound on the backlog present in any queuing system within a flow path; then, the queue size should be set at least as much as the backlog bound. We can compute backlog bounds for any queuing system by using network calculus, as long as we know an arrival curve of the input traffic and a service curve offered to this queue.

1.2.3 Worst-Case Jitter

The “delay jitter" (also called “jitter") for a given flow is the difference between the worst-case delay and “best-case delay" (i.e., the minimum delay of all packets sent by the flow during its lifetime); it is referred to as IP Packet Delay Variation in RFC 3393 [40].

Most of the mechanisms proposed in time-sensitive networks such as a credit-based shaper, IEEE802.1 Qcr Asynchronous Traffic Shaping [41], and deficit round-robin [1] are designed to provide guarantees over worst-case delays. These mechanisms, on their own, do not provide low jitter as required by a set of applications that include industrial automation[5] and avionics [6], and by the emerging applications in large-scale time-sensitive networks, such as the industrial Internet of Things [8] and electricity distribution[9]. This issue can be addressed with dampers that are asynchronous mechanisms to reduce delay jitter [42, 43, 44]. A damper delays every time-sensitive packet by an amount written in a packet header field that carries an estimate of the earliness of this packet with respect to a known delay upper-bound of upstream systems. The main challenge with respect to dampers is the lack of a proper analysis; in the existing works, the analysis is done under limited assumptions on the network settings. Also, the existing analyses assume that the network operates with one ideal clock. Whereas, in practice, this assumption does not hold and can have non-negligible side effects [45].

In this thesis, we focus on the analysis of dampers, as asynchronous jitter-reduction mechanisms. The analysis of time-triggered mechanisms, such as cyclic queuing and forwarding [46] and time-aware shapers, [47] are out of the scope of this thesis.

1.2.4 In-Order Packet Delivery

Time-sensitive networks permit some limited amount of packet reordering. This can occur due to parallelism in network elements such as switches and routers, the routing of packets via different paths, or packet duplication [48, 49, 50]. The IETF DetNet states in [37] that the amount of reordering is a key quality-of-service attribute of a flow; but neither IETF DetNet nor IEEE TSN specify the meaning of this in detail. If a time-sensitive flow is subject to possible reordering and the application requires in-order packet delivery, a re-sequencing buffer is used to restore packet order [51]. The buffer is typically placed at the final destination, but it is also proposed in [37] to be placed at intermediate points inside the network. This would be done, for example, when the network path between the re-sequencing buffer and the destination preserves order, or simply in order to reduce the amount of reordering.

A re-sequencing buffer stores the early packets until all packets with smaller sequence numbers arrive [51, 52, 53, 54]. A timer is used to limit the waiting time of a packet in the re-sequencing buffer because, otherwise, the loss of a packet in the network would cause indefinite holding of the packets with larger sequence numbers. A challenge in this context is to analyze the effect of placing re-sequencing buffers on the computation of worst-case delay and jitter. Surprisingly, such computations are currently done without taking into account the effect of re-sequencing buffers. Therefore, in this thesis, we fill this gap.

1.2.5 Seamless Redundancy

Due to the criticality of the applications in time-sensitive networks, redundancy mechanisms are provided in order to recover from potential network failures such as broken transmission links. The existing mechanisms are the high-availability seamless redundancy (HSR) and parallel redundancy protocol (PRP) [55]. In both mechanisms, a source replicates any packet and sends it through all of its ports; then the replicates are eliminated at destination. Newer methods, i.e., IEEE802.1CB frame replication and elimination for redundancy[56] and DetNet's packet replication and elimination functions (PREF) [57], suggest to replicate and to eliminate packets also at intermediate switches or routers.

A challenge of such redundancy protocols is knowing how to analyze their effect on other metrics, i.e., bounds on delay, jitter, backlog, and the amount of packet reordering: As a packet might be replicated and eliminated at multiple points, each replica experiences different delays. At the intermediate points, this in turn causes the flows to have increased burstiness, which ultimately leads to an increase of worst-case delay, jitter, and backlog; also, as a direct side-effect of such delay variations, the order of packets might not be preserved. An analysis of time-sensitive networks in the presence of such redundancy protocols is done in [58] and is out of the scope of this thesis.

1.3 Contributions

In this dissertation, we provide a network-calculus toolbox for analyzing asynchronous mechanisms in time-sensitive networks, and we address the challenges mentioned in the previous section, in particular the following:

1. We derive novel delay bounds for flows with packet-level constraints; these flows share a FIFO system with a known service curve and a fixed transmission rate. Our bounds dominate the existing ones in time-sensitive networks; these bounds are obtained by deriving bit-level arrival curve from packet-level constraints.

2. Using this method of proof, we obtain improved delay bounds for flows with other regulation constraints, i.e., bit-level arrival curves (e.g., token bucket) and g-regulation (e.g., length-rate quotient).

3. We derive and formally prove credit upper-bounds for the credit-based shaper (CBS) in a TSN scheduler, in the presence of a higher priority class for control-data traffic; our bounds improves on the existing bounds.

4. Using the computed credit bounds, we obtain a service curve for every AVB (audio-video bridging) class at a CBS system in a TSN scheduler.

5. We obtain delay and backlog bounds for an interleaved regulator that is followed by a FIFO system with CBS. We show numerically that the obtained delay bound is tight.

6. We show that one of the metrics in RFC4737 [59], the reordering late time-offset (RTO), equals the minimal timeout value for a re-sequencing buffer such that no packet is lost due to spurious timeout and such that packets are delivered in-order.

7. We show that another metric in RFC4737, the reordering byte offset (RBO), is equal to the required size of the re-sequencing buffer when the network can be assumed to be lossless. Otherwise, we show that, if packet losses cannot be ignored, the RBO underestimates the required buffer size for which we give a formula that involves the RTO.

8. We show that, if a flow is lossless between its source and a re-sequencing buffer, the worst-case delay and the delay jitter are not increased by the re-sequencing buffer (i.e., re-sequencing is ”for free“ in terms of delay in the lossless case). In contrast, if the flow can be subject to packet losses on its path from the source to the re-sequencing buffer, then the worst-case delay can be increased by an amount up to the timeout value of the re-sequencing buffer.
9. We evaluate the value of re-sequencing buffers at intermediate points, in addition to the destination. We find that the placement of re-sequencing buffers— after every non-order-preserving element (typically a switching fabric)— does not improve the worst-case delay or the delay jitter if the network is lossless. But, in the presence of network losses, it does reduce the worst-case delay, delay jitter, and RTO at the destination.
10. We present a taxonomy of dampers that classifies the existing implementations as dampers with or without FIFO constraint.
11. We provide formulas for computing tight delay and jitter bounds for dampers without FIFO constraints under general network configuration with non-ideal clocks. We see that the effect of non-ideal clocks can be non-negligible in cases with low jitter requirements. As a result of this analysis, we derive conditions in which clock synchronization throughout a network does not affect the performance of dampers.
12. We capture the propagation of arrival curve at the output of dampers and see how dampers can solve the burstiness cascade issue.
13. We show that existing implementations of dampers without FIFO constraints might cause undesired packet reordering due to clock non-idealities, even in synchronized networks. This problem is avoided with dampers that enforce FIFO constraints.
14. We model two classes of dampers with FIFO constraints: re-sequencing dampers and head-of-line dampers. For the former class, we show that when all network elements are FIFO, the delay and jitter bounds are not affected by the re-sequencing operation. For the latter class, there is a small penalty, which we quantify exactly, due to head-of-line queuing. In contrast, if some network elements are non-FIFO, the jitter bounds for dampers with FIFO constraints can be considerably larger.

1.4 Roadmap

This thesis is organized as follows.

In Chapter 2, we explore the literature on the analysis of various aspects of time-sensitive networks such as traffic specification, scheduling mechanisms, and traffic regulation and dampers. In Chapter 3, we provide the necessary mathematical background, as well as network-calculus concepts and existing results.

In Chapter 4, we present a novel delay bound in time-sensitive networks; it improves the delay bound obtained by classic network-calculus. The main result of Chapter 4 is the derivation of delay bounds for flows with packet-level constraints that are aligned to the traffic specification in time-sensitive networks. Currently, an approach for obtaining a delay bound is to derive a bit-level arrival curve from a packet-level constraint then to apply the classic network-calculus delay bound. We show that such a method is not tight and that better bounds can be obtained by directly exploiting the arrival curves expressed at the packet level. Our analysis method also obtains better bounds when flows are constrained with bit-level arrival curve and g-regulation, such as the recently proposed length-rate quotient rule. The content of this chapter was published in [60, 61].

In Chapter 5, we study an asynchronous configuration of IEEE TSN scheduling with CBSs and in the presence of interleaved regulators. In this chapter, we first derive and formally obtain novel credit upper-bounds for the CBS; our bounds improve on existing bounds. Then, by applying the obtained credit bounds, we derive a service-curve characterization for a CBS and obtain a per-flow bound for the response time of the CBS. We also compute a per-flow bound on the response time of an interleaved regulator. Using the above results, we compute bounds on the per-class backlogs. Then, we use the newly computed delay bounds, along with recent results on interleaved regulators from the literature, to derive tight end-to-end latency bounds and to show that these are less than the sums of per-switch delay bounds. The content of this chapter was published in [62, 63].

In Chapter 6, we analyze time-sensitive networks for when in-order packet delivery is required. This is typically done by the means of re-sequencing buffers with two parameters, time-out value and buffer size. First, we show that a previously proposed per-flow metric, reordering late time-offset (RTO), determines the timeout value. If the network is lossless, another previously defined metric, the reordering byte offset (RBO), determines the required buffer. If packet losses cannot be ignored, the required buffer might be larger than RBO and depends on jitter, an arrival curve of the flow at its source, and on the timeout. Then we develop a calculus to compute the RTO for a flow path: the method uses a novel relation with jitter and arrival curve, together with a decomposition of the path into non-order-preserving and order-preserving elements. We also analyze the effect of re-sequencing buffers on worst-case delay, jitter, and the propagation of arrival curves. We show in particular that, in a lossless (but non order-preserving) network, re-sequencing is “for free”, in other words, it does not increase worst-case delay nor jitter; whereas, in a lossy network, re-sequencing increases the worst-case delay and jitter. We apply the analysis to evaluate the performance impact of placing re-sequencing buffers at intermediate points and illustrate the results on two industrial test cases. The content of this chapter was published in [64].

In Chapter 7, we focus on dampers and provide a taxonomy of all existing implementations in general network settings and analyze their timing properties in the presence of non-ideal clocks. We give, in particular, formulas for computing residual jitter bounds of networks with dampers of any kind. We show that non-FIFO dampers can cause reordering due to clock non-idealities and that the combination of FIFO dampers with non-FIFO network elements can very negatively affect the performance bounds. Our results can be used to analyze timing properties and burstiness increase in any time-sensitive networks, as we illustrate on an industrial case-study. The content of this chapter was published in [65].

We conclude this thesis in Chapter 8 by summarizing our results. The proofs of theorems, propositions, and lemmas are given in the appendices.

2 Time-Sensitive Networks

The science of today is the technology of tomorrow.
— Edward Teller

As mentioned in Chapter 1, the applications in time-sensitive networks require guarantees in terms of delay and delay-jitter bounds, zero congestion-loss and in-order packet delivery. Such guarantees are different across applications[66]: For example, audio and video applications require delay bounds of, respectively, 2 ms and 50 ms with a jitter not more than 100 μs for a proper interactive communication, and in the automotive industry, the requirement on delay bounds is sub-milliseconds and on the delay jitter is a few microseconds. Therefore, time-sensitive networks should be able to provide a wide range of QoS guarantees.

Being able to provide such guarantees depends on the behavior of the flow sources and the intermediate systems that are in charge of forwarding the flows of data to their destinations. In time-sensitive networks, a flow is characterized with certain *traffic specification* (TSPEC), e.g., the classic token-bucket and periodic data-generator. The coexistence of numerous flows raises the need to share the network resources, in particular the physical transmission line. This leads to queuing within intermediate systems (also sources, if they generate multiple flows) that, in turn, affect the end-to-end delay. The queuing inside intermediate systems is often per aggregate rather than per flow. To arbitrate among the flows, a scheduling mechanism is placed before the shared medium to decide which queue can use the link. Strict-priority and round-robin scheduling, as well as weighted fair queuing, are examples of such scheduling mechanisms. As mentioned in Chapter 1, the burstiness cascade and the circular dependencies make the computation of delay bounds difficult in networks with aggregate queuing. To avoid them, a solution is to regulate flows at intermediate systems [67, 43].

A regulator, called a shaper in RFC 2475 [68][1], delays some or all of the packets of a flow in

---

[1] In time-sensitive networks, the word "shaper" is commonly used in some scheduling mechanisms such as "credit-based shaper“ and "time-aware shaper“, which do not perform as shapers defined in RFC2475. Moreover, the term "regulator” has been used frequently in the literature especially with the emergence of "interleaved

order to bring it into compliance with a traffic profile, typically the same as the flow TSPEC. A damper delays the packets of the flows in order to preserve their inter-spacing hence is able to provide low jitter. Therefore, regulators and dampers break the cyclic dependencies and avoid the burstiness cascade, and facilitate achieving the QoS guarantees in time-sensitive networks.

In this chapter, we provide an overview of the most common TSPEC and scheduling mechanisms, as well as traffic regulation and dampers in time-sensitive networks.

2.1 Traffic Specification

Token (leaky) bucket is among the first regulation considered for real-time application, such as audio and video, in communication networks[69, 27, 70]. A token-bucket regulator has two parameters, rate r and bucket size b. It is often seen as a bucket with size b that is filled with tokens by rate r. A source with token-bucket TSPEC can generate traffic only as much as the amount of tokens stored in the bucket. Parallel to the token-bucket regulator, the generic cell rate algorithm (GCRA) was used within ATM[2] networks [71], and has two parameters: drain rate r, and bucket depth b. In ATM networks, a flow is characterized by a peak cell rate, a sustainable cell rate, a maximum burst size and a cell delay variation tolerance [72, 73]. Then the source of a flow is tested with two GCRAs: The first is with r= peak cell rate and b=cell delay variation tolerance, and the second is with r=sustainable cell rate and b=burst tolerance+cell delay variation tolerance. The integrated service (IntServ) unifies the token bucket and GCRA, and it defines the general TSPEC with four tuples, burst tolerance b, sustainable rate r, peak rate p and maximum packet size L [74]. The IntServ TSPEC, called a dual token-bucket in [23], can be interpreted as two token buckets, where the first has size L and rate p, and the second has size b and rate r.

In industrial communication, as well as in the automotive and aerospace industries, sources of flows can generate periodic messages such as command and control data or non-periodic messages, e.g., even-based control, alarms and warnings [33, 32, 75, 66, 76, 77, 78, 79]. The message periods are different based on application, e.g., parametric data for fly-by-wire or military sensing in aerospace has a period in range 1 – 10 ms, the safety-relevant commands in automotive network are sent with a period within 1 – 20 ms, and isochronous traffic in industrial communication is generated within a period less than 2 ms.

IEEE TSN unifies the TSPEC from various industries, by using three terms: MaxIntervalFrame (MIF), MaxFrameSize (MFS), and Class Measurement Interval (CMI) [35, Section 34.6.1]. According to TSN TSPEC, a *talker* (the TSN term for a source) can generate up to MIF frames with a maximum size of MFS, within each duration equal to a CMI. In IETF DetNet, the TSPEC is defined with more terms due to a vast variety of applications that DetNet intends to cover[7]. The DetNet TSPEC consists in Interval, MaxPacketsPerInterval (MaPPI), MaxPayloadSize

---

regulators“. Hence to avoid possible confusion regarding the term "shaper“, we decided to use “regulator“.

[2] Asynchronous Transfer Mode

(MaPS), MinPayloadSize (MiPS), and MinPacketsPerInterval(MiPPI) [80]. The Interval is the period of time in which the TSPEC is specified; MaPS [resp., MiPS] is the maximum [resp., minimum] size of packet that is sent within one Interval; MaPPI [resp. MiPPI] the maximum [resp., minimum] number of packets that is sent within one Interval.

As we saw, the TSPEC in time-sensitive networks can be periodic or non-periodic and at the bit level or packet level. In this thesis, we map the TSPEC to arrival curves in order to perform our analysis. An arrival curve can be at the bit or packet level, hence making a proper choice affects the delay bounds, as we will explore more in Chapter 4.

2.2 Scheduling Mechanisms

The mechanisms in time-sensitive networks can be categorized into two main families of *synchronous* (time-driven) and *asynchronous* (event-driven) schedulers. The synchronous schedulers perform the arbitration among the queues, based on a known time schedule, e.g., IEEE802.1Qbv Time-Aware Shaper (TAS) [81] and IEEE802.1Qch Cyclic Queuing and Forwarding (CQF)[46]. TAS has a number of gates. Each gate is associated with a queue, permits packets to pass when opened, and blocks packets when closed. The states of the gates are determined by the gate control list (GCL). CQF (also known as a peristaltic shaper) is a two-buffer scheduling mechanism; the buffers operate in cyclic manner: At any cycle time, one buffer accumulates the received packets and the other one transmits the already stored packets from the previous cycle. The time-dependency of such schedulers requires time synchronization and global scheduling throughout network in order to achieve end-to-end guarantees. The asynchronous schedulers such as a credit-based shaper (CBS), strict priority (SP) scheduler, and a deficit round-robin (DRR), run independent of time schedules. Their operations rely on packet arrivals/departures, as well as scheduler parameters and variables, e.g., an idleslope and credit value for CBS, a priority value for SP, and a quantum value for DRR. In this thesis, we focus on asynchronous mechanisms. Further studies of synchronous mechanisms are beyond the scope of this thesis and can be found in [82, 83, 84, 85].

In the rest of this section, we explore the most common asynchronous scheduling mechanisms in time-sensitive networks. From the network-calculus perspective, we abstract these mechanisms by using service curves (see Section 3.2), as we will see in Chapter 5.

2.2.1 Strict Priority

Strict-Priority (SP) scheduling is among the simplest schedulers used in time-sensitive networks and has been extensively studied [86, 87, 88, 89]. In this scheme, each queue is associated with a priority level. Whenever one or multiple queues have packets to transmit, the SP scheduler selects the one with highest priority to send its head packet. This process is repeated at the end of each packet transmission, until all queues become empty. This is referred to as a non-preemptive SP, as the transmission of a packet is not interrupted until it is completed.

On the contrary, a preemptive SP interrupts the packet transmission, when a queue of higher priority has packets to send, and then starts the transmission of the head packet from the higher priority queue.

The SP scheduling enforces a priority queue to wait until all the higher priority queues (if any) are empty. Therefore, providing guarantees for a priority queue, except the highest, is dependent on the input traffic of higher priority queues. Such guarantees are only possible when the traffic of higher priority queues is regulated. This is the reason that an SP is often used in combination with other mechanisms such as CBS and BLS (see Section 2.2.3).

2.2.2 Fair Queuing

Generalized processor sharing (GPS) is a theoretical scheduling policy proposed for providing minimum rate guarantees [90, 91, 92]. With the GPS, each queue is assigned a *weight* used to tune the minimum rate offered to each queue. The rate guarantees provided by the GPS are based on the assumption of a fluid flow of data; in practice, this assumption does not hold as flows are often packetized. Therefore, several packet-based approximations of this policy are proposed, among which is the packetized GPS (PGPS), also known as weighted fair queuing.

PGPS operates as follows. When the server is ready to transmit the next packet (either at the start of a backlog or at the end of a packet transmission), a theoretical *finish time* is computed based on the GPS policy for all packets in the queues and in the absence of future arrivals. Then, the scheduler selects a packet, with smallest finish time among all packets, for transmission. Compared to the GPS, the PGPS increases the worst-case delay, only by the transmission of one maximum packet size [91, Theorem 1]. Even though the PGPS is a practical approximation of the GPS, its implementation is still hard due to the complexity of selecting a packet with smallest finish time [93].

Round-robin (RR) scheduling resembles the GPS policy and provides minimum rate guarantees among the queuing systems [94, 95]. The original RR scheduling arbitrates among the queues in a circular pattern and gives a fixed time to each queue with packets to send them. When queues are subject to addressing various QoS requirements, a proper arbitration is to treat them differently by allocating weights; the scheme that is referred as Weighted RR (WRR) [96, 97]. The classic WRR assigns a *weight*, in unit of packet, to each queue. When a queue is selected for transmission, the scheduler sends packets up to the assigned weight or the end of queue. This behavior of WRR makes it a simple work-conserving scheduler. However, multiple packets (up to the weight) can be served consecutively for each queue, which causes a large worst-case waiting time for other queues [98]. Interleaved WRR (IWRR) is an implementation of WRR that avoids this issue [96]. With IWRR, each cycle is split into a number of rounds equal to the maximum of the weights assigned to the queues. At any round r, a queue can send one packet, only if r is less than its weight. Compared to WRR, IWRR has smaller packet bursts, which causes smaller worst-case delays for the queues. As both WRR and IWRR assign weights in units of packets, these schedulers are good approximations of the GPS in providing minimum rate allocations, but only when the packets are of the same size, e.g., in classic ATM networks. In time-sensitive networks, this assumption often does not hold as packets can have variable lengths. The deficit RR (DRR) scheduling [99] is proposed to address this limitation of WRR and IWRR, by keeping track of the transmitted traffic, at the byte level, for each queue.

In DRR, each queue has a parameter *quantum* and a variable *deficit counter*, both in bytes. At each cycle, when a queue is not empty, its deficit counter is increased by the quantum (the value of deficit counter expresses the maximum amount of data, in bytes, that a queue can send at each cycle). Then, if the size of the head packet is not larger than the deficit counter, it is emitted and the deficit counter is decreased by the size of the packet. This continues until the deficit counter is smaller than the size of the head packet and then the scheduler moves to the next queue. The deficit counter of a queue remains for the next cycle, except when the queue becomes empty that resets the deficit counter to zero.

Among the RR scheduling mechanisms, DRR catches the most attention in time-sensitive networks due to adaptive rate guarantees among the queues, by means of the quantum values, and $O(1)$ complexity (under the condition that the quantum value of each queue is larger than the maximum packet length). However, achieving low delay bounds and a proper tuning of the quantum values are the known issues [100]. A worst-case analysis of DRR has been done extensively in literature [100, 101, 1], among which [1] gives the most recent results using network calculus.

2.2.3 Credit-Based Mechanisms

As mentioned in Section 2.2.1, SP scheduling can provide guarantees only when the traffic of higher priority queues is well-behaved. To this end, mechanisms such as CBS and BLS are proposed to make the guarantees for a priority queue, in SP scheduling, independent of the input traffic of the higher priority queues. We explore them hereafter.

Credit-Based Shaper

In time-sensitive networks, CBSs are initially proposed in IEEE802.1BA AVB [18] to control the guaranteed rates given to the two highest priority classes that are dedicated to audio and video traffic. CBSs are placed behind the SP scheduler. For any queue with a CBS, packets are sent according to the available *credit* for that queue. Each CBS is assigned two parameters, *idleslope* and *sendslope*, and it has a counter to keep track of the available credit [35, Annex L]: The credit of a queue increases with the rate of the idleslope when the queue is non-empty and the other queues with CBSs transmit packets; the credit decreases with the rate of the sendslope when the queue is sending packets. A detailed description of CBS is covered in Chapter 5.

There are a number of works on the analysis of CBS [13, 102, 103, 104, 79, 105, 106]. Among them, some use network calculus to perform the analysis: [13] is the primary work on the analysis of CBS in IEEE802.1BA AVB [18]; [103, 79] perform the analysis on a IEEE TSN scheduler, in which the highest priority class is without a CBS; and [104] extend the analysis in the presence of multiple queues with CBSs in a TSN scheduler.

Burst-Limiting Shaper

BLSs are primarily presented to be used for the highest priority class, namely Control Data Traffic (CDT), in IEEE TSN [107]. Their purpose is to provide fairness among CDT and lower priority classes, such as audio and video, while still guaranteeing a low latency to the CDT class. Differently from the CBS, the BLS alternates the priority of the queues, seen by the non-preemptive SP, between the highest value and a lower value. This alternation is done based on the amount of BLS's credit and on its low- and high-threshold parameters: If the credit reaches the high threshold, the queue is given the low priority, and if it reaches the low threshold, the priority of the queue is set to high; in other cases, the priority is unchanged.

The credit evolution in BLS is as follows. The credit of a queue increases with the rate of the *sendslope*, when the queue transmits packets; it stops when the credit reaches the high threshold, and stays at this level until the end of transmission of the current packet. If the queue does not transmit packets, its credit is decreased with the rate of the *idleslope* until it reaches 0. Note that the priority change occurs as soon as the credit value reaches the lower threshold, a further decrease is often due to an ongoing packet transmission of other queues. We see that the credit in BLS evolves in the opposite manner than in CBS: In BLS, sending packets results in an increase of credit, whereas in CBS it leads to a credit reduction.

An analysis of BLS is conducted in several studies [108, 109, 110, 111]. [108] presents the very first results on the performance of BLS; a formal analysis of BLS is made in [109], assuming zero interference from higher priority queue. A more general, yet formal, analysis is performed by [110]. [111, 112] analyze BLS in the presence of a mixture of critical flows by using a network-calculus approach. [113, 114] provide the timing analysis for multiple queues with BLSs.

2.2.4 TSN and DetNet

IEEE TSN specifies a number of per-class scheduling mechanisms, called Transmission Selection Algorithm in [35]. In the current version of the IEEE 802.1Q document [35], there are three scheduling algorithms[3], i.e., SP, CBS, and Enhanced Transmission Selection (ETS), and one enhancement to be used for time-triggered traffic. The ETS is added to the document in order to provide an allocation of bandwidth to less sensitive traffic classes with larger bandwidth requirements so that these classes can coexist with low delay traffic (which use SP and CBS). The ETS includes any scheduling mechanisms, such as the family of fair queuing

---

[3] These mechanisms are agreed to be a part of TSN document. The working group still permits the vendors to implement their own scheduling mechanism, as stated in [35, Table 8.6].

(Section 2.2.2), able to provide the required performance guarantees. The TSN enhancement for time-triggered traffic uses the IEEE Time-Aware Shaper that is a synchronous mechanism. This TSN enhancement is called "enhancement" because a set of TAS is added in front of the aforementioned scheduling mechanisms to enable transmission of the traffic scheduled relative to a known timescale.

Contrary to IEEE TSN, the DetNet working group of IETF does not focus on the standardization of specific queuing and forwarding mechanisms; rather, the group discusses the requirements for the forwarding behavior of DetNet systems[21].

2.3 Flow Regulation and Dampers

Zhang and Ferrari propose the concept of regulators as a part of the rate-control service discipline (RCSD) for real-time flows [115]. Each regulator in RCSD scheme monitors one flow and forces it to follow a desired traffic pattern. An end-to-end delay bound is then computed by summing up the delay bound of each individual system in a flow path. Then, it is proved in [116] that using per-flow regulation leads to improvement over the bounds obtained by [115]. A first formal analysis on regulators was done by Cruz, with a focus on token-bucket regulators [30]. A generalization of the flow regulators is then defined within network calculus as a bit-processing device that forces its output to conform an arrival curve [117, 118]. Later, it was shown that obtaining end-to-end delay bounds in a network with aggregate scheduling is difficult unless flows are regulated [67]. In time-sensitive networks, flow regulation is used as a method for addressing the issue of burstiness cascade and for avoiding cyclic dependency (discussed in Chapter 1) and is studied in various domains, e.g., embedded real-time systems [119], automotive communication [120], and avionic communication [121].

Traffic regulation using per-flow queues, as in the original RCSD scheme, has scalability issues in time-sensitive networks. This was the inspiration for the introduction of the urgency-based shaper (UBS) as a per-flow regulator with aggregate queuing [122]. The UBS is standardized within TSN under the name of IEEE802.1Qcr Asynchronous Traffic Shaping (ATS) [41]. The analysis of UBS in the original paper is limited to two regulation algorithms: token bucket and length-rate quotient (Section 4.2.2). UBS, as well as ATS, are formally analyzed under a general family of *interleaved regulators* (IR) [123]. With IR, the packet at the head of the queue is regulated based on its regulation constraints (e.g., token bucket); the packet is released at the earliest time possible, without violating the constraint. One key feature of the IRs is their "for-free" property, i.e., when an interleaved regulator is appended to a FIFO subsystem, it does not increase the worst-case delay of the latter [123, Theorem 5]. Using IRs within time-sensitive networks is a sound choice, as the IRs address the burstiness cascade and cyclic dependency issues with aggregate queuing (instead of per-flow queuing) and they have the "for free" property.

Dampers are introduced to reduce delay jitter in time-sensitive networks [42, 43, 44]. A damper delays every packet by an amount written in a packet-header field, called the damper header:

The damper header of a packet is an estimate of the earliness of this packet with respect to a known delay upper-bound of upstream systems. Similarly to IRs, dampers address the burstiness cascade and cyclic dependency issues. Unlike IRs, the dampers do not need any flow-state information at the intermediate systems as the required information is carried over in the packet header. This makes dampers a more scalable solution for time-sensitive networks. Using dampers or IRs adds hardware complexity to intermediate systems: Dampers require time stamping of packets at their arrival, and IRs need to modify the flow information for every packet. Furthermore, with dampers, a packet-level modification for the flows is also needed in order to carry the damper header. Chapter 7 gives details about dampers, including existing implementations.

2.4 Conclusion

In this chapter, we have explored the literature on the TSPECs of flows and scheduling mechanisms commonly used in time-sensitive networks. We have further investigated the flow regulation and dampers as asynchronous mechanisms for tackling the issues of burstiness cascade and cyclic dependencies. In this thesis, our analysis is based on network calculus, and uses the arrival-curve mappings for the TSPECs of flows and the service-curve characterizations for the scheduling mechanisms in systems within time-sensitive networks.

3 Technical Background

Mathematics is the queen of science, and arithmetic the queen of mathematics.
— Carl Friedrich Gauss

In this chapter, we cover the general technical concepts in order to follow the subsequent chapters of this thesis. It consists, in particular, in mathematical concepts such as pseudo-inverse functions and a general overview of min-plus algebra as the foundation of network calculus. Finally, we provide a summary of network-calculus, including the most important concepts and results used throughout this thesis. The necessary notation and symbols used in this chapter are given in Section 3.4.

3.1 Mathematical Foundation

In this thesis, we often use wide-sense increasing functions. A function w is wide-sense increasing if and only if $\forall s \le t \,:\, w(s) \le w(t)$. Then, $\mathscr{F}^0_{\text{inc}}$ is the set of wide-sense increasing functions $w : [0,+\infty) \to [0,+\infty]$ such that $w(0) = 0$.

3.1.1 Pseudo-inverse Functions

Consider a function $w \in \mathscr{F}^0_{\text{inc}}$. Then, the lower pseudo-inverse of w is $w^{\downarrow}$, defined as

$$w^{\downarrow}(x) \overset{\text{def}}{=} \inf\{s \ge 0 \mid w(s) \ge x\} = \sup\{s \ge 0 \mid w(s) < x\}. \tag{3.1}$$

and the upper pseudo-inverses is $w^{\uparrow}$, defined as

$$w^{\uparrow}(x) \overset{\text{def}}{=} \sup\{s \ge 0 \mid w(s) \le x\} = \inf\{s \ge 0 \mid w(s) > x\}. \tag{3.2}$$

Figure 3.1 illustrates the pseudo-inverse functions and the differences between them. The pseudo-inverse functions are obtained by flipping the graph of w around the line $y = x$.

The resulting graph does not correspond to a function as the plateau part of w i.e., x_1 to x_2, causes ambiguity. With the lower pseudo-inverse (left figure), $w^{\downarrow}$, the ambiguity is resolved by selecting the infimum, i.e., $w^{\downarrow}(y_1) = x_1$. With the upper pseudo-inverse (right figure), $w^{\uparrow}$, the ambiguity is resolved by selecting the supremum, i.e., $w^{\uparrow}(y_1) = x_2$.

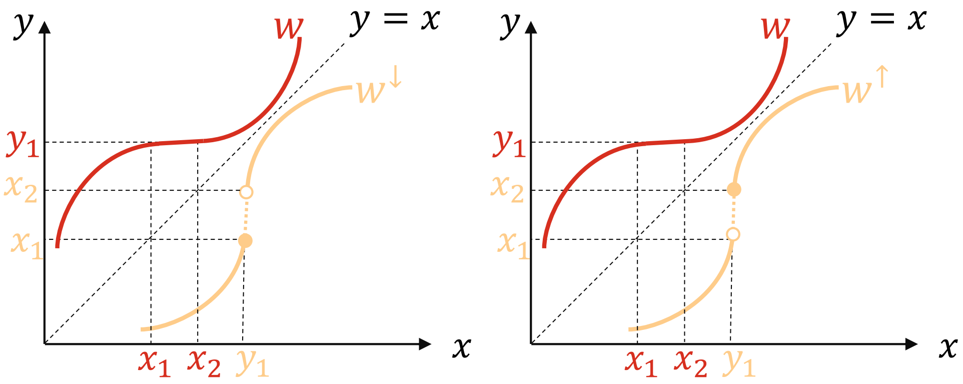


Figure 3.1: Illustration of pseudo-inverse functions.

By [124, Section 10.1]:

- $w^{\downarrow}$ is non-decreasing and left continuous.
- $w^{\uparrow}$ is non-decreasing and right continuous.

Some of the common functions and their lower-pseudo inverses are

$$\begin{aligned} F(t) = rt + b, t > 0; F(0) = 0 &\implies F^{\downarrow}(x) = F^{\uparrow}(x) = \left[\frac{x-b}{r}\right]^{+}, \\ F(t) = b\lceil\frac{t}{\tau}\rceil &\implies F^{\downarrow}(x) = \tau\lceil\frac{x-b}{b}\rceil, x > 0; F^{\downarrow}(0) = 0, \;\; F^{\uparrow}(x) = \tau\lfloor\frac{x}{b}\rfloor, \\ F(t) = R[t-T]^{+} &\implies F^{\downarrow}(x) = T + \frac{x}{R}, x > 0; F^{\downarrow}(0) = 0, \;\; F^{\uparrow}(x) = T + \frac{x}{R}. \end{aligned} \tag{3.3}$$

3.1.2 Min-plus Algebra

A min-plus algebra refers to a dioid $(\mathbb{R} \cup \{+\infty\}, \wedge, +)$ [25, Chapter 3.1][124, Chapter 9], where the operations are the minimum ($\wedge$) and the addition ($+$), and $\mathbb{R}$ is the set of real numbers. In network calculus, the commonly used operations of min-plus algebra are the convolution and deconvolution of functions that are defined as follows.

Definition 3.1 (Min-plus Convolution and Deconvolution)**.** *Consider two functions* $w, w' \in$

$\mathscr{F}_{\text{inc}}^0$. *The min-plus convolution and deconvolution of* w *and* w' *are respectively defined as*

$$(w \otimes w')(t) \overset{\text{def}}{=} \inf_{0 \le s \le t} \{w(t-s) + w'(s)\}, \tag{3.4}$$

$$(w \oslash w')(t) \overset{\text{def}}{=} \sup_{s \ge 0} \{w(t+s) - w'(s)\}, \tag{3.5}$$

for all $t \ge 0$.

The sub-additive functions are another set of important functions used in network calculus. A function $w \in \mathscr{F}_{\text{inc}}^0$ is sub-additive if and only if

$$w(t) + w(s) \ge w(t+s) \qquad \forall s, t \ge 0. \tag{3.6}$$

For a given function $w \in \mathscr{F}_{\text{inc}}^0$, $0 \le w \otimes w \le w$. The sequential application of min-plus convolution on w, tends to decrease the function each time until it converges to some function referred to as sub-additive closure that is defined formally as follows.

Definition 3.2 (Sub-additive closure)**.** *Consider a function* $w \in \mathscr{F}_{\text{inc}}^0$. *Then, the sub-additive closure of* w *is defined as*

$$\bar{w} \overset{\text{def}}{=} \delta_0 \wedge w \wedge (w \otimes w) \wedge (w \otimes w \otimes w) \wedge \ldots, \tag{3.7}$$

where δ_0 *is the impulse function with* $T_0 = 0$ *defined as*

$$\delta_{T_0}(t) = \begin{cases} 0 & t \le T_0 \\ +\infty & t > 0 \end{cases}. \tag{3.8}$$

Two important quantities in network calculus are the maximal vertical and horizontal deviations between two wide-sense increasing functions. The mathematical definitions for these quantities are as follows [25] and illustrated in Figure 3.2.

Definition 3.3 (Vertical and horizontal deviations)**.** *Consider two functions* $w, w' \in \mathscr{F}_{\text{inc}}^0$. *The vertical deviation* $v(w', w)$ *and the horizontal deviation* $h(w', w)$ *are defined as*

$$v(w', w) \overset{\text{def}}{=} \sup_{t \ge 0} \{w'(t) - w(t)\}, \tag{3.9}$$

$$h(w', w) \overset{\text{def}}{=} \sup_{t \ge 0} \left\{ w^{\downarrow}\left(w'(t)\right) - t \right\}. \tag{3.10}$$

The vertical and horizontal deviations can be expressed with min-plus deconvolution. Specifi-

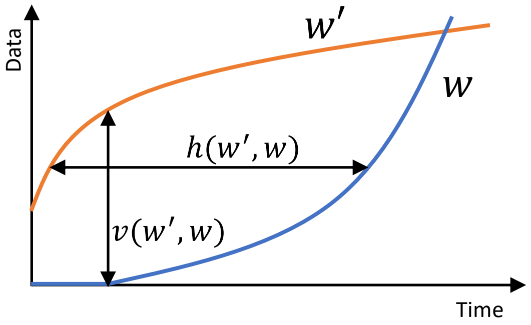


Figure 3.2: The horizontal and vertical deviations from w' to w.

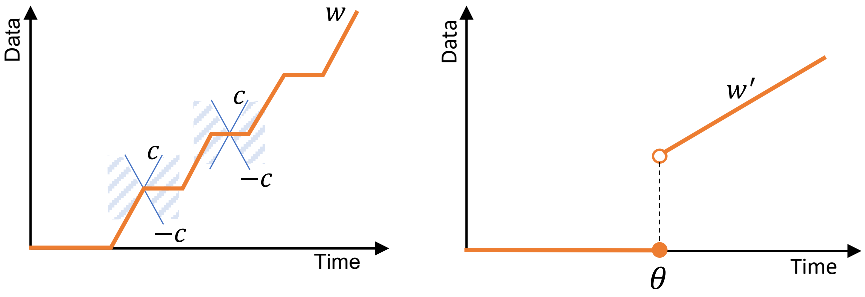


Figure 3.3: A c-Lipschitz function w (left, DRR service curve in [1]) and a non c-Lipschitz function w' (FIFO residual service curve in [2, Section 7.3.1]). The slope of the function is within $[-c, c]$ at any point in time for w. The function w' is not continuous at time θ.

cally,

$$v(w', w) = (w' \oslash w)(0), \tag{3.11}$$

$$h(w', w) = \left((w'^{\downarrow} \circ w) \oslash \lambda_1\right)(0), \tag{3.12}$$

where $\lambda_1(t) = t$.

3.1.3 Additional Definitions

Definition 3.4. *A function $w \in \mathscr{F}^0_{\text{inc}}$ is called c-Lipschitz (for $c \geq 0$) if $\forall\, t_1, t_2 \in \mathbb{R}^+$ [125, Section 41.5]:*

$$|w(t_2) - w(t_1)| \leq c|t_2 - t_1|. \tag{3.13}$$

A c-Lipschitz function is necessarily continuous, and the slope of the function is within $[-c, c]$ at any point in time. Fig. 3.3 shows examples of a c-Lipschitz and non-c-Lipschitz functions.

For $w \in \mathscr{F}^0_{\text{inc}}$, w^+ and w^- are the right and left limits, defined as

$$\forall x \in \mathbb{R}^+ \ : w^+(x) = \lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} w(x+\varepsilon), \tag{3.14}$$

$$\forall x \in \mathbb{R}^+ \ : w^-(x) = \lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} w(x-\varepsilon). \tag{3.15}$$

3.2 Network Calculus

Network calculus is a theory designed to compute backlog and delay bounds in networks by the means of arrival and service curves. In this section, we first provide a summary of the basic concepts in network calculus; then, we present the main theorems for obtaining backlog and delay bounds for a system in isolation. Finally, we give a list of methods and techniques to compute end-to-end delay bounds within a network.

3.2.1 Flow Model

In network calculus, a common approach to modeling the data flows is to use cumulative functions [25]. A cumulative function $\mathscr{A}(t) \in \mathscr{F}^0_{\text{inc}}$ is the number of bits observed on the flow within an interval $[0,t)$. By definition, this function is necessarily wide-sense increasing. The cumulative functions are a general way to model flows at the bit level; however, a flow is often made of packets with specific lengths. To this end, network calculus defines the concept of *packetizers*. A packetizer is a server that groups the bits of a flow into packets. More specifically, it stores the bits of traffic until the last bit of the packet is arrived; then it releases the whole packet. Let $\mathscr{L}(n)$ be a cumulative length of the first n packets, i.e., $\mathscr{L}(n) = \sum_{k=1}^{n} l_k, \mathscr{L}(0) = 0$, where l_k is the length of packet k. Then an $\mathscr{L}$-packetizer for $\mathscr{A}$ is defined as [2, Section 8.1]:

$$P^{\mathscr{L}}(\mathscr{A})(0) = 0, \quad P^{\mathscr{L}}(\mathscr{A})(t) = \sup_{n \in \mathbb{N} \cup \{0\}} \{\mathscr{L}(n) \,|\, \exists u < t, \mathscr{A}(u) \geq \mathscr{L}(n)\} \quad \forall t > 0. \tag{3.16}$$

A flow $\mathscr{A}$ is $\mathscr{L}$-packetized if $\mathscr{A} = P^{\mathscr{L}}(\mathscr{A})$.

To model a packetized flow, another alternative is to use packet sequence (called marked point process in [126]). A packet sequence (A,L) consists of two sequences of variables $A = \{A_1, A_2, A_3, \ldots\}$ and $L = \{l_1, l_2, l_3, \ldots\}$, where A_n and l_n are the arrival time and the length of packet n.

For a packetized flow, these two models can be derived from each other. Assume a flow modeled as a packet sequence (A,L). Then, the corresponding $\mathscr{L}$-packetized cumulative function is $\mathscr{A}$, obtained as

$$\begin{aligned}
&\mathscr{A}(t) = \sum_{n\in\mathbb{N}} l_n 1_{\{A_n<t\}},\\
&\mathscr{L}(0) = 0,\ \mathscr{L}(n) = \sum_{k=1}^{n} l_k \ \ \forall n \in \mathbb{N}. \qquad (3.17)
\end{aligned}$$

Now consider an $\mathscr{L}$-packetized flow $\mathscr{A}$. Then, the corresponding packet sequence model is obtained as

$$\begin{aligned}
&A_n = \inf\{s \geq 0 \mid \mathscr{A}(s) \geq \mathscr{L}(n)\} = \mathscr{A}^{\downarrow}(\mathscr{L}(n)) \quad \forall n \in \mathbb{N},\\
&l_n = \mathscr{L}(n) - \mathscr{L}(n-1) \quad \forall n \in \mathbb{N}. \qquad (3.18)
\end{aligned}$$

Using the aforementioned models, we present the main building blocks of network calculus, i.e., arrival and service curves.

3.2.2 Arrival and Service Curves

In network calculus, flows are constrained by the mean of arrival curves, defined as below [25].

Definition 3.5 ((Bit-level) arrival curve)**.** *Consider a flow expressed as a cumulative function* $\mathscr{A} \in \mathscr{F}^0_{\text{inc}}$. *The flow is constrained by a bit-level arrival-curve* $\alpha \in \mathscr{F}^0_{\text{inc}}$ *if and only if*

$$\mathscr{A}(t) \leq (\mathscr{A} \otimes \alpha)(t) \qquad \forall t \geq 0. \qquad (3.19)$$

As shown in [123], for a packet sequence (A, L), the above definition of arrival curve is equivalent to

$$\sum_{k=m}^{n} l_k \leq \alpha^{+}(A_n - A_m) \qquad \forall m, n \in \mathbb{N}, m \leq n \qquad (3.20)$$

The above equation is also equivalent to

$$A_n - A_m \geq \alpha^{\downarrow}\Big(\sum_{k=m}^{n} l_k\Big) \qquad \forall m, n \in \mathbb{N}, m \leq n. \qquad (3.21)$$

Frequently used arrival-curves are [25, 1.2.2]

- the token-bucket (or leaky-bucket) arrival-curve with rate r and burst b, defined by $\alpha(t) = rt + b; t > 0$, $\alpha(0) = 0$;
- the staircase function with burst b and interval τ defined by $\alpha(t) = b\lceil\frac{t}{\tau}\rceil$. It expresses the constraint that the flow has at most b bits within any interval of duration τ.

Figure 3.4 illustrates the two arrival curves.

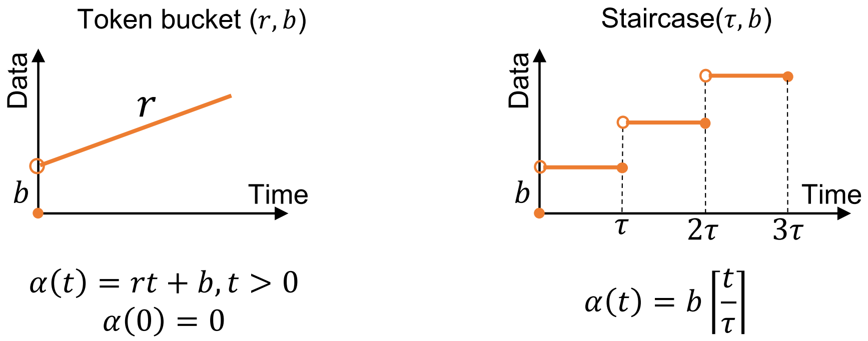


Figure 3.4: Token-bucket (r, b) and staircase (τ, b) arrival curves.

Whereas arrival curves are used to deliver information on the input traffic to a system, service curves provide an abstraction of the system and the service offered to the input traffic.

Definition 3.6 ((Minimum) Service Curve)**.** *Let* $\mathscr{A},\mathscr{D} \in \mathscr{F}^0_{\text{inc}}$ *be such that* $\mathscr{A}(t)$ *[resp.* $\mathscr{D}(t)$*] denotes the cumulative number of bits arrived in [resp. departed from] a system until time* t *(excluded). A function* $\beta \in \mathscr{F}^0_{\text{inc}}$ *is a (minimum) service curve for the system if*

$$\mathscr{D}(t) \geq \left(\mathscr{A} \otimes \beta\right)(t) \qquad \forall t \geq 0. \tag{3.22}$$

The above definition is equivalent to

$$\mathscr{D}(t) \geq \beta(t-s) + \mathscr{A}(s) \qquad \forall t \geq 0, \exists s \in [0, t]. \tag{3.23}$$

A service-curve characterization is available for many systems, see for example [103, 127, 111, 128, 129]. Rate-latency service curves are functions of the form $\beta(t) = R[t-T]^+$ with R, T being the rate and latency terms. A system that offers a rate-latency service curve can be interpreted as behaving, for the flows of interest, as if it would be a server with rate R and vacation T. The rate R is the rate guaranteed to the flow and is typically less than the line rate c. Rate-latency service curves are often used because of their simplicity, but better delay bounds can also be obtained with more complex service-curves [1]. Figure 3.5 shows a rate-latency service curve and the best existing service-curve for a deficit round-robin scheduler.

In the analysis of networks, we might encounter a number of systems where we know only an upper bound on the delay, e.g., switching fabric, and the network backbone. The question is how to abstract a service curve for such systems. Assume a system with delay bound D. Then, for this system we have

$$\mathscr{D}(t) \geq \mathscr{A}(t-D), \quad \forall t \geq 0. \tag{3.24}$$

The definition of the impulse function in 3.8 implies that $\mathscr{A}(t-D) = (\mathscr{A} \otimes \delta_D)(t)$, thus

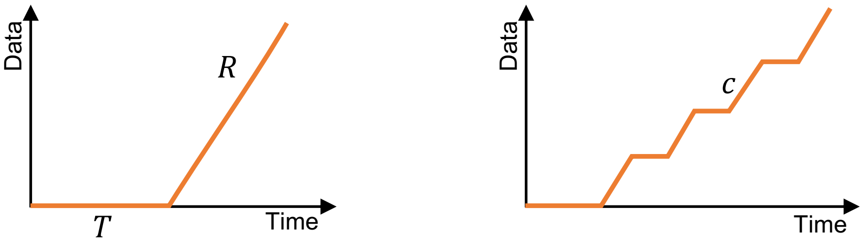


Figure 3.5: Rate-latency service curve with rate R and latency T (left), and a DRR service-curve connected to a physical link with rate c [1] (right).

$\mathscr{D} \geq \mathscr{A} \otimes \delta_D$; hence, δ_D is a service curve for this system.

The concept of strict service-curves is less general than the minimum service-curve. However, as we will see later, it is needed for some analysis. To define strict service-curves, we first need to define *backlog period*: an interval of time $(s,t]$ is a backlog period when $\mathscr{A}(s) = \mathscr{D}(s)$ and $\forall u \in (s,t] : \mathscr{A}(u) - \mathscr{D}(u) > 0$.

Definition 3.7 (Strict Service-Curve)**.** *Let $\mathscr{A}, \mathscr{D} \in \mathscr{F}^0_{\text{inc}}$ be such that $\mathscr{A}(t)$ [resp., $\mathscr{D}(t)$] denotes the cumulative number of bits arrived in [resp., departed from] a system until time t (excluded). A function $\beta \in \mathscr{F}^0_{\text{inc}}$ is a strict service-curve for the system if for any backlog period $(s,t]$, $\mathscr{D}(t) - \mathscr{A}(s) \geq \beta(t-s)$.*

One of the strongest results of network calculus is the service-curve concatenation for a tandem of systems [25].

Theorem 3.1 (Service-Curve Concatenation)**.** *Assume a flow traverses a sequence of M systems where system $i \in \{1,2,\dots,M\}$ offers a service curve $\beta_i \in \mathscr{F}^0_{\text{inc}}$ to this flow. Then, the concatenation of the systems offers a service curve $\beta = \beta_1 \otimes \beta_2 \otimes \cdots \otimes \beta_M$ to the flow.*

Let us observe this result in an example. Assume two systems where system $i \in \{1,2\}$ offers a rate-latency service $\beta_i(t) = R_i[t - T_i]^+$. Then, the concatenation of the two systems offers a service curve:

$$\beta(t) = (\beta_1 \otimes \beta_2)(t) = R[t-T]^+, \qquad R = \min(R_1, R_2), T = T_1 + T_2. \tag{3.25}$$

3.2.3 Local Analysis

As discussed in Chapter 1, obtaining bounds on backlog and delay is crucial in time-sensitive networks. The goal of this section is provide these bounds for a single system in isolation by the means of network calculus. In this context, the backlog of a flow is the amount of bits held in a system, and the delay is defined as the minimum time to serve a bit, provided that all the bits received earlier are served before it. They are defined mathematically as follows.

Definition 3.8 (Backlog)**.** *Let $\mathscr{A},\mathscr{D} \in \mathscr{F}^0_{\text{inc}}$ be such that $\mathscr{A}(t)$ [resp. $\mathscr{D}(t)$] denotes the cumulative number of bits arrived in [resp., departed from] a system until time t (excluded). The backlog $b(t)$ at any given time $t \geq 0$ is defined as*

$$b(t) \overset{\text{def}}{=} \mathscr{D}(t) - \mathscr{A}(t). \tag{3.26}$$

Alternatively, for a packetized flow, let (A, L) [resp., (D, L)] be the arrival [resp., departure] packet sequence. Then, the backlog $b(t)$ at any given time $t \geq 0$ is defined as

$$b(t) \overset{\text{def}}{=} \sum_{n\in\mathbb{N}} l_n 1_{\{D_n<t\}} - \sum_{m\in\mathbb{N}} l_m 1_{\{A_m<t\}}. \tag{3.27}$$

Definition 3.9 (Delay (Latency))**.** *Let $\mathscr{A},\mathscr{D} \in \mathscr{F}^0_{\text{inc}}$ be such that $\mathscr{A}(t)$ [resp., $\mathscr{D}(t)$] denotes the cumulative number of bits arrived in [resp., departed from] a system until time t (excluded). The delay of a bit arriving at any given time $t \geq 0$ is $d(t)$, defined as*

$$d(t) \overset{\text{def}}{=} \inf\{\tau \geq 0 : \mathscr{D}(t+\tau) \geq \mathscr{A}(t)\}. \tag{3.28}$$

Alternatively, for a packetized flow with arrival [resp., departure] packet sequence (A, L) [resp. (D, L)], the delay of a packet n is defined as

$$d_n = D_n - A_n. \tag{3.29}$$

Definition 3.10 (Delay-Jitter (Jitter))**.** *Let $\mathscr{A},\mathscr{D} \in \mathscr{F}^0_{\text{inc}}$ be such that $\mathscr{A}(t)$ [resp. $\mathscr{D}(t)$] denotes the cumulative number of bits arrived in [resp., departed from] a system until time t (excluded). Also, let $d(t)$ be the delay of a bit arriving at any given time $t \geq 0$ as defined in* (3.28)*. Then, the "delay jitter" is defined as*

$$V = \sup_{t\geq 0}\{d(t)\} - \inf_{s\geq 0}\{d(s)\}. \tag{3.30}$$

Alternatively, for a packetized flow, let (A, L) [resp., (D, L)] be the arrival [resp., departure] packet sequence. Also, let d_n be the delay of the packet n as defined in (3.29)*. The worst-case delay of the flow is $\max_n\{d_n\}$ where the max is over all packets sent by the flow during its lifetime. Similarly, the 'best-case delay of the flow is $\min_m\{d_m\}$. The delay jitter is the difference, i.e.,*

$$V = \max_n\{d_n\} - \min_m\{d_m\}, \tag{3.31}$$

so that $d_m - d_n \leq V$ for any m, n.

Delay jitter is called, in RFC 3393 [40], IP packet delay variation. In the definitions of delay and delay jitter, times are assumed to be measured according to the true time, i.e., the international atomic time (temps atomique international, TAI). In reality, times are measured with local clocks that might not be synchronized. Some small corrections could need to be applied to delay and jitter bounds [45].

The primary results on network calculus use the vertical and horizontal deviations to obtain backlog and delay bounds as stated below [25].

Theorem 3.2 (Backlog Bound). *Assume a flow, constrained by an arrival curve* $\alpha \in \mathscr{F}^0_{\text{inc}}$, *traverses a system that offers a service curve* $\beta \in \mathscr{F}^0_{\text{inc}}$ *to this flow. Then the backlog at the system is bounded by the vertical deviation,* $v(\alpha,\beta)$, *defined in Definition 3.3.*

Theorem 3.3 (Delay Bound). *Assume a flow constrained by an arrival curve* $\alpha \in \mathscr{F}^0_{\text{inc}}$ *traverses a system that offers a service curve* $\beta \in \mathscr{F}^0_{\text{inc}}$ *to this flow. Then the delay of the flow is bounded by the horizontal deviation,* $h(\alpha,\beta)$, *defined in Definition 3.3.*

As proven in [25], the above theorems provide tight bounds. The tightness means that there is a causal system with an input flow $\mathscr{A}$ and output flow $\mathscr{D}$, where the input is constrained by an arrival curve α and offered a service curve β, where the bounds are achieved. A causal system means $\mathscr{A} \le \mathscr{D}$.

The third important result is the obtainment of an arrival curve at the output of a system. The result is useful for obtaining backlog and delay bounds for downstream systems and end-to-end analysis of complex networks.

Theorem 3.4 (Arrival-Curve Propagation). *Assume a flow constrained by an arrival curve* $\alpha \in \mathscr{F}^0_{\text{inc}}$ *traverses a system that offers a service curve* $\beta \in \mathscr{F}^0_{\text{inc}}$ *to this flow. Then, at the output of the system, the flow is constrained by an arrival curve* $\alpha^* = \alpha \oslash \beta$.

Let us now examine the above results in a simple example. Assume a flow constrained by token-bucket arrival curve $\alpha(t) = rt + b, t > 0, \alpha(0) = 0$ traversing a system that offers a rate-latency service curve $\beta(t) = R[t - T]^+$ to this flow, where $R \ge r$. Then, by Theorem 3.2, the backlog at this system is

$$\begin{aligned} B = v(\alpha,\beta) &= \sup_{t\ge 0}\{\alpha(t) - \beta(t)\} = 0 \vee \left(\sup_{0<t\le T}\{rt + b - 0\}\right) \vee \left(\sup_{t>T}\{rt + b - R(t-T)\}\right) \\ &= 0 \vee (rT + b) \vee \left(\sup_{t>T}\{(r-R)t + b + RT\}\right) = (rT+b) \vee ((r-R)T + b + RT) = rT + b, \end{aligned} \tag{3.32}$$

where $\vee$ is the maximum operator. The delay bound for this flow is obtained by applying Theorem 3.3:

$$\begin{aligned} \delta = h(\alpha,\beta) &= \sup_{t\ge 0}\left\{\beta^{\downarrow}(\alpha(t)) - t\right\} = 0 \vee \sup_{t>0}\left\{\beta^{\downarrow}(\alpha(t)) - t\right\} = 0 \vee \left(\sup_{t>0}\left\{T + \frac{\alpha(t)}{R} - t\right\}\right) \\ &= 0 \vee \left(\sup_{t>0}\left\{T + \frac{rt+b}{R} - t\right\}\right) = 0 \vee \left(T + \frac{b}{R} + \sup_{t>0}\left\{\frac{rt - Rt}{R}\right\}\right) = T + \frac{b}{R}. \end{aligned} \tag{3.33}$$

And finally, the propagated arrival curve of the flow is obtained by Theorem 3.4 as

$$\begin{aligned}\alpha^*(t) &= (\alpha \oslash \beta)(t) = \sup_{s\geq 0}\{\alpha(t+s)-\beta(s)\} = 0 \vee \left(\sup_{0<s\leq T}\{\alpha(t+s)-\beta(s)\}\right) \vee \left(\sup_{s>T}\{\alpha(t+s)-\beta(s)\}\right)\\ &= 0 \vee \left(\sup_{0<s\leq T}\{r(t+s)+b-0\}\right) \vee \left(\sup_{s\geq T}\{r(t+s)+b-R(s-T)\}\right) = 0 \vee (rt+b+rT)\\ &\vee \left(rt+b+RT+\sup_{s>T}\{rs-Rs\}\right) = (rt+b+rT) \vee (rt+b+RT+rT-RT) = rt+b+rT.\end{aligned} \tag{3.34}$$

3.2.4 End-to-End Analysis

In the previous section, we have seen how to obtain delay and backlog bounds for service-curve elements. In practice, a flow typically passes by a sequence of systems from its source to its destination. The question is how to obtain end-to-end delay bound for the flow under such condition. Hereafter, we explore various techniques and methods for obtaining end-to-end delay bounds.

Total-Flow Analysis (TFA)

The TFA algorithm uses the primary network-calculus theorems to obtain end-to-end delay bound, within a feed-forward network [1][131]. The TFA algorithm, first sums up the arrival curves of the flows that share a service-curve element. Then, it computes backlog and delay bounds for each individual system by applying Theorems 3.2 and 3.3. Finally, it computes a propagated arrival-curve of every flow at the output of each system (i.e., entrance of the downstream system) by using Theorem 3.4. An end-to-end delay bound is computed as the sum of each individual delay bound.

TFA++ is an improved version of the original TFA algorithm [132]. The main idea in TFA++ is to exploit the information on the physical-line rate to improve the aggregate input arrival-curves, as well as per-flow propagated arrival-curves.

As both TFA and TFA++ can be applied only to feed-forward networks, the fixed-point TFA (FP-TFA) algorithm is proposed to overcome this limitation [133]. The FP-TFA algorithm initially performs cuts in a network with cyclic dependencies to make it feed forward. The next steps are iterative: (1) Assuming an initial vector of bursts for the flows at the cuts, it computes the local delay bounds and the propagated arrival curves using TFA++ and the improvement due to the effect of packetization [133]. (2) It compares the propagated arrival curves at the cuts with the initial vector; if it is larger, it redoes the iteration using the new vector of bursts, otherwise, the obtained delay bounds are valid and the iteration is over. Finally, an end-to-end delay bound is computed by summing up the local delay bounds.

---

[1] A network is feed forward, if there exists a numbering of its links where for any flow there is an increasing sequence of links the flow traverses [130].

Separated-Flow Analysis (SFA)

The SFA algorithm performs the end-to-end analysis by isolating the arrival and service curves for each individual flow in feed-forward networks. For any flow, SFA first computes the aggregate arrival-curve for the competing flows by summing the individual arrival-curves. Second, it obtains the residual service-curve for the flow within its path by using residual service-curves [2, Theorem 7.5]. Then, it uses the concatenation result in Theorem 3.1 to compute an end-to-end service curve for the flow. Finally, the SFA algorithm applies Theorem 3.3 using this service curve and the flow arrival-curve at the source, and computes an end-to-end delay bound for the flow.

Pay-Multiplexing-Only-Once (PMOO)

The PMOO analysis consists in a smart way of aggregating the flows that share the same path, and applying the concatenation result in Theorem 3.1 [130]. Specifically, the PMOO analysis tries to group together the flows that share a consecutive sequence of systems and to obtain an aggregate arrival-curve for each group. Then, it computes a residual service-curve [2, Theorem 7.5] for the group of the flows and applies the concatenation result in Theorem 3.1 to obtain a service curve for a sequence of systems. Finally, for the flow of interest, it computes a residual service-curve for the concatenated systems. With PMOO analysis, the burst of the competing flows is accounted once within a flow path hence improves the end-to-end delay bound.

Machine-Learning-Based Analysis

As we have seen in the previous analysis methods for computing end-to-end delay bounds, a proper concatenation of systems and an aggregation of interfering flows can lead to an improvement in the computation of delay bounds, due to the PBOO and PMOO effect. The exhaustive search to find the best flow aggregation and system concatenation is time consuming [134]. One way to tackle this issue is to use machine learning to find out an efficient composition for computing end-to-end delay bounds. DeepTMA [135] and DeepFP [136] are two such methods based on graph neural networks and reinforcement learning.

Optimization-Based Analysis

A number of analysis methods use linear programming to compute worst-case delay. The main idea in these approaches is to transform the network-calculus relations, including arrival and service curves, into linear constraints and to maximize the delay. A least upper delay bound (LUDB) [137] obtains a number of linear programming (LP) problems by using the arrival and service curves in a tandem network; solving each one of the LPs produces an upper bound on the worst-case delay. Then, the least upper bound is given by solving each individual LP problem and by taking their minimum. LUDB provides tight delay bounds for certain scenarios in tandem networks; then the mixed-integer LP (MILP) method in [138] fills in the gap by obtaining worst-case delays in FIFO tandem networks. As the method contains integer variables, it can be applied to only small networks in practice. Hence, [138] also proposes a relaxed LP version of the MILP method that leads to a delay that is close to the worst-case delay. However, these approaches are still costly due to an exponential number of variables.

The most recent approach is the polynomial-size LP (PLP) [139] that keeps the number of constraints and variables polynomial in network size. Keeping this number polynomial can lead to pessimism. However, the PLP method incorporates the delay bounds obtained by the SFA and TFA++ algorithms in order to improve the delay bounds.

3.3 Conclusion

We have presented a summary of required mathematical backgrounds, including pseudo-inverse functions, Lipschitz continuity, and min-plus algebra. We have given an overview of the main network-calculus concepts, including flow models, arrival and service curves, and the classic results to compute backlog and delay bounds. Finally, we have presented a list of methods for computing end-to-end delay bounds, among which we will use the FP-TFA method to perform our evaluations as we consider networks with cyclic dependencies.

3.4 Notation

Term	Description
b	The burst of arrival curve, when α is token-bucket.
$h(w', w)$	The horizontal deviation from function w' to function w (Definition 3.3).
r	The rate of arrival curve, when α is token-bucket.
$v(w', w)$	The vertical deviation from function w' to function w (Definition 3.3).
R	The rate of service curve, when β is rate-latency.
T	The latency of service curve, when β is rate-latency.
$\mathscr{L}(n)$	The cumulative length of the first n packets.
$P^{\mathscr{L}}(\mathscr{A})$	The $\mathscr{L}$-packetizer for a cumulative function $\mathscr{A}$.
α	The arrival curve of at the entrance of a system.
β	The service curve of a system.
$\mathscr{F}^0_{\text{inc}}$	The set of wide-sense increasing functions such that $\forall w \in \mathscr{F}^0_{\text{inc}}: \ w(0) = 0$.
$\mathbb{N}$	The set of positive integers, i.e., $\{1, 2, \dots, \}$.
w^+	The right limit of function w (Section 3.1.3).
w^-	The left limit of function w (Section 3.1.3).
$w^\downarrow$	The lower pseudo-inverse of function w (Section 3.1.1).
$w^\uparrow$	The upper pseudo-inverse of function w (Section 3.1.1).
$\lceil x \rceil$	The ceiling of x.
$\lfloor x \rfloor$	The floor of x.
$[x]^+$	$\max\{x, 0\}$.
$\wedge$	The minimum operator.
$\vee$	The maximum operator.
$\otimes$	The min-plus convolution operator (Definition 3.1).
$\oslash$	The min-plus deconvolution operator (Definition 3.1).

4 Novel Network Calculus Delay Bounds in Time-Sensitive Networks

> *All truths are easy to understand once they are discovered;*
> *the point is to discover them.*
> *— Galileo Galilei*

As mentioned in Chapter 1, one of the main goals in time-sensitive networks is to provide guarantees on worst-case delay, and not average delay. In order to obtain such guarantees, flows are assumed to be regulated at the sources. A classical form of source regulation is the bit-level arrival curve constraint. As described in Chapter 3, formally proven delay bounds can be obtained by using *classical* network calculus by taking the horizontal deviation of the arrival and service curves.

As discussed in Chapter 2, in time-sensitive networks, flow regulation at sources is often expressed in terms of number of packets rather than number of bits [80] [35, Section 35.2.2.8.4]; for example in TSN, the number of packets observed within any fixed *class measurement interval* is upper-bounded by a constant value. To obtain delay bounds for such flows, network calculus is often used [140], for which a bit-level characterization is required. Hence, a common approach is to derive a bit-level arrival-curve from the packet-level constraint and then apply network calculus [141, 142]. However, as we show in Table 4.1 and Section 4.7, this leads to delay bounds that can be improved. Indeed, our main result, in Theorem 4.1, is a novel delay bound for flows regulated with packet-level constraints, which improves on the one that is obtained by deriving a bit-level arrival-curve from the packet-level constraint. We show that the obtained delay bound is tight at least for c-Lipschitz [125] service curves where c is the physical line-speed (theorems 4.2 and 4.3).

Our method uses a novel modelling of packet-level constraint with g-regulation (Proposition 4.2). The concept of g-regulation was introduced by C.S. Chang [126] as an alternative to bit-level arrival-curve and uses max-plus algebra, whereas results for bit-level arrival-curves tend to use min-plus algebra. Our improvement in the delay bound is made possible by combining min-plus representation of service curves and max-plus representation of the input traffic to obtain a bound on the queuing delay (Lemma 4.1).

As a by-product of our method of proof, we also obtain delay bounds for flows with g-regulation which share a FIFO system with known service curve (Theorem 4.4). The most common form of g-regulation is length-rate quotient (LRQ) [122] that is among the simplest regulation constraints to control flow rate. These bounds improve on the ones obtained by deriving a bit-level arrival curve from g-regulation and then using the classic network-calculus bound. Moreover, it generalizes the results by [122] that was specifically applicable to flows with LRQ constraints and priority queues with constant rate servers. Moreover, using the same method, we provide a delay bound for flows with bit-level arrival curve that improved the bound obtained by classical network calculus presented in Chapter 3 (Theorem 4.5). We already published a subset of these results for bit-level arrival curve and rate-latency service curves [60]; in this chapter, we present the generalized result that can be applied to a wider family of service curves than rate-latency ones.

The rest of this chapter is organized as follows. Section 4.1 presents the state-of-the-art and related works. Section 4.2 includes the system model, notation, and the definition of the considered flow regulation constraints. Section 4.3 presents the relations among different regulation constraints. In Section 4.4, we present the main contribution of this chapter, namely, novel delay bounds for flows with packet-level constraints, and we show that these bounds are tight. Section 4.5 gives a generalization of the existing delay bounds for flows with g-regularity constraint and bit-level arrival-curve, as a by-product of our method of proof in Section 4.4. Section 4.6 shows that the best delay bound for a given flow is obtained by directly applying the theorem corresponding to its initial constraint. Section 4.7 provides a numerical illustration of the theorems presented in this chapter and Section 4.8 concludes the chapter. For the reader's convenience, Section 4.9 gives the notation used throughout this chapter. Appendix A gives proofs.

4.1 Related Works

As we saw in Chapter 3, the classical network-calculus proves delay bounds for a FIFO system with bit-level arrival and service curves [25, 2]. Specifically, for a flow with bit-level arrival-curve α that enters a FIFO system with service curve β, the bound is obtained by taking the horizontal deviation between the two curves, i.e., $h(\alpha, \beta)$, which is defined in Definition 3.3. For example, when a flow has leaky bucket arrival curve $\alpha(t) = rt + b$ and the FIFO system has rate-latency service-curve $\beta(t) = \max(R(t - T), 0)$, we have $h(\alpha, \beta) = T + \frac{b}{R}$ provided that $r \leq R$. This bound is tight if only arrival and service curves are known. However, we often have more information, specially in the context of time-sensitive networks, e.g., TSN schedulers with credit-based shapers [63] and DRR schedulers [1]: when a packet starts its transmission, it is transmitted with full line rate. In our initial result, such information was exploited for rate-latency service-curves to provide a delay bound that improves the classical network-calculus bound [62]; more precisely, the classical network-calculus bound is reduced by $L^{\min}\left(\frac{1}{R} - \frac{1}{c}\right)$, where c is the physical line-rate and $L^{\min}$ is the minimum packet length of the input traffic.

While rate-latency service-curves are commonly used in the literature, they may not provide a perfect characterization of a system. A number of works provide more complex service curves that in turn leads to better delay bounds. In [1], the authors obtained a non rate-latency service-curve for DRR schedulers that can incorporate the arrival curves of the interfering flows. Such non rate-latency service-curves are obtained as well for weighted round-robin [2, Section 8.2.4] and interleaved weighted round-robin [98]. Hence, the improvement of [62] does not apply to these service curves. In this chapter, we present a generalized version of the results of [62] to improve delay bounds for non rate-latency service-curves. Furthermore, we improve the bounds in [62] when the flows have packet-level constraints.

A number of other works focus on improving the arrival curves of the flows in time-sensitive networks by taking advantage of input-line shaping effect. For example, for a flow with leaky-bucket arrival curve $\alpha(t) = rt + b$ that passes a physical line with rate c, we can obtain a better arrival curve $\alpha'(t) = \min(rt+b, ct)$. In [141, 142], the authors study a TSN network and assume the input traffic has packet-level constraint [35, Section 35.2.2.8.4]. To obtain delay bounds, they first derive a bit-level arrival-curve from the packet-level constraint; then they exploit input-line shaping to improve the obtained arrival curve. Finally, they use network calculus to obtain delay bounds. The improvement by the input-line shaping effect is complementary to the improvements shown in this chapter.

Recently, LRQ was introduced in the context of interleaved regulators in [122] as a per-flow regulation constraint that is simpler to implement than token-bucket. LRQ is in fact a specific case of a shifted-rate regulator (when the delay term is set to zero) [126, Section 6.2.1], which belongs to the family of g-regulation. [122] obtains delay bounds for flows with LRQ constraint within constant-rate servers and strict-priority queuing. This analysis was extended for guaranteed-rate servers [143]. Unlike the previous works, our results on g-regulated flows cover the whole family of g-regulation (i.e., are not limited to LRQ constraint) and apply to a broader class of network nodes (i.e., nodes with arbitrary service curves).

4.2 System Model

We study a FIFO system with a queue and a transmission link, as in Fig. 4.1. Each queue is shared among a number of flows. Upon arrival, packets of different flows enter the queue and are stored in FIFO order. A scheduler decides when the packet at the head of the queue is selected for transmission. The scheduler typically arbitrates between this queue and other queues (not shown), therefore the packet at the head of the queue may have to wait even if there is no packet of this queue in transmission. When the packet at the head of this queue is selected for transmission, it is transmitted at a constant rate c until it is completely transmitted, i.e., there is no preemption.

Every flow is assumed to be constrained by either bit-level arrival curve (Section 3.2.2), g-

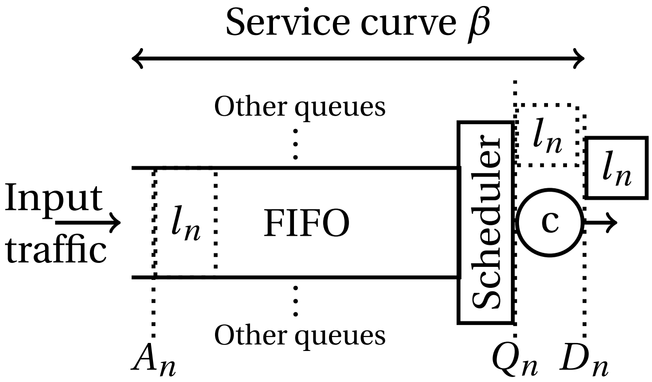


Figure 4.1: The considered FIFO system.

regulation (Section 4.2.2) or packet-level arrival curve (Section 4.2.3). For each flow f, let L_f^{min} and L_f^{max} denote the minimum and maximum packet lengths, in bits. We also assume that the total flow of all incoming packets is packetized, i.e., we consider that all bits of any packet arrive at the same time instant.

Let A_n be the arrival time of packet n to the FIFO system, where the numbering of packets is by order of arrival and Q_n be the time at which packet n is selected for transmission. The FIFO assumption means that $Q_n \le Q_{n+1}$. We call $Q_n - A_n$ the queuing delay of packet n in the FIFO system. Let l_n be the length in bits of packet n and assume that it belongs to flow f, so that $L_f^{\text{min}} \le l_n \le L_f^{\text{max}}$; then packet n leaves the system at time $D_n = Q_n + \frac{l_n}{c}$. We call $D_n - A_n$ the response time of the FIFO system for packet n.

We assume that the scheduler is such that the FIFO system offers to the total flow of all incoming packets a service curve β, defined in Section 3.2.2.

4.2.1 Bit-level Arrival Curve

We follow the definition of bit-level arrival curve as in Section 3.2.2. Moreover, we know by [25, Lemma 1.2.1] that if α is an arrival-curve for a flow, then so is its left limit, i.e., α^-. Also, since input is packetized, $\alpha^+(0)$ is an upper bound on the size of any packet. Therefore, in this chapter, we assume that α is left-continuous and $\alpha^+(0) \ge L^{\text{max}}$.

4.2.2 g-regulation

The g-regulation constraint was introduced in [126]; it specifies that the time inter-spacing between packets is lower bounded by a left-continuous function $g \in \mathscr{F}_{\text{inc}}^0$ [1] [126]. A packetized

---

[1] The original g-regularity in [126] uses functions defined on $\mathbb{N} \cup \{0\}$ but it is simpler to consider functions defined on $\mathbb{R}^+$.

input has g-regulation constraint if and only if for any packet indices m, n, where $m \le n$:

$$A_n - A_m \ge g\left(\sum_{k=m}^{n-1} l_k\right). \tag{4.1}$$

Note the difference between the bit-level arrival-curve in (3.21) and the g-regulation in (4.1): for the g-regulation the length of the last packet, i.e., l_n, does not play a role in (4.1) while it is included in (3.21). Since the argument of the function g only takes values in a discrete set containing sums of packet sizes, we assume g is left continuous at each point in this set.

Shifted-rate regulation [126, Section 6.2.1] is a type of g-regulation with $g(x) = \frac{1}{r}[x-d]^+$, where r is the regulation rate and d is the regulation delay, i.e., the constraint is

$$A_n - A_m \ge \frac{1}{r}\left[\left(\sum_{k=m}^{n-1} l_k\right) - d\right]^+, \forall m, n \in \mathbb{N}, m \le n. \tag{4.2}$$

Length-rate quotient, introduced by [122], is a traffic regulation which specifies the minimum interspacing between two consecutive packets as a function of a regulation rate r and the length of the earlier packet. i.e. the constraint is

$$A_n - A_{n-1} \ge \frac{l_{n-1}}{r}, \quad \forall n \in \mathbb{N}, n \ge 2. \tag{4.3}$$

By LRQ, the arrival of packet n is constrained by the regulation rate, the arrival time and the length of the previous packet; the fact that there is dependency on only the last packet, renders the implementation of LRQ very simple. It is easy to see that, when $d = 0$, (4.2) is equivalent to (4.3), namely, LRQ is shifted-rate regulation with $d = 0$; therefore, LRQ is a form of g-regulation with $g(x) = \frac{x}{r}$.

4.2.3 Packet-level Arrival Curve

The packet-level arrival-curve is mainly used in the context of IEEE TSN and IETF DetNet and expresses traffic constraints at the packet level. More formally, consider a left-continuous wide-sense increasing function $N : \mathbb{R}^+ \to \mathbb{N} \cup \{0\}$, where $N(t)$ is the cumulative number of packets observed on a flow of interest until time t (excluded). Then, we say that the flow has a packet-level arrival-curve $\alpha_{\text{pkt}} : \mathbb{R}^+ \to \mathbb{N}$, $\alpha_{\text{pkt}}(0) = 0$, if and only if

$$N(t) - N(s) \le \alpha_{\text{pkt}}(t-s), \forall t \ge 0 \text{ and } s \in [0, t]. \tag{4.4}$$

Similarly to the bit-level arrival-curve constraint, (4.4) is equivalent to

$$n - m + 1 \le \alpha^+_{\text{pkt}}(A_n - A_m) \forall m, n \in \mathbb{N}, m \le n. \tag{4.5}$$

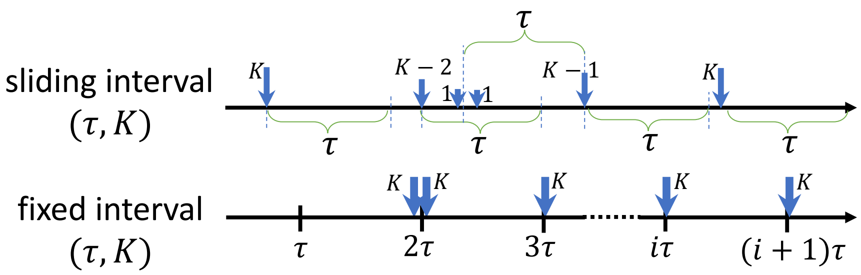


Figure 4.2: The two interpretations of TSN and DetNet interval.

It is known that an arrival curve can always be assumed to be sub-additive as defined in (3.6), since otherwise it can be replaced by its sub-additive closure (Definition 3.2) [25]. Also, following the same steps as the proof of [25, Lemma 1.2.1], we can prove that if α_{pkt} is a sub-additive packet-level arrival-curve for a flow, then so is its left limit α^{-}_{pkt}. Also, since the input is packetized, $\alpha^{+}_{\text{pkt}}(0)$ gives an upper bound on one packet. Therefore, in the rest of this chapter, we assume that α_{pkt} is sub-additive, left continuous and $\alpha^{+}_{\text{pkt}}(0) \geq 1$.

As discussed in Chapter 2, in IEEE TSN the traffic specifications for a flow is defined as [35, Section 34.6.1]:

> "... during each class measurement interval, it [a source] can place up to MaxIntervalFrames data frames, each no longer than MaxFrameSize into that stream's queue."

Similarly, the traffic specification in IETF DetNet allows packet level constraint [80, Section 5.5] by the attributes "Interval" as the period of time in which the traffic specification is specified and "MaxPacketsPerInterval" as the maximum number of packets that a source transmits in one Interval.

By the above specifications, the interval (i.e. class measurement interval in TSN or Interval in IETF DetNet) can be interpreted differently as seen in Fig. 4.2:

1. Sliding interval (τ, K): the number of packets is limited by K =MaxIntervalFrames (or MaxPacketsPerInterval) at any sliding interval of duration τ.
2. Fixed interval (τ, K): the number of packets is limited by K =MaxIntervalFrames (or MaxPacketsPerInterval) at any reference interval of duration τ; the reference intervals are consecutive and non-overlapping.

The first interpretation, sliding interval (τ, K), is equivalent to

$$N(t+\tau) - N(t) \leq K; \quad \forall t \geq 0, \tag{4.6}$$

which is also equivalent to saying that the flow has a packet-level arrival-curve given by

$$\alpha_{\text{pkt}}(t) = K\lceil \frac{t}{\tau} \rceil. \tag{4.7}$$

This is a staircase packet-level arrival-curve with burst equal to K packets and period τ. It is easy to verify that it is sub-additive and left-continuous. Applying the right limit of (4.7) into (4.5), (4.6) is equivalent to the following for any packet indices m, n, $m \le n$:

$$n - m + 1 \le K\lfloor \frac{A_n - A_m}{\tau} \rfloor + K. \tag{4.8}$$

The second interpretation, fixed interval (τ, K), is equivalent to saying that there exists some offset θ such that:

$$\begin{aligned} N(\theta) &= N(0) = 0, \\ N(\theta + (i+1)\tau) - N(\theta + i\tau) &\le K; \quad \forall i \in \mathbb{N}. \end{aligned} \tag{4.9}$$

As shown in Lemma A.3 in Appendix A, the second interpretation, fixed interval (τ, K), implies a packet-level arrival-curve with,

$$\alpha_{\text{pkt}}(0) = 0, \;\; \alpha_{\text{pkt}}(t) = K\lceil \frac{t}{\tau} \rceil + K : t > 0, \tag{4.10}$$

It is easy to verify that this function is sub-additive and left-continuous. However, the converse does not hold, i.e., it is not true that all flows that have the packet-level arrival curve in (4.10) satisfy fixed interval (τ, K) (because such an arrival curve allows $2K$ packets in an interval of duration less than τ). Nonetheless, we show in Section 5.2 that the delay bound obtained using the packet-level arrival-curve in (4.10), is tight for flows with the fixed interval (τ, K) regulation constraint.

Another form of packet-level regulation found in the literature is the token-bucket packet-level constraint [123], which limits the number of packets within any time interval t to $\rho t + B$, where $\rho > 0$ and $B \ge 1$ are respectively the packet rate and burst. This constraint enjoys the superposition property, i.e., for an aggregation of flows each with such constraint, the superposition is constrained by a token-bucket packet-level constraint with ρ and B equal to the sum of the packet rates and bursts of the flows respectively. Such a constraint is equivalent to the following packet-level arrival-curve:

$$\alpha_{\text{pkt}}(0) = 0, \;\; \alpha_{\text{pkt}}(t) = \lceil \rho t + B - 1 \rceil : t > 0. \tag{4.11}$$

It can be easily verified that this function is sub-additive and left-continuous. Now, by (4.5), for any packet indices m, n, where $m \le n$, (4.11) is equivalent to:

$$n - m + 1 \le \lfloor \rho(A_n - A_m) + B \rfloor. \tag{4.12}$$

The (λ, ν)-constraint introduced in [144], is equal to the token-bucket packet-level arrival-curve with $\rho = \lambda$ and $B = \nu + 1$. The staircase packet-level arrival-curve in (4.5) implies a token-bucket packet-level arrival-curve with $\rho = \frac{\tau}{K}$, $B = K$.

4.3 Relations among Flow Regulations

We have seen various families of traffic regulations in Section 4.2. We can find regulation-specific toolbox for delay analysis, e.g., [25, 2] for bit-level arrival curves. The immediate question is whether there is a relation among these regulation types. Such relations can open up new opportunities to apply the existing toolbox of one regulation type to another, that may lead to delay improvements, as we will see in Section 4.4. The goal of this section is to present such derivations between the regulation types.

In the next proposition, we present a relation between g-regularity and bit-level arrival-curve. The derivation of bit-level arrival-curve from g-regularity already exists in literature and we put it here for completeness [126].

Proposition 4.1. *Consider a flow f.*

1. *If f has a bit-level arrival-curve $\alpha(t)$, it also conforms to a g-regularity constraint with $g(x) = \alpha^{\downarrow}(x + L^{\min})$, $x > 0; g(0) = 0$.*
2. *If f conforms to a g-regularity constraint, it also has a bit-level arrival-curve $\alpha(t) = g^{\downarrow}(t) + L^{\max}$.*
3. *The sequential application of items 1 and 2 or items 2 and 1 gives a regulation constraint that is weaker than the initial one, except if all packets have the same size.*

The proof is in Appendix A.2.1. Proposition 4.1 shows that even though bit-level arrival-curve and g-regularity are two different families of traffic constraints, they can be derived from each other in items (1) and (2). However, item (3) shows that when packets are of different sizes, such derivations lead to weaker constraints than the initial traffic constraints.

Proposition 4.2. *Consider a flow with packet-level arrival-curve α_{pkt}, then:*

1. *The flow conforms to a g-regularity constraint with:*

$$g(x) = \alpha^{\downarrow}_{\text{pkt}}\left(\frac{x}{L^{\max}} + 1\right), \tag{4.13}$$

and to a bit-level arrival-curve constraint, α, with:

$$\alpha(t) = L^{\max}\alpha_{\text{pkt}}(t). \tag{4.14}$$

2. *The g-regularity constraint in* (4.13) *is stronger than the bit-level arrival-curve constraint in* (4.14)*; i.e., the sequential application of* (4.13) *and then item 1 of Proposition 4.1 gives*

(4.14) *while the sequential application of* (4.14) *and then item 2 of Proposition 4.1 gives a looser constraint than* (4.13)*, except if all packets have the same size.*

The proof is in Appendix A.2.2. By item (1), we can use the bit-level arrival-curve and g-regularity to compute delay bounds for flows with packet-level arrival-curves. In fact, for TSN/DetNet traffic regulation, such bit-level arrival-curve derivation already exists [141] and is used for delay-bound computation. While in literature only the bit-level arrival-curve derivation is used to obtain delay bounds, item (2) shows that the g-regularity derivation is a stronger constraint, and it leads to delay improvements as we will see in Theorem 4.1.

4.4 Delay Bounds for Packet-level Arrival Curves

The focus of this section is to find delay bounds for flows with packet-level arrival-curve that share a FIFO system described in Section 7.2. To do so, we use a novel combination of g-regularity and bit-level arrival-curves derived from packet-level arrival-curves. We start with a technical lemma about queuing delay.

Lemma 4.1. *Consider a FIFO system with service curve $\beta \in \mathscr{F}^0_{\text{inc}}$. Assume that we know some $w \in \mathscr{F}^0_{\text{inc}}$ such that for any two packet indices m, n, $m \le n$, we have*

$$\sum_{k=m}^{n-1} l_k \le w(A_n - A_m), \tag{4.15}$$

where A_m, A_n are packet arrival times and l_k is the length of packet k. Then, the queuing delay of the FIFO system is bounded by $h(w, \beta)$, i.e. for any packet index n:

$$Q_n - A_n \le h(w, \beta)$$

where $h(.)$ is the horizontal deviation.

The proof is in Appendix A.2.3; it combines min-plus representation of the service curve and the max-plus representation of the input traffic in (4.15). Using Lemma 4.1, we obtain our first result on delay bounds, on which all delay bounds found in this chapter are based.

Lemma 4.2. *Consider a FIFO system offering a service curve β to the aggregate of the flows sharing it. Assume that (i) flow 1 is the flow of interest and conforms to a g-regularity constraint, and (ii) flow 2 represents the aggregate of the remaining flows sharing the FIFO system and has bit-level arrival-curve α. In addition, assume that as soon as a packet starts to be transmitted, it is transmitted with rate c. Then,*

(i) An upper bound on the response time of a packet with length l of flow 1 at this FIFO system is:

$$\Delta^{\text{AG}}(l) = h(g^{\uparrow} + \alpha^{+}, \beta) + \frac{l}{c}, \tag{4.16}$$

(ii) An upper bound on the response time of any packet of flow 1 *at this FIFO system is:*

$$\Delta^{\mathrm{AG}} = h(g^{\uparrow} + \alpha^{+}, \beta) + \frac{L_1^{\max}}{c}. \tag{4.17}$$

The proof is in Appendix A.2.4. We can now apply Lemma 4.2 to packet-level arrival curves and obtain the following theorem:

Theorem 4.1. *Consider a FIFO system offering a service curve β to the aggregate of the flows sharing it. Assume any flow $f \in \{1, \dots, M\}$ has $\alpha_{\mathrm{pkt},f}$ as packet-level arrival-curve at the entrance of the FIFO system and has maximum packet length $L_f^{\max}$. In addition, assume that as soon as a packet starts to be transmitted, it is transmitted with rate c. Then,*

(i) An upper bound on the response time of a packet with length l of a flow f at this FIFO system is:

$$\Delta^{pkt}(l) = h\left(\sum_{i=1}^{M} L_i^{\max}\alpha_{\mathrm{pkt},i}^{+} - L_f^{\max}, \beta\right) + \frac{l}{c}. \tag{4.18}$$

(ii) An upper bound on the response time of any packet of flow f at this FIFO system is:

$$\Delta^{pkt} = h\left(\sum_{i=1}^{M} L_i^{\max}\alpha_{\mathrm{pkt},i}^{+} - L_f^{\max}, \beta\right) + \frac{L_f^{\max}}{c}. \tag{4.19}$$

Proof. Use item 1 of Proposition 4.2 to derive a g-regularity constraint for the flow of interest and a bit-level arrival-curve for the competing flows. Then apply Lemma 4.2. □

The delay bound in Theorem 4.1 improves the state-of-the-art delay bounds at least when packet sizes are different. Let us observe the improvement for a simple case of rate-latency service-curves in the following example; the bounds are summarized in Table 4.1.

Example. Consider a FIFO system with rate-latency service-curve $\beta(t) = R[t-T]^{+}$ connected to a link with transmission rate $c > R$. Assume flows 1 and 2 have packet-level arrival-curves $\alpha_{\mathrm{pkt}}(t) = K\lfloor\frac{t}{\tau}\rfloor$ and $\alpha'_{\mathrm{pkt}}(t) = K'\lfloor\frac{t}{\tau'}\rfloor$ respectively ($K, K' \in \mathbb{N}$, $\tau, \tau' > 0$). To avoid unbounded response time, we assume $\frac{KL_1^{\max}}{\tau} + \frac{K'L_2^{\max}}{\tau'} \le R$. We want to compute a delay bound for flow 1.

The classical approach is to first derive the bit-level arrival curves corresponding to the two flows. By Proposition 4.2, they are:

$$\begin{aligned} \alpha(t) &= L_1^{\max}\alpha_{\mathrm{pkt}}(t) = KL_1^{\max}\lfloor\frac{t}{\tau}\rfloor, \\ \alpha'(t) &= L_2^{\max}\alpha'_{\mathrm{pkt}}(t) = K'L_2^{\max}\lfloor\frac{t}{\tau'}\rfloor. \end{aligned} \tag{4.20}$$

Table 4.1: Comparison of the delay upper-bound for packet-level arrival-curve in the example of Section 4.4.

Approach	Delay upper-bound
Classical approach: bit-level arrival-curve + classical network-calculus bound	$\Delta^{\text{NC}} = T + \frac{KL_1^{\max}+K'L_2^{\max}}{R}$
Improved approach: bit-level arrival-curve + improved bound in Theorem 4.5	$\Delta^{\text{A}} = \Delta^{\text{NC}} - L_1^{\min}\left(\frac{1}{R}-\frac{1}{c}\right)$
Improved approach: packet-level arrival-curve + new bound in Theorem 4.1	$\Delta^{\text{pkt}} = \Delta^{\text{NC}} - L_1^{\max}\left(\frac{1}{R}-\frac{1}{c}\right)$

Then, a delay bound is computed by the classical network-calculus bound. For rate-latency service-curves, the classical network-calculus bound is improved by [62], which is a special case of our result in Theorem 4.5 (we will see in the next section). Our approach, however, is to consider the original packet-level arrival-curves and directly apply Theorem 4.1. Table 4.1 shows the obtained bounds for the above approaches.

Comparing the bound of Theorem 4.1 with the improved version of the classical approach, we have:

$$\Delta^{\text{pkt}} - \Delta^{\text{A}} = -(L_1^{\max} - L_1^{\min})\left(\frac{1}{R}-\frac{1}{c}\right) < 0, \tag{4.21}$$

which implies that the bound obtained by Theorem 4.1 is strictly less than the improved classical approach, except when all packets of flow 1 are of the same size ($L_1^{\min} = L_1^{\max}$).

The novel delay bound in Theorem 4.1 is derived from Lemma 4.2 and thus uses g-regularity constraint and bit-level arrival-curves derived from the packet-level arrival-curves. Therefore, it is legitimate to wonder whether it is the best possible bound derived from packet-level arrival curves. We show in the following theorem that this is indeed the case.

Theorem 4.2. *The bound of Theorem 4.1 is tight.*

More specifically, consider a c-Lipschitz service curve β, sub-additive left-continuous functions $\alpha_{\text{pkt},f}$ as packet-level arrival-curves for flow $f \in \{1,\dots,M\}$, with maximum packet lengths $L_f^{\max}$, and a transmission rate c. There exists a simulation trace of a FIFO system, shared between the M flows, where a packet of flow 1 reaches the bound in (4.19) of Theorem 4.1.

The proof is in Appendix A.2.5; it consists in building a trajectory with *greedy* sources, i.e. with sources for which the cumulative packet arrival function $N(t)$ satisfies $N(t) = \alpha_{\text{pkt}}(t - t_0)$ for some offset t_0.

Recall that there are two interpretations of the packet-level regulation constraint of TSN/Det-

Net (Section 4.2.3). The former, sliding interval, is equivalent to a packet-level arrival curve constraint; the latter, fixed interval, implies a packet-level arrival curve constraint but is not equivalent to it. However, as we show in the next theorem, the delay bound obtained by using packet-level arrival-curves and Theorem 4.1 is the best possible bound for flows with TSN/DetNet constraints.

Theorem 4.3. *Consider a FIFO system offering a c-Lipschitz service curve β to an aggregate of M flows, where, as soon as a packet starts to be transmitted, it is transmitted with rate c. Assume that every flow $f \in \{1,\dots,M\}$ conforms to either the Sliding interval (τ_f, K_f) or the Fixed interval (τ_f, K_f) constraint of TSN/DetNet (Section 4.2.3) and has maximum packet size $L_f^{\max}$. For every flow f, we can derive a packet-level arrival curve, by using* (4.7) *or* (4.10)*, and apply Theorem 4.1 to obtain a delay bound Δ_f^{pkt}.*

Then, for every flow f and every $\varepsilon > 0$, there exists a simulation trace of this system where a packet of flow f experiences a delay in the interval $[\Delta_f^{pkt} - \varepsilon, \Delta_f^{pkt}]$.

The proof is in Appendix A.2.6. The main issue here is that a flow that conforms to Fixed interval (τ, K) has packet-level arrival curve given by (4.10), but a greedy source for this arrival curve does not satisfy Fixed interval (τ, K) (as it generates $2K$ packets in one interval of duration τ). We overcome this by considering, for every $\varepsilon > 0$, a greedy source for the packet-level arrival curve $K\lceil \frac{[t-\varepsilon]^+}{\tau} \rceil + K$, which does conform to Fixed interval (τ, K).

4.5 Improvements on Existing Network-Calculus Delay-Bounds

In this section, as a by-product of Lemmas 4.1 and 4.2, we derive novel delay bounds for flows with g-regularity constraints (Theorem 4.4) and bit-level arrival-curves (Theorem 4.5). The obtained delay bounds for flows with g-regularity constraint generalize the results of [122], which are specific to LRQ and strict-priority schedulers. Similarly, for flows with bit-level arrival-curve, our delay bounds improve on the ones obtained by classical network-calculus and generalize the results of [62], which are specific to rate-latency service-curves.

Theorem 4.4. *Consider a FIFO system offering a service curve β to the aggregate of n flows sharing it. Assume any flow $f \in \{1,\dots,M\}$ has g_f-regulation constraint at the entrance of the FIFO system and has maximum packet length $L_f^{\max}$. In addition, assume that as soon as a packet starts to be transmitted, it is transmitted with rate c. Then,*

(i) An upper bound on the response time of a packet with length l of a flow f at this FIFO system is given as follows:

$$\Delta^G(l) = h\left(\sum_{i=1}^{M} g_i^{\uparrow} + \sum_{i=1}^{M} L_i^{\max} 1_{i \neq f}, \beta\right) + \frac{l}{c}, \tag{4.22}$$

(ii) An upper bound on the response time of any packet of flow f at this FIFO system is given as

follows:

$$\Delta^G = h\left(\sum_{i=1}^{M} g_i^{\uparrow} + \sum_{i=1}^{M} L_i^{\max} 1_{i\neq f}, \beta\right) + \frac{L_f^{\max}}{c}. \tag{4.23}$$

The proof consists in two steps; we first take the sum of the bit-level arrival-curves of the competing flows derived using item 1 of Proposition 4.1; second, we apply Lemma 4.2.

For the most common form of g-regularity constraint, i.e., LRQ, the current available delay bound is by [122] which only applies to strict-priority schedulers; under the same assumption, Theorem 4.4 gives the same delay bounds as [122]. The delay bounds presented in [122] do not apply to a vast majority of schedulers, e.g., WRR, IWRR, and DRR. Theorem 4.4, however, can be used to obtain delay bounds for such scheduling mechanisms. We provide a numerical illustration of Theorem 4.4 for a DRR scheduler in Section 4.7.2.

Next, we present the delay bounds for flows with bit-level arrival-curves. It follows from the definition that the bit-level arrival-curves can be superimposed: an aggregate of flows conforms to a bit-level arrival-curve equal to the sum of the bit-level arrival-curves of the flows. Hence, in the next theorem, we assume an aggregate of flows (that share a FIFO system with the flow of interest) is constrained by one bit-level arrival-curve constraint.

Theorem 4.5. *Consider a FIFO system offering a service curve β to the aggregate of the flows sharing it and such that, as soon as a packet starts to be transmitted, it is transmitted with rate c. Flow 1 is the flow of interest and flow 2 represents the aggregate of the remaining flows sharing the FIFO system. Assume that that flows 1, 2 have bit-level arrival-curves α, α' respectively; then,*

(i) an upper bound on the response time of a packet with length l of flow 1 at this FIFO system is

$$\Delta^A(l) = h(\alpha^+ + \alpha'^+ - l, \beta) + \frac{l}{c}; \tag{4.24}$$

(ii) if β is c-Lipschitz, an upper bound on the response time of any packet of flow 1 at this FIFO system is

$$\Delta^A = h(\alpha + \alpha' - L_1^{\min}, \beta) + \frac{L_1^{\min}}{c}. \tag{4.25}$$

The proof is in Appendix A.2.7; it consists in two steps. We first derive the queuing delay bound for the FIFO system using Lemma 4.1 and then we add the transmission time for the packet of interests. To obtain the per-flow delay-bound in item (ii), we take the supremum of per-packet delay-bound in item (i) for all the range of packet lengths, i.e., $[L_1^{\min}, L_1^{\max}]$. When the service-curve is c-Lipschitz (Definition 3.4), the supremum is achieved at $l = L_1^{\min}$ (in Appendix A.3, we show that this does not always hold for a non c-Lipschitz service-curve). The c-Lipschitz condition in Theorem 4.5 implies that the slope of the service curve is not more than the transmission rate c at any point in time. In fact, the rate-latency service-curves as well as the existing service-curves for the common scheduling mechanisms in time-sensitive networks, e.g., TSN schedulers with credit-based shapers [63, 145], DRR [1, 127], IWRR[98], are c-Lipschitz; therefore, the bound in Theorem 4.5 can be applied to such systems.

Theorem 4.5 gives the same delay bound as [62] (our initial result on delay improvement of service-curve elements) for rate-latency service-curves. For a number of scheduling mechanisms, e.g., DRR and IWRR, the rate-latency service-curves are not accurate; to this end, better non rate-latency service-curves are obtained that provide more precise characterization for these mechanisms [1, 98]. For such non rate-latency service-curves, the delay-bound improvement of [62] cannot be used; instead, the existing approach is to use classical network calculus to obtain a delay bound, which is $h(\alpha+\alpha',\beta)$. This bound is improved by Theorem 4.5. We provide a numerical illustration of Theorem 4.5 for a non rate-latency service-curve in Section 4.7.3.

4.6 Comparison of the Different Bounds

In sections 4.4 and 4.5 we obtained delay bounds for flows with various regulation types. Moreover, we showed in Section 4.3 that regulation types can be derived from each other. Combining the mentioned results, we can derive a regulation type from another and then apply the corresponding theorem, e.g., deriving bit-level arrival-curve from g-regularity using item (1) of Proposition 4.1 and then apply Theorem 4.5. Now, the question is how such delay bounds are compared with the ones obtained by directly applying the theorem to initial flow constraint. The goal of this section is address this question.

We start with packet-level arrival-curve. Our tight delay bound for packet-level arrival-curves in Theorem 4.1 uses the g-regularity derivation of Proposition 4.2 for the flow of interests. The other approach is to use bit-level derivation of packet-level arrival-curves in Proposition 4.2; this is indeed the state-of-the-art approach. We already showed in Table 4.1 that this leads to sub-optimal delay bounds for rate-latency service-curves. The following proposition shows that this is not accidental and is true in general.

Proposition 4.3. *Consider the assumptions in Theorem 4.1. Using Proposition 4.2 we can derive a bit-level arrival-curve and, by applying Theorem 4.5, obtain a delay bound* Δ_f^{A} *for every flow* f*. Let* Δ_f^{pkt} *be the delay bound obtained by a direct application of Theorem 4.1. Then* $\Delta_f^{\mathrm{pkt}} \le \Delta_f^{\mathrm{A}}$*. Furthermore, if packets of flow* f *have a constant size (*$L_f^{\min} = L_f^{\max}$*) then* $\Delta_f^{\mathrm{pkt}} = \Delta_f^{\mathrm{A}}$*, else* $\Delta_f^{\mathrm{pkt}} < \Delta_f^{\mathrm{A}}$*, in general.*

The proof is in Appendix A.2.8. In the above, "in general" means that we can find schedulers for which the inequality is strict, for example when the service curve is rate-latency with rate $R < c$.

Similarly, we evaluate the delay bounds achieved by deriving g-regularity from bit-level arrival-curve [resp. bit-level arrival-curve from g-regularity] and then applying Theorem 4.4 [resp. Theorem 4.5]. Then, the question is whether we obtain the same delay bounds by applying theorems 4.4 and 4.5 respectively to the derived g-regularity constraint and bit-level arrival-curve. The following proposition shows that the answer to this question is negative and the obtained delay bound is generally worse, except for flows with packets of constant size.

Proposition 4.4. *Consider the FIFO system assume in Theorem 4.5.*

1. *Assume that the flows have bit-level arrival-curves. Then, Theorem 4.5 gives a better response time upper-bound than the sequential application of item 1 of Proposition 4.1 and Theorem 4.4. Specifically:* $\Delta_f^A \leq \Delta_f^G$*; furthermore, if packets of flow* f *have a constant size (*$L_f^{\min} = L_f^{\max}$*) then* $\Delta_f^A = \Delta_f^G$ *and else* $\Delta_f^A < \Delta_f^G$ *in general.*
2. *Assume that the flows have* g*-regularity constraints. Then, Theorem 4.4 gives a better response time upper-bound than the sequential application of item 2 of Proposition 4.1 and Theorem 4.5. Specifically:* $\Delta_f^A \leq \Delta_f^G$*; furthermore, if packets of flow* f *have a constant size (*$L_f^{\min} = L_f^{\max}$*) then* $\Delta_f^A = \Delta_f^G$ *and else* $\Delta_f^G < \Delta_f^A$ *in general.*

The proof is in Appendix A.2.9. Propositions 4.3 and 4.4 indicate that the best delay bound for a given flow is obtained by directly applying the theorem corresponding to its initial constraint, as delay bounds obtained from derived flow constraints are pessimistic.

4.7 Numerical Illustration

This section provides three example applications of the theorems presented in this chapter to commonly used schedulers. To compute the delay bounds, we use the RealTime-at-Work (RTaW) tool[146] that has efficient implementation of network calculus operations.

4.7.1 Flows with Packet-level Arrival Curve

Consider a FIFO system with TSN scheduler and CBSs with per-class FIFO queuing [63]; from highest to lowest priority, the classes are CDT, A, B, and Best Effort (BE). The CBSs are used separately for classes A and B. The CBS parameters *idleslope*s are set to 50% and 25% of the link rate, $c = 1$ Gbps, respectively for classes A and B. The CDT traffic has a token-bucket arrival curve with rate 6.4 Kbps and burst 64 Bytes. The maximum packet length of class BE is 1.5 KB. Using the results in [63], a rate-latency service curve offered to class A has latency $T_A = 12.5\mu$s and rate $R_A = 499.92$ Mbps, and the one offered to class B has $T_B = 36.6\mu$s and rate $R_B = 249.75$ Mbps. There are 5 periodic flows for each of classes A and B; each flow send 1 packet at each period. Table 4.2 shows the flow information. We want to compute delay bounds for flows 1 and 6 in classes A and B.

The state-of-the-art approach is to obtain bit-level arrival-curve for all flows using Proposition 4.2; then by Theorem 4.5 we obtain a delay bound of 63.58 μs for flow 1 (class A) and 158.47 μs for flow 6 (class B). However, we can also directly apply Theorem 4.1 to obtain delay bounds;

Table 4.2: Flow information for Section 4.7.1.

Class A			Class B		
id	$L^{\max}$ (Bytes)	period (ms)	id	$L^{\max}$ (Bytes)	period (ms)
1	1442	16	6	1438	64
2	185	4	7	619	64
3	537	16	8	773	128
4	414	4	9	459	128
5	350	8	10	592	128

for flows 1 and 6 we obtain 50.46 μs and 126.32 μs respectively, which shows 20% improvement in both cases.

4.7.2 Flows with g-regulation

Consider a FIFO system with aggregate queuing and DRR arbitration policy, with $n = 8$ queues sharing a link with rate $c = 1$ Gbps. Assume all flows have maximum packet length of $L^{\max} = 1.5$ KB and the queues have same quantum value $Q = L^{\max}$. Then, by [1, Theorem 1], we obtain a service curve offered to any queue,

$$\beta(t) = (\theta \circ \zeta)(t) + \min\Big(\big[ct - 2(n-1)(2L^{\max} - \epsilon)\big]^+, \epsilon\Big), \tag{4.26}$$

with,

$$\theta(t) = \inf_{0 \le s \le t} \left\{ t - s + Q \lfloor \frac{s}{(n-1)Q} \rfloor \right\}$$
$$\zeta(t) = [ct - (n-1)(4Q - \epsilon) + \epsilon]^+,$$

and we set $\epsilon = 1$ Byte.

Now, assume a flow of interest, conforms to LRQ with rate r, shares a queue with a number of other flows with LRQ regulation where the sum of their maximum packet lengths is 5 KB. Also assume that the minimum packet length of the flow is 100 Bytes.

We obtain a bit-level arrival-curve constraint for the flows using Proposition 4.1. Then by Theorem 4.5 we obtain a delay bound of 632.75 μs for the flow of interest. Using the results of this chapter, we can also directly apply Theorem 4.4 to obtain delay bounds that gives 552.74 μs, which shows 12.6% improvement.

4.7.3 Flows with Bit-level Arrival curve

Consider a FIFO system with per-flow queuing and DRR arbitration policy, with $n = 16$ queues sharing a link with rate $c = 1$ Gbps. Assume all flows have maximum packet length of $L^{\max} = 1.5$ KB and the queues have same quantum value $Q = L^{\max}$. Then, similarly to the previous case, we obtain a service curve β as in (4.26) offered to any per-flow queue.

Consider a flow with $\alpha(t) = rt + 2L^{\max}$ and minimum packet length of 0.5 KB. Since, the service curve is not rate-latency, we cannot apply the improvement in [62]. Hence, the state-of-the-art approach is to use the classical network-calculus bound, $h(\alpha, \beta)$; it is equal to 915.88 μs. Using our improved delay bound in Theorem 4.5, the delay bound is reduced to 743.88 μs, which improves the classical network-calculus bound by 18.8%.

4.8 Conclusion

We presented a theory to compute delay bounds for flows with packet-level arrival-curve, which improves the state-of-the-art bounds. The improvement is made possible by a novel modelling of packet-level arrival-curve with g-regularity and bit-level arrival-curve together with exploiting the information on the transmission rate. Our method of proof also led to delay improvement for flows with g-regularity constraint and bit-level arrival-curve. These results can be easily integrated with FP-TFA and PLP to improve the end-to-end delay bounds. In time-sensitive networks, this result can open a discussion on the operation of flow regulation: even though in such networks the traffic specification at a source is at the packet level, flow regulation is performed at the bit level, based on a bit-level arrival-curve derived from the flow constraints at the source. As a result of the analysis of this chapter, using packet-level traffic regulation leads to better delay bounds at the intermediate routers and switches. As the operation of bit-level regulation mechanisms is mainly based on full reception of a packet, e.g., as in IEEE802.1 Qcr Asynchronous Traffic Shaping, an implementation of packet-level regulation appears to be feasible and even simpler.

4.9 Notation

Term	Description
A_n	The arrival time of packet n to the FIFO system
c	The transmission rate of the output link
D_n	The departure time of packet n from the FIFO system
$h(w', w)$	The horizontal deviation from function w' to function w (Definition 3.3)
K	The maximum number of packets in each interval of TSN/DetNet traffic constraint
$L_f^{\max}$	Maximum packet length of flow f
$L_f^{\min}$	Minimum packet length of flow f
l_n	The length of packet n in bits
$N(t)$	The cumulative number of packets arrived at the FIFO system until time t (excluded)
Q_n	The start of transmission of packet n from the FIFO system
R	Rate of service curve, when β is rate-latency
T	Latency of service curve, when β is rate-latency
β	The service curve of the FIFO system
Δ^{A}	The response time bound on a flow with bit-level arrival-curve
Δ^{AG}	The response time bound on a flow with g-regularity constraint competing with an aggregate of flows with bit-level arrival-curve
Δ^{G}	The response time bound on a flow with g-regularity constraint
Δ^{pkt}	The response time bound on a flow with packet-level arrival-curve
τ	The interval duration in TSN/DetNet traffic constraint
$\mathbb{N}$	The set of positive integers, i.e., $\{1, 2, \dots, \}$
$\mathbb{R}^+$	The set of non-negative real numbers, i.e., $[0, \infty)$
$\mathscr{F}_{\mathrm{inc}}^0$	The set of wide-sense increasing functions such that $\forall w \in \mathscr{F}_{\mathrm{inc}}^0 :\ w(0) = 0$
w^+	The right limit of function w (Section 3.1.3)
w^-	The left limit of function w (Section 3.1.3)
$w^\downarrow$	The lower pseudo-inverse of function w (Section 3.1.1)
$w^\uparrow$	The upper pseudo-inverse of function w (Section 3.1.1)
$\lceil x \rceil$	The ceiling of x
$\lfloor x \rfloor$	The floor of x
$[x]^+$	$\max\{x, 0\}$
$\circ$	The function-composition operator, i.e., $(f \circ g)(x) = f(g(x))$
$1_{\{C\}}$	It is equal to 1 when the condition C is true and is equal to 0 otherwise.

5 Delay and Backlog bounds in TSN with Interleaved Regulators

Science is bound, by the everlasting vow of honour,
to face fearlessly every problem which can be fairly presented to it.
— William Thomson, 1st Baron Kelvin

As discussed in Chapter 1, the per-class mechanisms in TSN can lead to two key issues of burstiness cascade and cyclic dependency. To this end, computing delay upper-bounds for per-class networks is difficult, unless flows are regulated at every hop [67, 147, 16, 148]. This is why a TSN proposal is to regulate flows at every hop, using the concept of interleaved regulator introduced in [122] and analyzed in [123] (called IEEE 802.1Qcr Asynchronous Traffic Shaping within TSN). An interleaved regulator regulates individual flows without per-flow queuing.

In [122], an end-to-end delay bound is computed for a network of FIFO constant rate servers with aggregate multiplexing that uses interleaved regulators to avoid the burstiness cascade. However, this does not account for the multi-class nature of a TSN network and for a representative combination of queuing and scheduling mechanisms proposed by TSN, called credit-based shaper. The first goal of this chapter is to extend these calculations to a more generic TSN network. However, the calculations in [122] are very complex; extending them seems to be intractable unless some higher level of abstraction is used, as described below. The second goal of this chapter is to provide backlog bounds, which can be used to dimension buffers.

To address these goals, we use network calculus concepts such as a service-curve characterization of CBS and extend the results in [13] to include high-priority control-data traffic (CDT). We combine this with the max-plus representation of interleaved regulators proposed in [123]. Further, we use the result of [123] that the upper bound on the delay in the combination of an interleaved regulator following a FIFO system is no greater than the upper bound on the delay of the FIFO system. Overall, in this chapter we compute delay upper bounds for the CBS, the interleaved regulator and end-to-end delay bounds along with backlog bounds for the first two. Our main contributions are listed below.

1. we derive and formally prove novel credit upper-bounds for CBS (Theorem 5.1). This is a general result and is applicable to any number of AVB classes with CBS.
2. We obtain a service curve for every AVB class at a CBS system, extending a similar result in [13] by accounting for the presence of CDT (Theorem 5.2). The service curves are used to decouple the interleaved regulator from CBS and are essential to obtain the other results mentioned below.
3. We obtain a delay bound for the interleaved regulator (Theorem 5.3), by using the delay bound at a CBS subsystem and that an interleaved regulator does not increase the delay bound of a FIFO system [123].
4. We use the delay bound of the interleaved regulator to derive a service curve for the interleaved regulator and hence a backlog bound at the interleaved regulator.
5. We are the first to compute a tight end-to-end delay bound for a TSN network of this kind. We show that the end-to-end delay bound obtained is less than the sum of delay bounds computed at every switch along the path of a flow. Ignoring this, as is often done, leads to a gross overestimation of the worst-case end-to-end latency.

The rest of this chapter is organized as follow. Section 5.1 describes the system model. Section 5.2 provides: credit upper-bounds for CBS; a service curve for the CBS subsystem; a delay bound on the response time in the CBS subsystem; a delay bound for the interleaved regulator; and a tight end-to-end delay bound. Section 5.3 uses these results to derive backlog bounds. Section 5.4 provides case studies, shows the tightness of the bounds and the sub-additivity of the end-to-end delay bound. Section 5.5 concludes the chapter. For the reader's convenience, Section 5.6 gives the notation used throughout this chapter. Appendix B gives the proofs.

5.1 System Model

We consider a network with a set $\mathscr{S}$ of nodes (switches and hosts) along with a set of flows, F, between hosts. Hosts are sources or destinations of flows. There are four types of flows, namely, control-data traffic (CDT), AVB class 1, AVB class 2, and best effort (BE) [149] in decreasing order of priority. Flows of AVB classes 1 and 2 are often called respectively class A and class B [150, 13]. We focus on delay and backlog bounds for AVB traffic. We assume a subset of TSN functions as described next.

5.1.1 Architecture of a TSN node

We assume that contention occurs only at the output port of a TSN node. Each node output port performs per-class scheduling with eight classes: one for CDT, one for class A traffic, one for class B traffic, and five for BE traffic denoted as BE0, ..., BE4 (TSN standard [151]). In addition each node output port also performs per-flow regulation for AVB flows using an

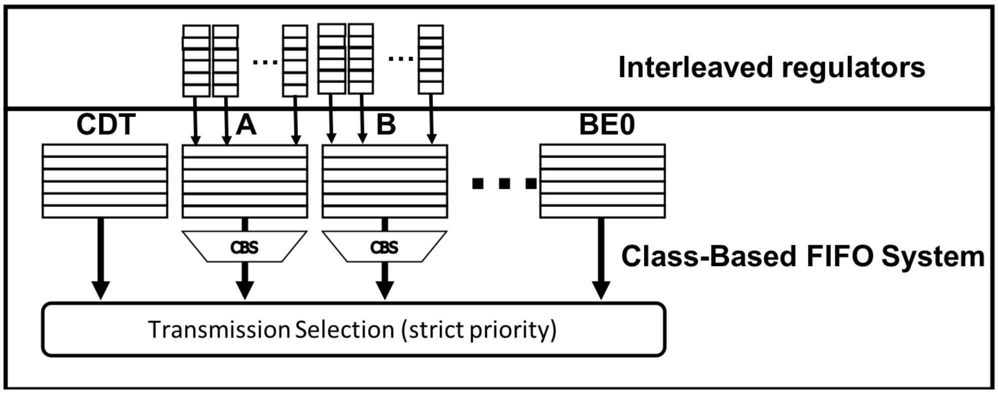


Figure 5.1: Architecture of the considered TSN node output port.

interleaved regulator. Thus, at each output port of a node, there is one interleaved regulator per-input port and per-class [123, 122]. The detailed picture of scheduling and regulation at a node output port is given by Fig. 5.1. The packets received at a node input port for a given class are enqueued in the respective interleaved regulator at the output port. Then, the packets from all the flows, including CDT and BE flows, are enqueued in a class based FIFO system (CBFS). The CBFS includes two CBS subsystems, one for each class 1 (A) and class 2 (B), operation of which is described in [35, Annex L] and recalled in Section 5.1.2. The CDT and BE_0-BE_4 flows in the CBFS are served by separate FIFO subsystems. Then, packets from all flows are served by a transmission selection subsystem that serves packets from each class based on its priority. All subsystems are non-preemptive.

Guarantees for AVB traffic can be provided only if CDT traffic is bounded; we assume that the CDT traffic from node i to node j has a token bucket arrival curve $r_{ij}t + b_{ij}$. How to derive such arrival curves involves other TSN mechanisms and is outside the scope of this chapter.

Fig. 5.2 shows a part of a TSN network with three switches serving four flows of class A. In switches SW_1 and SW_2 two flows are coming from two different input ports, thus, they use different interleaved regulators. The flows entering switch SW_3 from switch SW_1 are going to different output ports, and use different interleaved regulators.

5.1.2 Operation of CBS

The CBS of an AVB class x has two parameter, the *idleslope* and the *sendslope*, $I^x > 0$. The sendslope, $S^x < 0$, is often defined by $I^x - S^x = c$, where c is the line transmission rate. The idleslope is the rate guaranteed to class x. Thus, it is assumed that $\sum_{i=1}^{p} I^i < c$ with p being the total number of AVB classes. Packets are scheduled according to the following rules (we repeat here the description in [152]):

R1: If the transmission line is free, the scheduler transmits a packet of the highest priority class that satisfies all the conditions: 1) its queue is not empty; 2) it has a non-negative credit

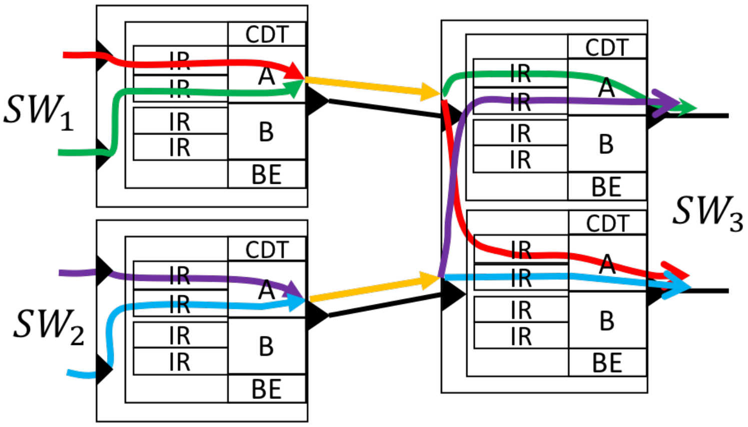


Figure 5.2: Illustration of the queuing policy in interleaved regulators (IR) by TSN switches for four flows of class A.

if it is an AVB class.

R2: The credit of the AVB class x reduces linearly with rate the sendslope, S^x, if x transmits.

R3: The credit of the AVB class x increases linearly with rate I^x, when the following conditions hold simultaneously for class x: 1) CDT is not transmitting; 2) its queue is not empty; and 3) other AVB or BE classes are transmitting.

R4: The credit of an AVB class remains constant if CDT is transmitting packets.

R5: Whenever class x has a positive credit and its queue becomes empty, the credit is set to zero; this is called *credit reset*. If the credit is negative and the queue becomes empty, the credit increases with rate I^x until the zero value.

Let $V_x(t)$ denote the value of the credit counter for AVB class x at time $t \geq 0$. We assume that the system is idle at time 0 and $V_x(0) = 0$. The function $V_x()$ can take positive or negative values and is continuous, except at credit reset times, which by R5, can occur only when the queue of class x becomes empty. At all other times, it linearly increases, decreases or remains constant. Negative credit value is reached when a packet with length larger than $-\frac{cV_x(t)}{S^x}$ starts its transmission at time t.

5.1.3 Flow Regulation

Following [122], we assume that flows are regulated at their source, according to either leaky bucket (LB) or LRQ. As mentioned in Section 3.2.2, the LB-type regulation forces flow f to conform to the arrival curve $r_f t + b_f$. Recall from Section 4.2.2 that the LRQ-type regulation with rate r_f ensures that the time separation between two consecutive packets of sizes l_n and l_{n+1} is at least l_n/r_f. Note that if flow f is LRQ-regulated, it satisfies the arrival curve constraint $r_f t + L_f$ where L_f is its maximum packet size (but the converse may not hold). For

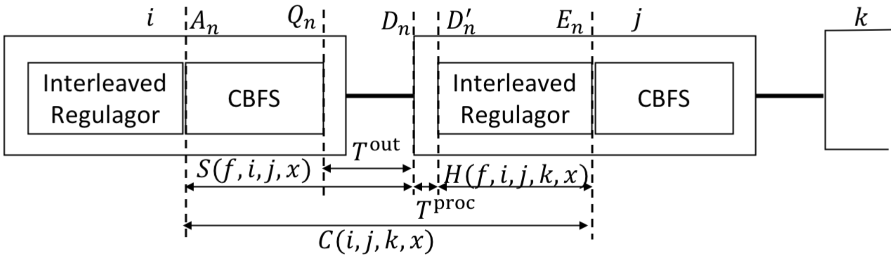


Figure 5.3: Timing Model in TSN

an LRQ regulated flow we set $b_f = L_f$. We also call M_f the minimum packet size of flow f. We assume that, at the source hosts, the traffic satisfies its regulation constraint, i.e. we can ignore the delay due to interleaved regulator at hosts.

According to [123], at each switch implementing an interleaved regulator, packets of multiple flows are processed in one FIFO queue; the packet at the head of the queue is regulated based on its regulation constraints; it is released at the earliest time at which this is possible without violating the constraint. The regulation type and parameters for a flow are the same at its source and at all switches along its path.

5.1.4 Other Notations and Definitions

The indices for nodes, e.g., i, j, k lie in $[1, |\mathscr{S}|]$. A directed link from node i to j is denoted by (i, j) with a capacity of c_{ij}. Also, $n \in \mathbb{N}$ is used as an index for packets, $f \in F$ is used as an index for flows, and $x \in \{A, B, E\}$ is used as an index for AVB classes A and B, and BE flows, respectively. The set of packets belonging to flow f is N_f. The set of flows of class x going from node i to node j is denoted by F^x_{ij} and those that continue to node k, by F^x_{ijk}. The maximum packet size of flows in F^x_{ij} is defined as L^x_{ij}. The flows in F^x_{ijk} use the same CBFS in node i and interleaved regulator in node j. As mentioned in Section 5.1.1, for each output port, there is a per-class per-input port interleaved regulator. Thereby, the interleaved regulator in node j connected to link (j, k) indicates an output port of node j connected to node k.

Fig. 5.3 shows the various delays of a packet n of a flow in F^x_{ijk}. We see five important time instants: (1) A_n is the arrival time of packet n in CBFS, (2) Q_n is the time that packet n starts transmission from CBFS, (3) D_n is the time that packet n is received at a node, (4) D'_n is the time that packet n is enqueued in the interleaved regulator, and (5) E_n is the time that packet n leaves the interleaved regulator.

$(D'_n - D_n)$ is the processing time at node j, which is defined as the delay from the reception of the last bit of a packet, coming from node i, to the time the packet is enqueued at the interleaved regulator. We assume that $D'_n - D_n \in \left[T^{\text{proc,min}}_{ij}, T^{\text{proc,max}}_{ij}\right]$. $(D_n - Q_n)$ is the output delay for packet n traversing form node i to node j, which is defined as the time required from the selection of a packet for transmission from a CBFS queue of node i to the reception of the last bit of the packet by the node j. Also, $D_n - Q_n = \frac{l_n}{c_{ij}} + T_{ij}^{\text{out},n}$, where l_n is the length of packet n and $T_{ij}^{\text{out},n}$ is in $[T_{ij}^{\text{out, min}}, T_{ij}^{\text{out, max}}]$.

We compute the following bounds for packets of AVB flows belonging to class x going from node i to j (see Fig. 5.3):

- $S(f,i,j,x)$: upper bound on the response time for flow f in CBFS, i.e. on $(D_n - A_n)$;
- $H(f,i,j,k,x)$: upper bound on the response time for flow f in the interleaved regulator at node j 's output port for link (j,k), i.e. a bound on $(E_n - D'_n)$;
- $C(i,j,k,x)$: upper bound for all flows on the response time in the combination of the CBFS at node i and the interleaved regulator at node j for link (j,k), i.e. a bound on $(E_n - A_n)$ for all flows.

5.2 Delay Bounds in TSN

The aim of this section is to compute bounds on the delays that an AVB flow experiences through CBFS, $S(f,i,j,x)$, and interleaved regulator at a node, $H(f,i,j,k,x)$. To do so, we derive a service curve of CBFS for an AVB flow, in presence of CDT with an LB arrival curve; this is directly affected by the upper bounds on the credit counters values of the CBS, denoted as credit upper bounds. Therefore, in Section 5.2.1, we first present our novel result on credit bound and then in Section 5.2.2, we obtain a service curve of CBFS for an AVB flow, in presence of CDT with an LB arrival curve. Then, in Section 5.2.3, we use this service curve to compute a bound on the response time for an AVB flow in the CBFS of a node, i.e., $S(f,i,j,x)$. Using $S(f,i,j,x)$, we compute $C(i,j,k,x)$. Consequently, we can compute a bound on the response time of an interleaved regulator, $H(f,i,j,k,x)$ in Section 5.2.4, and therefore we have all the elements to compute a bound on the delay of a single TSN node. We also compute a tight end-to-end delay bound for an AVB flow in Section 5.2.5.

5.2.1 Novel Credit Bound for CBS

In this section, we present our novel and general result on credit bound for CBS for multiple classes $x = 1, \ldots, p$; in the delay analysis we use part of this result for the two AVB classes[1].

Regarding credit upper bounds, two sets of results were published. The first, by J. Cao et al, (“J-bounds", [153]) provides tight credit upper bounds for the first two AVB classes. Note that since these bounds are tight, they cannot be improved. The second, by H. Daigmorte et al (“H-bounds", [152]), applies to any number of AVB classes. For the top priority AVB class, J- and H-bounds are identical. For the second priority class, J-bounds are smaller than

---

[1] This section is taken from our published paper [62] and it is adapted to the content of this chapter. The results in the original paper can be applied to several integration policies mentioned in [152] including the preemption or non-preemption of CDT over the rest of the classes.

H-bounds. For third and lower priority classes, only H-bounds are available. The credit bound of a class depends on lower priority classes only via their maximum packet length. Thus, the J-bounds can be applied for the first two priority classes for any system, independently of the total number of classes. For the second priority class, the H-bound is larger than the J-bound, which suggests that the H-bounds can be improved for all classes.

The J-bounds in [153] for the first two AVB classes are,

$$V_1(t) \leq \bar{L}\frac{I^1}{c} := V_1^{\max,\mathrm{J}}, \tag{5.1}$$

$$V_2(t) \leq \frac{I^2}{c}\Big(L^{BE} + L^1 + L^{BE}\frac{I^1}{-S^1}\Big) := V_2^{\max,\mathrm{J}}, \tag{5.2}$$

where L^i and L^{BE} are maximum packet lengths of AVB class i and BE. The H-bounds in [152] apply to any value of p and give, for $x = 1, \ldots, p$:

$$L^x\frac{S^x}{c} \leq V_x(t) \leq \frac{\bar{L}^x}{c}\sum_{j=1}^{x} I^j - \sum_{j=1}^{x-1} S^j\frac{L^j}{c} := V_x^{\max,\mathrm{H}}, \tag{5.3}$$

where $\bar{L}^x = \max(L^{BE}, L^{>x})$, and $L^{>x}$ is the maximum packet length of the classes having less priority than class x.

For class 1, the J-bounds and H-bounds are identical, i.e., $V_1^{\max,\mathrm{J}} = V_1^{\max,\mathrm{H}}$. For class 2, we have $V_2^{\max,\mathrm{J}} < V_2^{\max,\mathrm{H}}$.

Theorem 5.1 (Improved Credit Bounds)**.** *The credit of an AVB class x, $V_x(t)$, is upper bounded, $\forall t \geq 0$, by:*

$$V_x^{\max} = \frac{I^x}{c(c - \sum_{j=1}^{x-1} I^j)}\left(c\bar{L}^x - \sum_{j=1}^{x-1} S^j L^j\right). \tag{5.4}$$

The proof is in Appendix B.1.

Remark. The bounds are valid for any number of classes, p, existing in the system as long as $\bar{L}^x$, $\forall x$ is appropriately determined for the corresponding system.

For the two top priority classes, the bounds, $V_x^{\max}$ (Eq. (5.4)), are equal with the J-bounds. Hence, they are tight for the top two priority classes, i.e., for each set of parameters and each class 1, 2, there is a scenario for which the credit counter attains the bound. The proof of tightness is in [153]. Note that, $V_x^{\max}$ depends on lower priority classes only through the parameter $L^{>x}$, and not on their number. We show that our bounds, $V_i^{\max}$, are always better than the existing bounds, $V_i^{\max,\mathrm{H}}$, for classes $i \geq 3$.

Proposition 5.1. *For any AVB class $x \geq 3$, $V_x^{\max} < V_x^{\max,\mathrm{H}}$.*

Proof. After some algebra we find

$$V_x^{\max,\mathrm{H}} - V_x^{\max} = \frac{c - \sum_{j=1}^{x} I^j}{c\left(c - \sum_{j=1}^{x-1} I^j\right)} \left(\bar{L}^x \sum_{j=1}^{x-1} I^j - \sum_{j=1}^{x-1} S^j L^j\right). \tag{5.5}$$

By hypothesis, $c > \sum_{j=1}^{x} I^j$. Since $I^j > 0$, $S^j < 0$ and $x \geq 2$, it follows that $\bar{L}^x \sum_{j=1}^{x-1} I^j - \sum_{j=1}^{x-1} S^j L^j > 0$, thus the last term of Eq. (5.5) is strictly positive. □

5.2.2 Service Curve Offered by CBFS to AVB flows

The following theorem provides service curves offered by a CBFS at a TSN node, for AVB flows in presence of CDT flows with LB arrival curve. In [13], the authors compute service curves for AVB flows according to the IEEE AVB standard [18], i.e., in absence of CDT. Note that service curves for AVB flows in TSN are proposed in [145]; however in their proof credit reset is not considered, and we show in Section 5.4 that it leads to incorrect response time bound for CBFS. We obtain different service curves than [145] and we use them to obtain tight delay bounds.

Theorem 5.2. *Assume a node i and a link (i, j), where the CDT has an LB arrival curve with parameters (r_{ij}, b_{ij}) and the line rate is c_{ij}. Then, the CBFS offers to class A flows a rate-latency service curve with parameters,*

$$T_{ij}^A = \frac{1}{c_{ij} - r_{ij}} \left(\bar{L}_{ij}^A + b_{ij} + \frac{r_{ij} \bar{L}_{ij}}{c_{ij}}\right), \tag{5.6}$$

$$R_{ij}^A = \frac{I_i^A (c_{ij} - r_{ij})}{I_{ij}^A - S_{ij}^A}, \tag{5.7}$$

where I_{ij}^A and S_{ij}^A are the idle slope *and* send slope*, correspondingly, of the CBS for class A and link (i, j), $\bar{L}_{ij}^A = \max(L_{ij}^B, L_{ij}^E)$, and $\bar{L}_{ij} = \max(L_{ij}^A, L_{ij}^B, L_{ij}^E)$. Similarly for class B flows, CBFS offers a rate-latency service curve with parameters,*

$$T_{ij}^B = \frac{1}{c_{ij} - r_{ij}} \left(L_{ij}^A - \frac{c \bar{L}_{ij}^B}{S_{ij}^A} + b_{ij} + \frac{r_{ij} \bar{L}_{ij}}{c_{ij}}\right), \tag{5.8}$$

$$R_{ij}^B = \frac{I_{ij}^B (c_{ij} - r_{ij})}{I_{ij}^B - S_{ij}^B}, \tag{5.9}$$

where I_{ij}^B and S_{ij}^B are the idle slope *and* send slope*, correspondingly, of the CBS for class B and link (i, j).*

The proof is available in Appendix B.2.

5.2.3 Upper Bound on the Response Time in CBFS

The rate-latency service curve offered by CBFS at node i for link (i,j) to class $x \in \{A,B\}$ has parameters R_{ij}^x, T_{ij}^x, as calculated in Theorem 5.2. Also, let $b_{ij}^{\text{tot},x} = \sum_{f\in F_{ij}^x} b_f$. Then, applying Lemma 4.2 and Theorem 4.5, we obtain an upper bound on the response time for a flow f at a CBFS of node i.

Corollary 5.1. *A tight upper bound on the response time in the CBFS of node i (following the interleaved regulator) for flow f of class $x \in \{A,B\}$, going from node i to j, is:*

$$S(f,i,j,x) = T_{ij}^x + \frac{b_{ij}^{\text{tot},x} - \psi_f}{R_{ij}^x} + \frac{\psi_f}{c_{ij}} + T_{ij}^{\text{out,max}}, \tag{5.10}$$

where the parameter ψ_f depends on the flow f and the type of regulator, namely, for LRQ: $\psi_f = L_f$ and for LB: $\psi_f = M_f$.

It is known from [123] that for all flows belonging to class x sharing the same CBFS queue at node i and interleaved regulator at node j (e.g., for link (j,k)), we have

$$C(i,j,k,x) = \sup_{f'\in F_{ijk}^x} S(f',i,j,x) + T_{ij}^{\text{proc,max}}. \tag{5.11}$$

Therefore, the following Corollary is a direct result.

Corollary 5.2. *Assume flows of class $x \in \{A,B\}$, going from node i to j, and enqueued in the interleaved regulator at node j for link (j,k). An upper bound, for each flow, of the combination of the response time in CBFS of node i (following the interleaved regulator of i) and the interleaved regulator at node j for link (j,k) is given by:*

$$C(i,j,k,x) = T_{ij}^x + \frac{b_{ij}^{\text{tot},x}}{R_{ij}^x} + T_{ij}^{\text{out,max}} + \sup_{f'\in F_{ijk}^x}\left(\frac{\psi_{f'}}{c_{ij}} - \frac{\psi_{f'}}{R_{ij}^x}\right) + T_{ij}^{\text{proc,max}}, \tag{5.12}$$

where for LRQ: $\psi_f = L_f$ and for LB: $\psi_f = M_f$.

5.2.4 Bound on the Response Time in the Interleaved Regulator

The following theorem proves an upper bound on the response time in the interleaved regulator, $H(f,i,j,x)$.

Theorem 5.3. *An upper bound on the response time for flow f of class $x \in \{A,B\}$ in the interleaved regulator at node j for link (j,k) that follows the CBFS of node i is:*

$$H(f,i,j,k,x) = C(i,j,k,x) - \frac{M_f}{c_{ij}} - T_{ij}^{\text{out,min}} - T_{ij}^{\textit{proc, min}}. \tag{5.13}$$

The proof is given in Appendix B.3.

Remark. It is shown numerically in Section 5.4, that H is tight for the flow f that achieves the maximum response time at the CBFS, i.e., for which $S(f,i,j,x) = C(i,j,k,x)$.

5.2.5 Upper Bound on the End-to-End Delay

Assume an AVB flow f routed through the nodes $(i_1,...,i_k)$, where the source is i_1 and destination is i_k. It is assumed that the arrival curves of the generated flows in source conform to the flows' regulation policies, and thus the flows do not experience delay at the interleaved regulators of the source nodes. An upper bound on the end-to-end delay for flow f of class x, namely, D_f^x, is,

$$D_f^x = \sum_{j=1}^{k-2} C(i_j, i_{j+1}, i_{j+2}, x) + S(f, i_{k-1}, i_k, x). \tag{5.14}$$

D_f^x can be easily computed by using Eqs. (5.10), (5.12). In Section 5.4.5, we show numerically that this bound is tight, in the sense that we exhibit an example that it achieves this bound.

5.3 Backlog Bounds

In this section, we determine an upper bound on the backlog for each AVB class of interleaved regulator and CBFS.

5.3.1 Backlog Bound on Interleaved Regulator

As mentioned in Section 3.2.3, computing upper bounds on the backlog requires information on arrival and service curves [25].

Service Curve Offered by Interleaved Regulator

Recall the "impulse" function $\delta_D(t)$[2] from Section 3.2; it is known that a service curve offered by a FIFO system which guarantees a maximum delay D is equal to $\delta_D(t)$. The interleaved regulator is a FIFO system, for which a delay upper bound is computed in Section 5.2.4. Therefore, a service curve offered by the interleaved regulator for class x, at node j for link (j,k), that follows a CBFS of node i is $\delta_{D(i,j,k,x)}(t)$, where $D(i,j,k,x) = \sup_{f' \in F_{ijk}^x} H(f', i, j, l, x)$ is computed using Theorem 5.3.

Arrival Curve of Interleaved Regulator Input

The output flows of the upstream CBFS (node i) may not share the same interleaved regulator. Let us consider the interleaved regulator of node j for link (j,k) that follows the CBFS of node i. Suppose that r_s and b_s are the sum of rates and bursts of the flows $f' \in F_{ijk}^x$ for $x \in \{A,B\}$.

---

[2] defined as $\delta_D(t) = 0,\ 0 \le t \le D$, else $\delta_D(t) = +\infty$.

In addition, r_w and b_w are the sum of rates and bursts of the flows that do not use the same interleaved regulator in downstream node with the previous flows. The CBFS offers a rate-latency service curve with parameters (R^x_{ij}, T^x_{ij}) to the class $x \in \{A, B\}$ (Theorem 5.2). Then, by Theorem 3.4, the output arrival curve of the former flows is an LB one, $r_s t + b_{out}$ with $b_{out} = b_s + r_s(T^x_{ij} + \frac{b_w}{R^x_{ij}})$.

On the other hand, the upstream line has constant rate, c_{ij}. Therefore, it also enforces an arrival curve to the input of the interleaved regulator equal to $c_{ij} t + \sup_{f' \in F^x_{ijk}} \{L_{f'}\}$.

As the CBFS follows the interleaved regulator, the input arrival curve of the interleaved regulator is,

$$\alpha(t) = \min\Big(c_{ij} t + \sup_{f' \in F^x_{ijk}} \{L_{f'}\}, r_s t + b_s + r_s(T^x_{ij} + \frac{b_w}{R^x_{ij}})\Big). \tag{5.15}$$

Backlog Bound on Interleaved Regulator

By Theorem 3.2, the backlog bound is calculated as $v(\alpha, \beta) = \sup_{s \geq 0}\{\alpha(s) - \beta(s)\}$, where $\alpha(t)$ is the arrival curve and $\beta(t)$, the service curve. By replacing the arrival and service curves obtained in the two previous subsections, we obtain the backlog bound of the interleaved regulator for class $x \in \{A, B\}$ at node j for link (j, k) that follows the CBFS of node i, denoted as $B^{\text{IR},x}_{ijk}$ and given:

$$B^{\text{IR},x}_{ijk} = \min\Big(c_{ij} D(i,j,k,x) + \sup_{f' \in F^x_{ijk}} \{L_{f'}\}, r_s D(i,j,k,x) + b_s + r_s(T^x_{ij} + \frac{b_w}{R^x_{ij}})\Big), \tag{5.16}$$

where T^x_{ij}, R^x_{ij} are computed in Theorem 5.2.

5.3.2 Backlog Bound on Class-Based FIFO System

Consider all flows $f \in F^x_{ij}$. The input of the CBFS for class x has an arrival curve equal to the sum of all α_f. Using Theorem 5.2 and following a process similar to the one followed for the interleaved regulator, the backlog bound of the CBFS at node i for link (i, j) and class x, denoted as $B^{\text{CBFS},x}_{i,j}$ is

$$B^{\text{CBFS},x}_{i,j} = \sum_{f' \in F^x_{ij}} b_{f'} + \sum_{f' \in F^x_{ij}} r_{f'} T^x_{ij}. \tag{5.17}$$

5.4 Case Study

In this section, we apply the results obtained in the previous sections to practical TSN networks (Fig. 5.4, 5.6). We highlight the tightness of the delay bounds obtained and the sub-additivity property of the end-to-end delay bound.

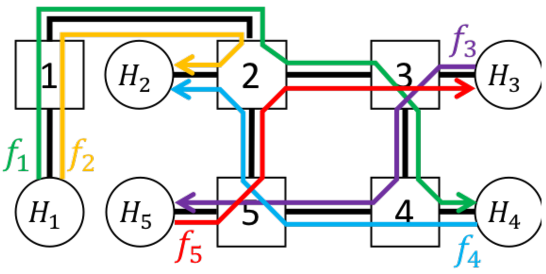
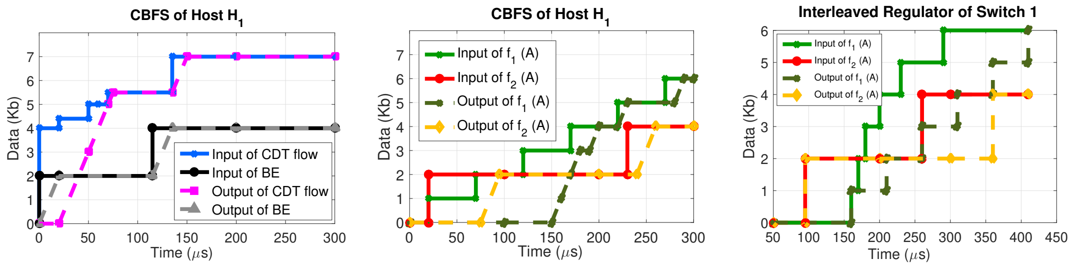


Figure 5.4: A practical TSN network used for the case study.



(a) CDT and BE flows (b) f_1 and f_2 flows (class A) (c) f_1 and f_2 flows (class A)

Figure 5.5: Cumulative data input and output curves for the CBFS of H_1 and interleaved regulator of switch 1, with respect to Fig. (5.4).

5.4.1 TSN Network Setup and Flows

We use the network shown in Fig. 5.4. It consists of five switches labeled 1-5, and five hosts (as sources and destinations of flows), namely H_1-H_5, with five class-A flows f_1-f_5. Flow f_1 is LRQ-regulated with rate $r_{f_1} = 20$ Mbps and has maximum packet length $L_{f_1} = 1$ Kb. Flows f_2-f_5 are LRQ regulated with rate 20 Mbps and maximum packet length 2 Kb. It is assumed that on each output port there is a CDT flow with an LB arrival curve (20 Mbps, 4 Kb), and a BE flow with maximum packet length of 2 Kb. As is shown in the Fig. 5.4, the network introduces circular dependency among the flows, in which case obtaining end-to-end delay bounds for the flows without the use of interleaved regulators is difficult.

For ease of presentation, we assume that the CBFS has only three classes: CDT, class A, and one BE. Moreover, $T_{ij}^{\text{out,max}}$ and $T_{ij}^{\text{proc,max}}$ are zero in all switches i, links (i, j) and all packets of a same flow have the same size. The line rate is equal to 100 $Mbps$. The parameters of CBS are $I_{ij}^{A} = 50$ Mbps and $S_{ij}^{A} = -50$ Mbps, $\forall\ i, j, H_1 - H_5$. We are interested in studying the worst case response time of flow f_1 in CBFS of host H_1 and its corresponding interleaved regulator in switch 1. Also, we compute the theoretical end-to-end delay bound of this flow and show its sub-additivity property.

5.4.2 Computation of Theoretical Bounds

We compute the obtained upper bounds for the response time in CBFS and interleaved regulator for flow f_1, and the backlog bounds for the host H_1 and switch 1. According to Corollary 5.1, the bound on the CBFS response time for flow f_1 in the host H_1 is $S(f_1, H_1, 1, A) = 140\ \mu$s. Also, from Theorem 5.3, the bound on the response time in interleaved regulator for flow f_1, enqueued in the output port for link $(1,2)$ on switch 1 is $H(f_1, H_1, 1, 2, A) = 130\ \mu$s. Also, the backlog bound for the same interleaved regulator (Eq. (5.16)) is 11.4 Kb. The backlog bound for CBFS of class A in host H_1 is 6.2 Kb (Eq. (5.17)). To compute the end-to-end delay, we use Eq. (5.14). Using Eq. (5.12), we find that $C(H_1, 1, 2, A) = C(1, 2, 3, A) = C(2, 3, 4, A) = C(3, 4, H_4, A) = 140\ \mu$s, and $S(f_1, 4, H_4, A) = 140\ \mu$s. Thus, for flow f_1 of class A we have the upper bound on delay $D^A_{f_1} = 700\ \mu$s.

5.4.3 Numerical Example of Tightness

Next, we show how these bounds are tight by presenting a particular series of packet arrivals as shown in Fig. 5.5. This figure shows the input and output curves related to f_1, f_2, CDT and BE flows in host H_1 and switch 1. A step in the input curve indicates the time of reception of the entire packet. According to Fig. 5.5a, at time 0, a packet of BE arrives and starts being transmitted. At time 0^+, a burst of CDT traffic arrives and then for time $t > 0$, CDT traffic continues to arrive with rate 20 Mbps up to the time 75 μs. The transmission of CDT traffic at time 0^+ is blocked by the transmission of the BE packet as all switches are non-preemptive. At time 20 μs, CDT traffic has accumulated a backlog and starts its transmission.

From Fig. 5.5b, we see that time 20 μs is the start of the backlog period of class A since a packet of flow f_2 and a packet of f_1 arrive, with first of the two being the former. The first packet of flow f_2 reaches at time 95 μs the interleaved regulator in switch 1 for link $(1,2)$ that implies a response time of 75 μs for flow f_2 in the CBFS of host H_1. The first packet of flow f_1 finishes its transmission at time 160 μs from CBFS in H_1, due to its earlier blockage by the CDT and f_2 traffic. This implies a response time of 140 μs for flow f_1 in CBFS of H_1, i.e., equal to the bound in Section 5.4.2.

Remark. Using the service curve computed in Theorem 1 of [145] gives a bound equals to 135 μs for the response time of CBFS for f_1. In the described scenario, flow f_1 faced a response time of 140 μs that is higher than 135 μs.

From Fig. 5.5c, we notice that the worst-case response time in the interleaved regulator for flow f_1 at switch 1 is for the packet that arrives at time 230 μs. This packet is declared eligible by the interleaved regulator at time 360 μs. This implies the response time of 130 μs in the interleaved regulator that is the upper bound for flow f_1 as computed in Section 5.4.2. The maximum response time seen in Fig. 5.5c for flow f_2 is for its packet that arrives at time 260 μs at the interleaved regulator at switch 1 and is equal to 100 μs.

Note that this packet of flow f_2 could have been declared eligible by the interleaved regulator

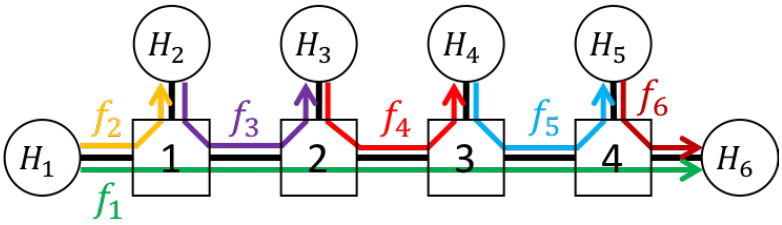


Figure 5.6: TSN network for tightness of end-to-end delay bound.

already at 260 μs but is blocked by preceding packets of flow f_1 that were not yet eligible at that time. Based on figures 5.5b and 5.5c, we observe that packets of flow f_1 experience a maximum delay of 140 μs from the time being enqueued in the CBFS of H_1 to the time being declared eligible by the interleaved regulator at switch 1 (equal to $C(H_1, 1, 2, A)$, computed in Section 5.4.2).

The maximum observed backlog for class A used in the CBFS at the output port of H_1 is equal to 4Kb during times 70 μs to 75 μs, which is 65% of the computed bound. Furthermore, the maximum backlog observed in the interleaved regulator at output port of switch 1 is equal to 5Kb during times 230 μs to 260 μs, which is 43% of the computed bound.

5.4.4 Sub-additivity of End-to-End Delay Bound

In TSN, the common way of computing the end-to-end delay bound is by adding the delay bounds of each switch in the path of a flow. However, Eq. (5.14) provides a much better upper bound. To show this, we first compute the delay upper bound for switch i following switch j and followed by switch k in the path of a flow f of class x. The bound is given by

$$d^{x,f}_{j,i,k} = H(f, j, i, k, x) + S(f, i, k, x) + T^{\text{proc,max}}_i, \tag{5.18}$$

where for i being a source, $H(f, j, i, k, A)$ is equal to zero. Considering $T^{\text{proc,max}}_i$ equal to zero in this case, an end-to-end delay bound for flow f_1 over the path $(H_1, 1), (1, 2), (2, 3), (3, 4), (4, H_4)$ can be computed as $(0 + 140) + 4 \times (130 + 140) = 1220$ μs.

From Section 5.4.3, we know that an upper bound on the end-to-end delay for flow f_1 is 700 μs which is 57% of 1220 μs, obtained using Eq. 5.18.

5.4.5 Tightness of End-to-End Delay Bound

Consider the network shown in Fig. 5.6, having four switches labeled 1 - 4, and six hosts, namely H_1 - H_6, with six class A flows f_1 - f_6. The assumptions on $f_1 - f_5$, CDT and BE traffic are as in Section 5.4.1 and f_6 is similar to $f_2 - f_5$.

To show the tightness of end-to-end delay bound of Eq. (5.14), we claim that each pair of f_1,

f_3 at switch 1, f_1, f_4 at switch 2, f_1, f_5 at switch 3, and f_1, f_6 at switch 4 experience the same input/output curves as the pair of flows f_1, f_2 in Fig. 5.5 but appropriately shifted in time, so that they take place sequentially. Thus, flow f_1 has a delay of 140 μs from the time being enqueued in the CBFS of H_1 to the time declared eligible from the interleaved regulator at 1. The same delay is experienced by flow f_1 at the rest pairs of switches in its path. Similar to Section 5.4.3, the response time of f_1 at CBFS of switch 4 is equal to 140 μs. Therefore, the end-to-end delay for f_1 is equal to $4 \times 140 + 140 = 700$ μs, which is equal to the bound computed from Eq. (5.14).

5.5 Conclusion

We have provided a set of formulas for computing bounds on end-to-end delay and backlog for class A and class B traffic in a TSN network that uses CBS and ATS. The bounds are rigorously proven, while we provide a representative case study that highlights the tightness of the delay bounds provided and shows the sub-additivity of the end-to-end delay bound.

5.6 Notation

Term	Description
A_n	The arrival time of packet n in CBFS.
c_{ij}	The rate of link (i,j).
$C(i,j,k,x)$	An upper bound on the response time of all flows in the combination of the CBFS at node i and the interleaved regulator at node j for link (j,k) (Fig. 5.3).
D_n	The time that packet n is received at a node.
D'_n	The time that packet n is enqueued in the interleaved regulator.
E_n	The time that packet n leaves the interleaved regulator.
F^x_{ij}	The set of flows of class x going from node i to node j.
F^x_{ijk}	The set of flows of class x going from node i to node j and continue to node k.
$H(f,i,j,k,x)$	An upper bound on the response time for flow f in the interleaved regulator at node j 's output port for link (j,k) (Fig. 5.3).
I^x	The CBS idleslope for class x.
L^x	The maximum packet size of class x.
L_f	The maximum packet size of flow f.
L^x_{ij}	The maximum packet size of flows in F^x_{ij}.
M_f	The minimum packet size of flow f.
l_n	The length of a packet n.
$\mathbb{N}$	The set of positive integers, i.e., $\{1,2,\ldots,\}$.
Q_n	The time that packet n starts transmission from CBFS.
N_f	The set of packets belonging to flow f is N_f.
$\mathscr{S}$	The set of all nodes.
$S(f,i,j,x)$	An upper bound on the response time for flow f in CBFS of link (i,j) (Fig. 5.3).
S^x	The CBS sendslope for class x.
T^{proc}_{ij}	The processing delay from the reception of the last bit of a packet, coming from node i, to the time the packet is enqueued at the interleaved regulator of node j, which is $\in [T^{\text{proc,min}}_{ij}, T^{\text{proc,max}}_{ij}]$ (Fig. 5.3).
T^{out}_{ij}	The output delay from the selection of a packet for transmission from a CBFS queue of node i to the reception of the last bit of the packet by the node j, which is $\in \left[T^{\text{out, min}}_{ij}, T^{\text{out, max}}_{ij}\right]$ (Fig. 5.3).
$V_x(t)$	The value of the credit counter for AVB class x at time $t \geq 0$.
$V^{\text{max,H}}_x(t)$	The H-bound, i.e., the credit upper-bound by [152], for AVB class x.
$V^{\text{max,J}}_x(t)$	The J-bound, i.e., the credit upper-bound by [153], for AVB class x.

6 Packet Reordering

There should be a place for everything, and everything in its place.
— Isabella Beeton

As discussed in Chapter 1, re-sequencing buffers are typically used to restore packet order and provide in-order packet delivery in time-sensitive networks. A re-sequencing buffer uses the assumption that the source increments a sequence number field by 1 for every packet of the flow. Early packets are stored until all packets with smaller sequence numbers arrive [52, 53, 54]. A timer is used to limit the waiting time of a packet in the re-sequencing buffer, as otherwise the loss of a packet in the network would cause indefinite holding of packets with larger sequence numbers. In time-sensitive networks, flows require a guarantee on worst-case delay and delay jitter, together with zero congestion-loss (no packet is discarded due to buffer overflow). Using network calculus as seen in chapters 3 and 5, the control or management plane computes worst-case delay and jitter bounds, together with the buffer sizes required for zero congestion-loss; such computations are currently done without taking into account the impact of re-sequencing buffers.

The main goal of this chapter is to bridge this gap and provide a theory to compute worst-case performance guarantees in presence of packet reordering and with re-sequencing buffers. Specifically, a first issue is to find appropriate per-flow information that enables proper setting of timeout value and buffer size at a re-sequencing buffer, such that deterministic guarantees hold for a time-sensitive flow, namely, no packet is lost due to spurious timeout or buffer overflow and packets are delivered in-order. A second issue is how to compute such per-flow information. A third issue is the effect of the re-sequencing buffers on worst-case delay and on delay jitter. Furthermore, an intriguing topic is to study the aforementioned issues in the interconnection of various network elements and intermediate re-sequencing buffers. A first challenge is how to systematically compute the propagation of per-flow information in a sequence of network elements that affect the re-sequencing parameters. The second challenge is to evaluate the impact of intermediate re-sequencing buffers on the end-to-end worst-case delay and delay jitter.

To address the first and second issues, we need appropriate metrics for per-flow reordering. We show in Theorem 6.2 that one of the metrics in RFC4737 [59], the reordering late time offset (RTO, the definition of which is recalled in Section 6.2.2), equals the minimal timeout value. Furthermore, combined with other information on the flow (namely arrival curve at source and delay jitter), the RTO can be used to derive the required buffer size (Theorem 6.3). In-line with the operation mode of time-sensitive networks, such a metric, or an upper bound on it, must be computed by the control or management plane before a flow is set up. This differs from the intended use of the metrics in RFC4737, which focus on ex-post measurements. Therefore, we propose a theory to compute tight upper bounds on RTO for flows, given the information that is otherwise available to the time-sensitive network control or management plane (Section 6.4). Such information includes bounds on delay jitter, arrival curve constraints of flows at their sources, and whether a network element is guaranteed to preserve per-flow order or not.

Another metric in RFC4737 is the reordering byte offset (RBO). We show in Theorem 6.3 that it is equal to the required size of the re-sequencing buffer when the network can be assumed to be lossless. Otherwise, if packet losses cannot be ignored, we show that the RBO underestimates the required buffer size, for which we give a formula that involves the RTO. This closes the first issue.

Concerning the third issue, observe that re-sequencing buffers may delay packets until they can be delivered in-order, therefore, they may increase the worst-case delay and the delay jitter. However, we show in Theorem 6.4 that, if a flow is lossless between its source and a re-sequencing buffer, the worst-case delay and the delay jitter are not increased by the re-sequencing buffer (i.e. re-sequencing is for free in terms of delay in the lossless case). In contrast, if the flow may be subject to packet losses on its path from source to the re-sequencing buffer, then the worst-case delay may be increased by an amount up to the timeout value of the re-sequencing buffer, which must be at least as large as the RTO between the source and the input of the re-sequencing buffer. These results are based on a novel input-output characterization of the re-sequencing buffer.

Finally, our theory also allows to evaluate the value of re-sequencing buffers at intermediate points, in addition to the destination, in time-sensitive networks. Regarding the first challenge, our formulas in Section 6.4.2 capture in particular the pattern of RTO amplification by downstream jitter: if a non order-preserving element (typically a switching fabric) has very small RTO but is followed by a per-flow order-preserving element (typically the queuing system on an output port) with large delay jitter, then the concatenation of the two produces a large RTO. This motivates some vendors to perform per-flow re-sequencing after every switching fabric. With respect to the second challenge, we find that such intermediate re-sequencing buffers do not improve the worst-case delay nor delay jitter if the network is lossless; but they do reduce the worst-case delay, delay jitter and RTO at destination in presence of network losses.

To quantify the effect of intermediate re-sequencing buffers, we also need to evaluate how arrival curves of flows are modified by re-sequencing, since such arrival curves are required to compute delay and jitter bounds (Section 6.3.3). We illustrate the application of our theory to two industrial test cases.

The rest of this chapter is organized as follows. The state-of-the-art is presented in Section 6.1. Common assumptions, including a formal description of the RTO and RBO metrics, are given in Section 6.2, together with background results on network calculus in non-FIFO networks, and a notation list. In Section 6.3 we provide a formal input-output characterization of the re-sequencing buffer, which is then used to establish the link between RTO and its required parameters, and to establish its effect on worst-case delay, delay jitter and output arrival curve. In Section 6.4, we show how RTO and RBO can be computed, as required to establish performance guarantees for time-sensitive flows; the method is in two parts: first, we develop formulas for an individual network element, given delay jitter and an arrival curve of the flow; then we develop a calculus to concatenate network elements. In Section 6.5 we apply the results to analyze the performance of intermediate re-sequencing in two industrial case studies. Section 6.6 concludes this chapter. For the reader's convenience, Section 6.7 gives the notation used throughout this chapter. Proofs of theorems and details of computations are in Appendix C.

6.1 Related Work

Kleinrock et al obtain the average re-sequencing delay in [154], assuming Poisson arrival of messages and a number of other simplifying assumptions. A more complete analysis is then performed in [155], where the distribution of the end-to-end response time, including re-sequencing delay, is obtained.

Later studies mainly focus on the statistical measurement of the occurrence of reordering in a communication network; in [156, 48], the authors indicate that the rate of packet reordering is high inside the network. Later, other works focus on the real-time techniques to measure packet reordering [157, 158, 159]. In [157], the authors provide a collection of measurement techniques that can estimate end-to-end reordering rates in TCP connections. In [158], the authors propose and implement an algorithm to measure reordering at a TCP receiver. The authors in [159] provide the probability density function for the amount of reordering of an arbitrary packet, based on received packets.

All the aforementioned works focus on the techniques to capture statistical information on packet reordering inside the network. Few works study the sizing of re-sequencing buffers: [54, 53] provide probability distribution of the re-sequencing buffer size. To the best of our knowledge, there is no prior work that computes the size of re-sequencing buffer and its timeout value in the context of worst-case performance (as required with time-sensitive networks) nor the effect of re-sequencing on worst-case delay and delay jitter.

6.2 Background Information

6.2.1 Network Assumptions

We consider a network that contains a set of nodes, a set of hosts, and a set of links with fixed capacity. Nodes are switches or routers. A node consists of elements that can be order-preserving for the flow of interest (e.g. output port FIFO queues) or non order-preserving (e.g. switching fabric). Every flow follows a fixed path, has a finite lifetime and emits a finite, but arbitrary, number of packets. We consider unicast flows (extension to single-source multicast flows is straightforward) with known bit-level or packet-level arrival curves at their sources (sections 3.2.2 and 4.2.3). A node may also implement a re-sequencing buffer to provide in-order packet delivery for one or several flows of interest. If a flow requires in-order packet delivery and if there is at least one non-order-preserving element on its path, then one re-sequencing buffer is required, and can be placed anywhere after the last non order-preserving element on the path. In some configurations, we will also consider that some additional intermediate re-sequencing buffers are placed inside the network.

Hosts are sources or destinations of flows. Packet sequence numbers are written at the source, starting with number 1 for the first packet sent by the flow. The sequence number is incremented by 1 for every packet of the flow, i.e., sequence numbering is per-packet per-flow. If a packet is lost in the network, with most time-sensitive applications, there is no packet retransmission; instead, the application hides the loss using some application-specific robustness mechanism (see e.g. [160]). If the source happens to retransmit the missing data, the resulting packets are assumed to have a new sequence number (larger than the already sent packets of the same flow).

This section provides a set of general assumptions to present a high-level view of the considered network. Further details, e.g., scheduling policy inside the nodes, are not required for understanding the theory presented in this chapter. In Section 6.5 we describe two case studies; there, we describe the network and flows with all details.

6.2.2 Packet Reordering Metrics

RFC 4737 defines a number of packet reordering metrics, two of which are of interest in the context of time-sensitive networks: the reordering late time offset (RTO) and the reordering byte offset (RBO), which we now formally define. Both metrics are defined for a flow and between an input and an output observation points. When the input observation point is not specified, it is implicitly assumed that it is the source of the flow.

We call packet with index n the nth packet observed at the input observation point, in chronological order. If the input observation point is the source, then packet n is the packet with sequence number n. Let E_n be the time at which packet n is observed at the output observation point. If this packet is lost between the two observation points, we take $E_n = +\infty$. Note that, since some network elements may be non order-preserving, E_n cannot be assumed to be a monotonic sequence. Simultaneous packet observations might be possible in some cases (e.g. if the observation of a packet depends on the realization of a software condition) and the system must use some tie-breaking rule to determine a processing order for packets; if this happens, we assume that we modify the timestamps E_n by some small amounts to reflect the tie-breaking rule i.e. we assume that if $j \neq n$ then $E_j \neq E_n$. If packet n is not lost, i.e. if $E_n < +\infty$, its reordering late time offset is

$$\lambda_n = E_n - \min_{j | j \geq n, E_j \leq E_n} E_j \tag{6.1}$$

i.e. λ_n is the largest amount of time by which a packet with index larger than n arrives earlier than packet n, i.e. the maximum amount of “overtaking” undergone by packet n; if there is no reordering after packet n, then $\lambda_n = 0$. The reordering late time λ_n is undefined if packet n is lost.

The reordering late time offset, RTO, of the flow between the two observation points is $\lambda = \max_{n|E_n<+\infty} \lambda_n$. It follows that, if packet n is not lost, then for any packet index $p \geq n$:

$$E_p \geq E_n - \lambda. \tag{6.2}$$

We always have $\lambda \geq 0$ and it is easy to see that $\lambda = 0$ if and only if the network path between the two observation points preserves the order of packets for this flow.

For the second metric, we need to count misordered bytes; observe that packet n is misordered between the two observation points if there exists some $j > n$ such that $E_j < E_n$. Then, if packet n is not lost, its reordering byte offset is defined by

$$\pi_n = \sum_{j | j > n, E_j < E_n} l_j. \tag{6.3}$$

where l_j is the size, in bytes, of packet j. Thus π_n is the cumulated number of bytes of packets with index larger than n that arrive earlier than packet n; if there is no reordered packet after n, then the sum is empty and $\pi_n = 0$. The reordering byte offset π_n is undefined if packet n is lost. The reordering byte offset, RBO, of the flow between the two observation points is $\pi = \max_{n|E_n<+\infty} \pi_n$.

This definition of RBO is in bytes and not in bits as is often done for buffer and packet lengths in the context of time-sensitive networks; this is to be consistent with the terminology in RFC 4737. Also observe that a similar definition could be given by counting packets rather than bytes, as is done in [52].

6.2.3 Re-sequencing Buffer

A re-sequencing buffer stores the packets of a flow until the packets with smaller sequence numbers arrive; then it delivers them in the increasing order of their sequence numbers. A re-sequencing buffer has two parameters, a size in bytes, B, and a timeout value, T. For any individual packet that is stored in the buffer, a timer is set that expires after T seconds. Then, if a timer for a packet expires, all the stored packets with smaller or equal sequence number are released in-order. Hence, a packet is released if any one of the following conditions holds: 1) all packets with smaller sequence numbers are received, 2) its timer expires or 3) the timer of a received packet with a larger sequence number expires. A detailed description of the re-sequencing buffer algorithm is provided in Appendix C.1.

By construction, the re-sequencing buffer delivers the packets that it does not discard in increasing sequence numbers. Furthermore, a packet is discarded by the re-sequencing buffer either when the buffer is full or when the sequence number of the arriving packet is less than the largest sequence number that was already released. The latter occurs when the timeouts of packets are too early, compared to the lateness of misordered packets. Therefore, to avoid discarding packets, the re-sequencing buffer size and the timeout value should be large enough. In Section 6.3, we analyze how to set the parameters such that these conditions hold.

6.2.4 Assumptions on Arrival Curves

We follow the definition of arrival curve in Section 3.2.2. Without loss of generality [25], the arrival curve α can be assumed to be sub-additive (Eq. (3.6)) and left-continuous. In this chapter we assume that entire packets are observed at the observation point, which imposes that $\alpha^+ \geq L^{\max}$, where $L^{\max}$ is the maximal packet size, otherwise no packet of maximal size can be sent by the flow.

An arrival curve α is “achievable” if, for any sequence of packet sizes between $L^{\min}$ and $L^{\max}$, there is a source with these packet sizes that achieves equality in the arrival curve constraint, or more specifically, if the sequence of packets obtained by packetizing the fluid source $R(t) = \alpha(t)$ satisfies the arrival curve constraint α. Note that the fluid source $R(t) = \alpha(t)$ always satisfies the arrival curve constraint α since we can always assume that α is sub-additive; however packetization may introduce some violations [161]. Any concave arrival curve that satisfies $\alpha(0) = 0$ and $\alpha(0^+) \geq L^{\max}$ is achievable [25, Theorem 1.7.3][161]. For a flow with packets of constant size, any arrival curve whose values are integer multiples of the packet size is achievable as soon as it is sub-additive, left-continuous and satisfies $\alpha(0) = 0$. Leaky-bucket arrival curve, $\alpha(t) = rt + b, t > 0$ and $\alpha(0) = 0$, is always achievable. The staircase arrival curve, $\alpha(t) = b\left\lceil \frac{t}{\tau} \right\rceil$, is achievable if all packets are of size b. If for example, in contrast, $b = 1500$bytes but the packets emitted by the source all have a size equal to 1200bytes, then this arrival curve is not achievable: b should be set to 1200, i.e. α should be replaced by a smaller arrival curve, which is then achievable.

As we saw in Section 4.2.3, instead of counting bytes, constraints can be expressed in number of packets using packet-level arrival curves. The common form of such arrival curves use in IEEE TSN is staircase one defined by $\alpha_{\text{pkt}}(t) = K\left\lceil \frac{t}{\tau} \right\rceil$, with period τ and burst of K packets. Packet-level arrival curves can always be replaced by an integer-valued sub-additive, left-continuous function that vanishes at zero; it is then always achievable.

Last, we also need two technical assumptions. First, we assume that arrival curves are not bounded from above, i.e. $\lim_{t\to+\infty} \alpha(t) = +\infty$. This holds for all arrival curves of interest. Second, we assume that every flow has a maximum and minimum packet size $L^{\max}$ and $L^{\min}$; then the number of bytes observed on any time interval must be an element of $\mathscr{L}$, the set of all possible sums of a finite number of packet sizes. If $L^{\max} \geq 2L^{\min}$, then $\mathscr{L}$ is made of all numbers $\geq L^{\min}$; if $L^{\max} = L^{\min}$ then $\mathscr{L}$ is made of all multiples of $L^{\max} = L^{\min}$. Unless otherwise specified, we assume either of these conditions holds, as otherwise $\mathscr{L}$ is cumbersome and tightness results would become very complex.

6.2.5 Network Calculus Results in Non-FIFO Networks

In a time-sensitive network, the burstiness of a flow may increase at every node, due to multiplexing and random delays. Thus an arrival curve constraint at the source is usually no longer valid inside the network. Analysis of time-sensitive networks uses bounds on the propagation of arrival curves [2]. Such results are based on network calculus theorems that were derived for order-preserving networks[25, Section 1.4.1], but which can be extended to non order-preserving networks, as we show next. Specifically, we will use the following two results, the proofs of which are in appendices C.2.1 and C.2.2.

Lemma 6.1. *Assume a flow has arrival curve α at the input of some system $\mathscr{S}$, which needs not preserve the order of packets of the flow. Assume the delay jitter of the flow through $\mathscr{S}$ is upper bounded by some quantity V. At the output of $\mathscr{S}$, the flow has arrival curve α' given by $\alpha'(t) = \alpha(t+V)$. The same holds, mutatis mutandi, for packet-level arrival curves.*

Lemma 6.2. *Assume a flow has arrival curve α at the input of some system $\mathscr{S}$, which needs not preserve the order of packets of the flow. Assume the worst-case delay of the flow through $\mathscr{S}$ is upper bounded by some quantity U. At any point in time, the amount of data of the flow that is present in $\mathscr{S}$ is upper-bounded by $\alpha(U)$. The same holds, mutatis mutandi, for packet-level arrival curves.*

6.3 Properties of the Re-sequencing Buffer

In this section we first provide a formal input-output characterization of the re-sequencing buffer. Then we use it to analyze the optimal parameter setting, assuming that bounds on RTO and RBO of the flow are known. Last, we characterize the performance effect of a re-sequencing buffer in terms of delay, delay jitter and arrival curve propagation.

6.3.1 Input-output Characterization of the Re-sequencing Buffer

We use the notation in Figure 6.1, where packets of the flow of interest are emitted in sequence by a source. Packet with sequence number n is emitted at time A_n, traverses a non order-preserving network, reaches the re-sequencing buffer at time E_n, from which it is released at time D_n. If the packet is lost by the network, then $E_n = D_n = +\infty$. If the re-sequencing buffer discards packet n, then $D_n = +\infty$. Recall that a packet may be lost in the network or discarded by the re-sequencing buffer. The latter may occur either (1) when the packet arrives after a packet with smaller sequence number was released; this is due to the timeout value T being too small; or (2) if the buffer size B is too small. In this subsection, we assume the buffer is large enough, and in Section 6.3.2 we compute the maximum buffer occupancy, which will give the required buffer size for a given flow. The following theorem characterizes the departure times from the re-sequencing buffer and is the basis from which the results in the rest of this section are derived.

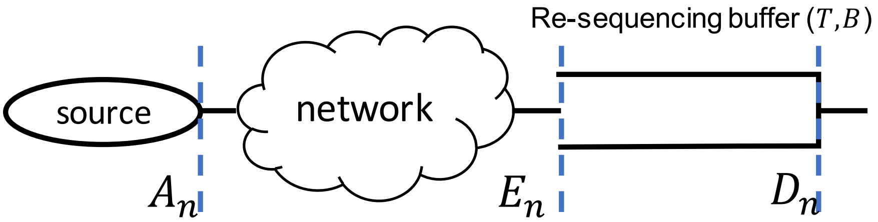


Figure 6.1: Notation used in Section 6.3.

Theorem 6.1. *Consider the re-sequencing buffer described in Section 6.2.3 with timeout value equal to T and with infinite buffer capacity $B = +\infty$. See Figure 6.1 for the notation.*

1. *The packet with sequence number n leaves the re-sequencing buffer at time D_n given by*

$$D_n = \begin{cases} I_n & \text{if } n = 1, \\ \max\{G_n, I_n\} & \text{if } n > 1 \end{cases} \tag{6.4}$$

$$\textit{with } I_n = \begin{cases} +\infty \text{ if } E_n > \min_{j \geq n}\{E_j\} + T, \\ E_n \text{ otherwise} \end{cases} \tag{6.5}$$

$$\textit{and, for } n \geq 2: \quad G_n = \min\left(D_{n-1}, T + \min_{j \geq n}\{E_j\}\right). \tag{6.6}$$

2. *Let λ be the RTO of this flow between the source and the input of the re-sequencing buffer. If $T \geq \lambda$ and packet n is not lost in the network (i.e. $E_n < +\infty$) then it also holds that*

$$D_n = \begin{cases} E_1 & \text{if } n = 1, \\ \max\{G_n, E_n\} & \text{if } n > 1 \end{cases} \tag{6.7}$$

where G_n is defined in (6.6).

3. *If $T \geq \lambda$ and the network is lossless (i.e. $E_n < +\infty$ for all n), then it also holds that $D_n = \max_{k \leq n}\{E_k\}$.*

The proof is in Appendix C.2.3. It is based on induction and uses the description of re-sequencing buffer presented in Section 6.2.3.

6.3.2 Optimal Dimensioning of the Parameters of the Re-sequencing Buffer

Recall that, by construction, the re-sequencing buffer always delivers packets in-order. However, it may do so by discarding late packets. We can now use the previous theorem to derive the minimal values of the timeout T and the size B of the re-sequencing buffer, such that it never discards any packet. We start with the timeout value.

Theorem 6.2. *Consider the re-sequencing buffer described in Section 6.2.3 and Figure 6.1, with timeout value of T and infinite buffer size $B = +\infty$. Let λ be the RTO of the flow of interest between the source and the input of the re-sequencing buffer. The minimum value of T that guarantees that the re-sequencing buffer never discards packets of this flow is $T = \lambda$.*

The proof of Theorem 6.2 is in Appendix C.2.4. It consists in two steps. First, using Theorem 6.1, item 2, we show that, if $T \geq \lambda$, there is no packet discard due to spurious timeout. Second, using Theorem 6.1, item 1, we show that, for any $\lambda > 0$, if $T < \lambda$ we can construct an execution trace with RTO λ such that a packet is discarded due to spurious timeout.

Theorem 6.2 thus establishes the central role of the RTO metric as far as the timeout value is concerned. For the required buffer size, the results are more complex, as shown in the next theorem.

Theorem 6.3. *Consider the re-sequencing buffer described in Section 6.2.3 and Fig. 6.1, with timeout value of T and buffer size B. Let λ, π and V be the RTO, RBO and delay jitter of the flow of interest between the source and the input of the re-sequencing buffer. Assume that $T \geq \lambda$. Also assume that the flow has arrival curve α at its source. The minimal size of the re-sequencing buffer required to avoid buffer overflow is*

1. *$B = \pi$, if the network in Fig. 6.1 is lossless for the flow;*
2. *$B = \alpha(V + T)$, if the network in Fig. 6.1 is not lossless for the flow.*

The proof is in Appendix C.2.5. It consists in four steps. First, assuming the network in Figure 6.1 is lossless for this flow and using Theorem 6.1, we show that the actual buffer content is upper bounded by π, which shows that a buffer of size $B = \pi$ is sufficient. Second, we show that for any $\lambda > 0$ and any valid RBO value π (a valid RBO value is a number that can be decomposed as the sum of an arbitrary number of packet sizes) there always exists one execution trace of a flow with RTO λ and RBO π, that achieves a buffer content equal to

π; therefore the minimal size cannot be less than π. If $\lambda = 0$, the network preserves packet order for this flow and thus $\pi = 0$ as well and the result is clear. This shows item 1. Third, using Lemmas 6.1 and 6.2 in Section 6.2.4, we show that, if the network is not lossless for this flow, the actual buffer content is upper bounded by $\alpha(V+T)$. Fourth, we show that, for any achievable arrival curve α, RTO λ, jitter V and timeout value T, we can construct an execution trace with RTO λ in which the buffer content can become arbitrarily close to $\alpha(V+T)$. This shows item 2.

Remark. It follows from Theorem 6.6 that the bound, π, in item 1, is always less than the bound, $\alpha(V+T)$, in item 2, as expected.

Remark. Loss-free operation is often considered as the normal case in time-sensitive networks, since congestion losses are avoided and transmission losses are very rare; a packet loss might then seen as an exceptional error case, treated by exception-handling routines. If such an assumption can be made, Theorem 6.3 shows that the required buffer content is only dependent on the RBO of the flow.

However, such an interpretation should be taken with care. Indeed, the loss of a single packet before the input to the re-sequencing buffer may delay a number of other packets: the first arriving packet with sequence number larger than the lost packet is delayed at the re-sequencing buffer by T, and, depending on the scenario, following packets may be delayed as well. Thus, the loss of a single packet, even rarely, may impose a delay increase to many more subsequent packets and may lead to the violation of the bound in item 1. Hence, if the bound in item 1 is used for dimensioning the re-sequencing buffer, then the loss of a single packet in the network may cause the loss of many more packets at the re-sequencing buffer due to an insufficient buffer size (since the bound in item 2 is always larger than in item 1). Quantifying this in detail is left for further study.

Remark. The RBO and buffer size B in Theorem 6.3 are expressed in bytes. Obviously, a similar result holds if we count in packets: if the flow is constrained at the source by a packet-level arrival curve α_{pkt}, then the size of the re-sequencing buffer, counted in packets, is upper-bounded by $\alpha_{\text{pkt}}(V+T)$.

6.3.3 Effect of Re-sequencing on Worst-case Delay, Jitter and Arrival Curve

When re-sequencing buffers are used, they may affect packet delay. In this section, we quantify this effect in the sense of worst-case delay and delay jitter, as required in time-sensitive networks.

Theorem 6.4. *Consider a flow as in Figure 6.1, and let λ be the RTO of the flow between the source and the input of the re-sequencing buffer. Assume that the timeout value T of the re-sequencing buffer satisfies $T \geq \lambda$. The worst-case delay and the delay jitter of the flow*

1. *are not increased, if the network is lossless;*
2. *are increased by up to T, if the network is not lossless.*

Formally, with the notation in Figure 6.1, the theorem means that, if the network is lossless (i.e. $E_n < +\infty$ for every n), then

$$\max_n(D_n - A_n) = \max_n(E_n - A_n), \tag{6.8}$$

$$\max_n(D_n - A_n) - \min_n(D_n - A_n) = \max_n(E_n - A_n) - \min_n(E_n - A_n) \tag{6.9}$$

as the former is the worst-case delay and the latter is the delay jitter.

In contrast, if there are some losses in the network, the theorem means that

$$\max_n(D_n - A_n) \le \max_n(E_n - A_n) + T, \tag{6.10}$$

$$\max_n(D_n - A_n) - \min_n(D_n - A_n) \le \max_n(E_n - A_n) - \min_n(E_n - A_n) + T \tag{6.11}$$

The proof is in Appendix C.2.6. It uses the input-output characterization of re-sequencing buffers in Theorem 6.1.

Remark. Item (2) of Theorem 6.4 is tight. Consider a packet n that experiences maximum delay $\delta^{\max}$ while packet $n-1$ is lost in the network. Therefore, packet n should wait in the re-sequencing buffer until its timer expires after T seconds. Then, packet n is delayed by $\delta^{\max} + T$.

Remark. Item (2) quantifies the price of misordering under lossy operation: the re-sequencing buffer, which is caused by the presence of misordering, increases the worst-case delay and the delay jitter of the flow by an amount (T) that is at least equal to the RTO.

Remark. The same remark about loss-free operation holds as in Section 6.3.2. Specifically, the loss of a single packet may impose a delay increase to many more subsequent packets (e.g. to all packets that arrive before timeout). For example, if the flow has packet-level arrival curve α_{pkt} at the source and jitter V between the source and the input of the re-sequencing buffer, it can easily be seen, using the same arguments as in the proof of Theorem 6.3, that the loss of a single packet may cause the delay bound in item 1 to be violated for a number of packets equal to $\alpha_{\text{pkt}}(V+T)$. This stresses again that the result in item 1 should be taken with care, and that the delay bounds in item (2) are more realistic.

We can apply Lemma 6.1 to the previous theorem and quantify the propagation of arrival curves through a re-sequencing buffer:

Corollary 6.1. *Consider a flow as in Figure 6.1; assume that it satisfies the arrival curve α at the source and that the delay jitter between source and input to the re-sequencing buffer is V. Also assume that the timeout value T of the re-sequencing buffer satisfies $T \ge \lambda$, where λ is the RTO of the flow between the source and the input of the re-sequencing buffer. At the output of the re-sequencing buffer, the flow has arrival curve α' defined by*

1. $\alpha'(t) = \alpha(t + V)$*, if the network is lossless;*

2. $\alpha'(t) = \alpha(t + V + T)$*, if the network is not lossless.*

The same applies, mutatis mutandi, to packet-level arrival curves.

Remark. Part (2) of the corollary is tight. This can be shown using the same arguments as in step (4) of the proof of Theorem 6.3. Specifically, the bound is achieved in a scenario where an isolated packet loss occurs, followed by a burst of in-order packets.

6.4 Computing RTO and RBO

In the previous section we saw how to dimension a re-sequencing buffer in the context of time-sensitive networks, assuming that we know the RTO of the flow and, to the extent that lossless metrics are of interest, its RBO. It remains to see how the RTO/RBO, or bounds on them, can be estimated by the control or management plane in order to setup the flow. To this end, we decompose a network path into elements that are either per-flow order-preserving or not. Examples of the former are the IEEE TSN class-based queuing subsystems [63]; examples of the latter are some switching fabrics which use parallel paths to improve throughput [162]. In Section 6.4.1 we give tight RTO and RBO bounds for network elements; in Section 6.4.2 we show how to concatenate them.

6.4.1 RTO and RBO for Network Elements

Theorem 6.5. *Consider a flow that traverses a system, with delay jitter upper-bounded by* V*. Then an upper bound on the RTO of this flow between the input and the output of the system is:*

1. $\left[V - \alpha^{\downarrow}(2L^{\min})\right]^+$*, if the flow has arrival curve* α *at the input of the system;*

2. $\left[V - \alpha^{\downarrow}_{\text{pkt}}(2)\right]^+$*, if the flow has packet level arrival curve* α_{pkt} *at the input of the system.*

The bounds are tight, i.e. for every achievable arrival curve and every value of $L^{\min}$ *and* V *there is a system and an execution trace that attains the bound.*

The proof is in Appendix C.2.7. It consists in two steps. First, based on the definition of jitter in Definition 3.10 and the alternative definitions of arrival curves in Sections 3.2.2 and 4.2.3, we show that the RTO is not larger than the bound in Theorem 6.5. Second, we show that there always exists one execution trace of a flow that its RTO is equal to the bound. Specifically, the bound is achieved in a scenario where the two packets enter a system in a greedy manner and the first one experiences the worse-case delay and the second one experiences the best-case delay.

Remark. If no arrival curve is known for the flow, we can always take $\alpha(t) = +\infty$ for $t > 0$ and then $\alpha^{\downarrow}(x) = 0$; this gives the RTO bound $\lambda = V$, i.e. jitter is a valid RTO bound for any system and any flow.

Remark. If $\alpha^{\downarrow}(2L^{\min}) \geq V$ or $\alpha^{\downarrow}_{\text{pkt}}(2) \geq V$ then the RTO bound given by the theorem is 0, i.e., there is no reordering for this flow. Thus, the theorem captures the cases where the packets sent by the flow are rare and the delay jitter of the non order-preserving system is small, so that reordering is impossible. See the next two examples.

Remark. If the RTO of the flow is larger than zero, according to the theorem $\alpha^{\downarrow}(2L^{\min}) < V$ or $\alpha^{\downarrow}_{\text{pkt}}(2) < V$. Then by [124, Property P7, Section 10.1], we have $\alpha(V) \geq 2L^{\min}$ or $\alpha_{\text{pkt}}(V) \geq 2$. This implies that if a flow has reordering, the input generates at least two packets within the duration of V.

Hereafter, we provide examples on computation of RTO for multi-path connections and common forms of arrival curves, i.e., staircase and leaky bucket.

Example. Consider an interconnection system with K paths. Every path k has a worst-case delay $d_k^{\max}$ and a best-case delay $d_k^{\min}$. The delay jitter of this interconnection system is $V = \max_{k=1\ldots K} d_k^{\max} - \min_{k=1\ldots K} d_k^{\min}$; the RTO for a flow at the output of this interconnection is then given by Theorem 6.5. Non order-preserving switching fabrics fall into this category; here, the delay jitter, and hence the RTO, are typically very small.

Example. Consider a flow that has packet-level arrival curve $\alpha_{\text{pkt}}(t) = K\lceil \frac{t}{\tau} \rceil$, expressing that at most K packets are allowed in any time window of τ seconds. Here we have $\alpha^{\downarrow}_{\text{pkt}}(x) = \tau\lceil \frac{x-K}{K} \rceil, x > 0$ and Theorem 6.5 gives an RTO bound equal to $\lambda = \left[V - \tau\lceil \frac{2-K}{K} \rceil\right]^+$.

Applying this with $K = 1$ gives that, for a flow that generates at most one packet every τ seconds:

1. if $\tau \geq V$, the flow experiences no reordering, i.e., $\lambda = 0$.
2. if $\tau < V$, the flow may experience reordering, and $\lambda = V - \tau$.

Example. Consider a flow that has a leaky bucket arrival curve, i.e., $\alpha(t) = rt + b, t > 0$, with rate r and burst $b \geq L^{\max}$. Then, we have $\alpha^{\downarrow}(x) = \left[\frac{x-b}{r}\right]^+$ and the RTO bound given by Theorem 6.5 is $\lambda = \left[V - \left[\frac{2L^{\min}-b}{r}\right]^+\right]^+$. It follows that:

1. if $b < 2L^{\min}$ and $V \leq \frac{2L^{\min}-b}{r}$, the flow experiences no reordering, i.e., $\lambda = 0$;
2. if $b < 2L^{\min}$ and $V > \frac{2L^{\min}-b}{r}$, the flow may experience reordering, and the RTO is bounded by $\lambda = V - \frac{2L^{\min}-b}{r}$;
3. if $b \geq 2L^{\min}$ the flow may experience reordering, and the RTO is bounded by $\lambda = V$.

The first case requires $L^{\max} < 2L^{\min}$, which we excluded when $L^{\max} \geq 2L^{\min}$, but it may occur when packets are all of the same size l; then reordering is impossible if the delay jitter is $\leq \frac{2l-b}{r}$.

Theorem 6.6. *Consider a flow that traverses a system, with delay jitter upper-bounded by* V *and with RTO upper-bounded by* $\lambda > 0$*. Then a bound on RBO of the flow between the input and the output of this system is:*

1. $\alpha(V) - L^{\min}$*, if the flow has arrival curve* α *at the input of the system and* $\alpha(V) \geq 2L^{\min}$*, and* 0 *if* $\alpha(V) < 2L^{\min}$*;*
2. $L^{\max}\left(\alpha_{\text{pkt}}(V) - 1\right)$*, if the flow has packet-level arrival curve* α_{pkt} *at the input of the system and* $\alpha_{\text{pkt}}(V) \geq 2$*, and* 0 *if* $\alpha_{\text{pkt}}(V) < 2$*.*

The bounds are tight, i.e. for every achievable arrival curve and every value of $L^{\min}$*,* $L^{\max}$*,* V *and* $\lambda > 0$ *there is a system and an execution trace such that the RBO of the flow is arbitrarily close to the bound.*

The proof is in Appendix C.2.8. It consists in two steps. First, based on the definition of jitter in Definition 3.10 and the alternative definitions of arrival curves in Sections 3.2.2 and 4.2.3, we show that the RBO is not larger than the bound in Theorem 6.6. Second, we show that there always exists one execution trace of a flow that its RBO is arbitrary close to the bound. Specifically, it is achieved in a scenario where a sequence of packets enter a system in a greedy manner and the first packet experiences the worst-case delay while the other packets already left the system.

Observe that we must have $\alpha^+(0) \geq L^{\max}$ and $\alpha^+_{\text{pkt}}(0) \geq 1$, therefore the expressions in items (1) and (2) are always non-negative. Also notice that the RBO bounds do not depend on λ but require that $\lambda > 0$; otherwise, namely if $\lambda = 0$, there is no reordering and the RBO is 0. Last, observe that the tightness result implies that the RBO can be extremely large if the arrival curve can also be large. In other words, it is not possible to bound the RBO solely by constraining the delay jitter; for example, a non order-preserving switching fabric can have a very large RBO, limited only by the speed of the input ports, if the flows are not otherwise constrained.

Example. Consider a flow that has a packet-level arrival curve $\alpha_{\text{pkt}}(t) = K\lceil \frac{t}{\tau} \rceil$, expressing that K packets are observed in any time window of τ seconds. An RBO bound for this flow is $\pi = KL^{\max}\lceil \frac{V}{\tau} \rceil - L^{\max}$ except if $K = 1$ and $V \leq \tau$ in which case it is 0.

Example. Consider a flow that has a leaky bucket arrival curve, i.e., $\alpha(t) = rt + b, t > 0$, with rate r and burstiness $b \geq L^{\max}$. Applying Theorem 6.6 for a system with jitter bound V, an RBO bound for this flow is $\pi = rV + b - L^{\min}$.

Remark. If we count reordering offset in packets instead of bytes as in [52], the bound in item (2) should be replaced by $\alpha_{\text{pkt}}(V) - 1$.

6.4.2 Concatenation Results

So far, we are able to compute RTO and RBO of a flow with known arrival curve for any system with known delay jitter. In practice, a flow typically traverses a sequence of network elements, of which some cause packet reordering and the rest preserve order. We are interested to compute RTO and RBO of the flow under such situation. To tackle this problem, a trivial method is to concatenate all the elements as a single system with a delay jitter equal to the sum of delay jitters of each network element; then by applying theorems 6.5 and 6.6, we compute RBO and RTO of the combination. However, we can do better, in two ways. First, as shown in Corollary 6.2, we can ignore all prefix order-preserving elements when computing RTO and all suffix order-preserving elements when computing the RBO of the sequence. Second, we can use extra information about the RTO of every network element, as shown in Theorem 6.7.

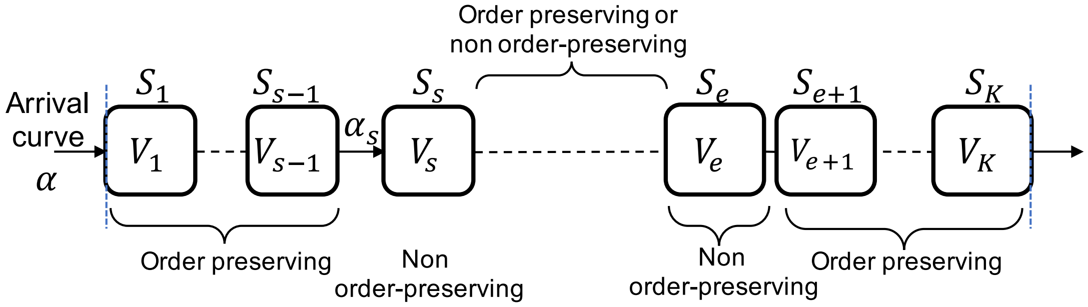


Figure 6.2: Notation for the sequence of network elements used in Section 6.4.2. S_s and S_e are respectively the first and the last non order-preserving elements in the sequence. α and α_s are arrival curves at the entrance of S_1 and S_s respectively.

Corollary 6.2. *Consider the notation in Fig. 6.2. An upper bound on the RTO of the flow between the input and the output of the sequence $S_1, ..., S_K$, denoted as $\Lambda'(K)$, is obtained by applying Theorem 6.5 only to $S_s, ..., S_K$. Specifically:*

$$\Lambda'(K) = \sum_{h=s}^{K} V_h - \alpha_s^{\downarrow}(2L^{\min}), \tag{6.12}$$

where α_s is an arrival curve of the flow at the entrance of S_s.
Moreover, an upper bound on the RBO of the flow between the input and the output of the sequence $S_1, ..., S_K$ is obtained by applying Theorem 6.6 and setting the jitter term as $\sum_{h=1}^{e} V_h$ and the arrival curve term as the one at the entrance of S_1.

The proof of the Corollary is immediate: the RTO bound is justified by the definition of RTO and $S_1, ..., S_{s-1}$ being order preserving. Similarly, since $S_{e+1}, ..., S_K$ are order preserving, they do not affect the RBO of the whole sequence. In (6.12), note that since S_s is non order-preserving, $V_s - \alpha_s^{\downarrow}(2L^{\min})) > 0$, thus $\sum_{h=s}^{K} V_h - \alpha_s^{\downarrow}(2L^{\min}) > 0$.

The bounds in Corollary 6.2 only exploit the information on the jitter of each element. However, as we now show, there are cases where an RTO bound λ_h for each element S_h can also be provided such that $\lambda_h < \lambda'_h$, where λ'_h is obtained by Theorem 6.5 using V_h and an arrival curve at the entrance of S_h. Let us see the following example to see how this extra information can be available. Consider a switch with different internal elements including order-preserving input processing units with jitter V_1, non order-preserving switching fabric with jitter V_2, and order-preserving output ports with jitter V_3. The jitter of the switch is $V = V_1 + V_2 + V_3$ and, a bound on the RTO of a flow with arrival curve α at the entrance of the switch is $\lambda = \left[V_2 + V_3 - \alpha^{\downarrow}(2L^{\min}))\right]^+$ (Corollary 6.2). If only the jitter V is exported by the switch, the RTO bound that can be computed by the control plane is $\lambda' = \left[V - \alpha^{\downarrow}(2L^{\min}))\right]^+$ and $\lambda' > \lambda$ in most cases (i.e whenever $\alpha^{\downarrow}(2L^{\min}) < V$). Therefore, it is desirable that this switch exports both its jitter bound V and its RTO bound λ.

This asks the question of which best RTO bound can be obtained from a concatenation of network elements, for each of which both jitter and RTO bounds are known. The answer is provided by the following theorem.

Theorem 6.7. *Assume that for every network element S_h and for the flow of interest, we know a bound V_h on the delay jitter and a bound λ_h on the RTO between the input and the output of S_h. Let S_s be the first network element in the sequence that has $\lambda_s > 0$. Then the RTO of the flow between the input and the output of the sequence is upper-bounded by*

$$\Lambda(K) = \lambda_s + \sum_{h=s+1}^{K} V_h. \tag{6.13}$$

The bound is tight, i.e., for every pair of sequences V_h, λ_h there exists a system and an execution trace that comes arbitrarily close to the bound.

The proof is in Appendix C.2.9. Obviously, we can assume that $V_s - \alpha_s^{\downarrow}(2L^{\min}) > 0$, since S_s is not order preserving for this flow, and, furthermore, that $\lambda_s \le V_s - \alpha_s^{\downarrow}(2L^{\min})$, since otherwise, by Theorem 6.5, we can replace λ_s by $V_s - \alpha_s^{\downarrow}(2L^{\min})$. It follows that the RTO bound given by Theorem 6.7 is always at least as good as that of Theorem 6.5, and improves on it whenever $\lambda_s < V_s - \alpha_s^{\downarrow}(2L^{\min})$, i.e. whenever the RTO bound exported by S_s does improve over the knowledge of its jitter alone.

Remark. Theorem 6.7 indicates that the only RTO that matters is that of the first non order-preserving element in the sequence (i.e. S_s). For subsequent network elements, it is their delay jitter V_h, not their RTO λ_h, that matters. Observe that RTO is always upper bounded by delay jitter (see remark after Theorem 6.5). Therefore, the RTO of the flow through a sequence of nodes may be larger than the sum of the RTOs of the flow through each individual node. This can be explained as follows: if packet 2 overtakes packet 1 at node S_s by a small amount up to λ_s, it may still happen that the delay of packet 2 through the subsequent nodes is much less than the delay of packet 1, by an amount up to the delay jitter $\sum_{h=s+1}^{K} V_h$. Thus, as destination, we observe that packet 2 overtakes packet 1 by an amount up to $\lambda_s + \sum_{h=s+1}^{K} V_h$. Imagine that all non-order preserving network elements are switching fabrics, for which the RTO is tiny (sub-microseconds) and that other network elements are class-based queuing subsystems, which preserve packet order for every flow, but may have a much larger delay jitter (milliseconds or more). The end-to-end RTO is then of the order of milliseconds, orders of magnitude larger than the amount of reordering introduced by any single network element. This is the pattern of “RTO amplification by downstream jitter”.

Similarly, we can ask whether the information on RTO bound of every element can improve the RBO bound presented in Corollary 6.2. The following theorem shows that the answer is no.

Theorem 6.8. *The RBO bound in Corollary 6.2 is tight, i.e., for every pair of sequences* V_h, λ_h *and every achievable arrival curve there exists a system and an execution trace that comes arbitrarily close to the bound.*

The proof of tightness is similar to the proof of tightness in Theorem 6.6.

6.5 Application to Performance of Intermediate Re-sequencing

In this section we illustrate how the results in the previous sections can be combined in order to evaluate the performance impact of intermediate re-sequencing.

6.5.1 Methodology

For a flow that requires in-order delivery but traverses a network where some elements do not preserve packet order, re-sequencing can be performed at the destination, but it is also possible to insert re-sequencing buffers at intermediate points. For example if one is placed for every flow at the output of every non order-preserving switching fabric, then the network becomes order-preserving and the end-system is relieved from the need to re-sequence. Every choice obviously comes with a different implementation cost; here we do not address such a cost. Instead, we focus on the performance impact, primarily in terms of end-to-end worst-case delay and delay jitter. In this illustration, we consider networks that do not perform flow regulation inside the network.

If losses in the network are rare enough to be ignored for standard operation, the conclusion is straightforward. Indeed, we know from Theorem 6.4 that, under such an assumption, re-sequencing does not increase the worst-case delay and the delay jitter.

In contrast, if lossy operation cannot be ignored, we also know from Theorem 6.4 that re-sequencing adds a penalty to worst-case delay and jitter that is at least equal to the upstream RTO. Furthermore, the pattern of RTO amplification due to downstream jitter may mean that the RTO at the destination is very large, even though RTOs at non order-preserving elements are minuscule. This suggests that intermediate re-sequencing may be beneficial. However, intermediate re-sequencing also introduces a delay penalty and modifies the propagated arrival curves, which must be accounted for.

To fix ideas, we consider a prototypical scenario as in Figure 6.3, where the path of the flow of interest goes through three subnetworks, with jitters $\{V_i\}_{i=1}^3$ and RTOs $\{\lambda_i\}_{i=1}^3$. The first

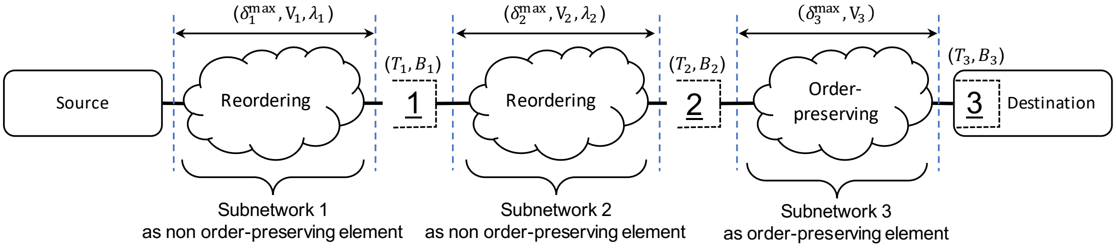

$(\delta_1^{max}, V_1, \lambda_1)$
$(\delta_2^{max}, V_2, \lambda_2)$
(δ_3^{max}, V_3)
Source
Reordering
(T_1, B_1)
1
Reordering
(T_2, B_2)
2
Order-preserving
(T_3, B_3)
3
Destination
Subnetwork 1
as non order-preserving element
Subnetwork 2
as non order-preserving element
Subnetwork 3
as order-preserving element

Figure 6.3: A prototypical scenario used to analyze the performance impact of intermediate re-sequencing. Potential placements of re-sequencing buffers are at points 1, 2, or 3.

subnetwork may for example represent the source output queuing, a transmission link and the switching fabric of the next node. The second may represent the output queuing that follows this switching fabric, plus transmission links and the switching fabric of the following node. The third may represent the output queuing that follows the second switching fabric, plus transmission links to the final destination. By Theorem 6.7, we would have $\lambda_1 << V_1$ and $\lambda_2 << V_2$. We consider the following possible placements of re-sequencing buffers:

- Only at 3 (at destination end-system).
- Only at 2 (at destination edge-switch). This is the case where the last edge-switch performs re-sequencing on behalf of the destination, just after the last non order-preserving element.
- At 1 and 2 (at every switch). This occurs when the network wants to guarantee that all switches preserve per-flow order, as some vendors do.
- At 1 and 3 (at the first switch and the destination end-system). This occurs when the network wants to guarantee that all switches except the destination edge-switch, preserve per-flow order. Then the destination performs re-sequencing with smaller timeout value.

We take as baseline the case where no re-sequencing is applied and compute the increase in worst-case delay and delay jitter with respect to the baseline, for each of the placements. We give the details for delay jitter, the computations are similar for the worst-case delay. We assume the timeout values are optimal, as given by Theorem 6.5. The delay jitter of the baseline is $V_1 + V_2 + V_3$.

For the first placement (only at 3), the re-sequencing buffer at destination increases the delay jitter in the lossy case by T_3, the timeout value of the re-sequencing buffer at point 3, which is equal to the RTO between the source and point 3. By Theorem 6.7, it is equal to $\lambda_1 + V_2 + V_3$. By Theorem 6.4, the increase on jitter is also $\lambda_1 + V_2 + V_3$.

For the last placement (at 1 and 3), the re-sequencing buffer at 1 modifies the arrival curve of the flow. This affects, in general, the downstream worst-case delay and jitter. We call ΔV_2

Table 6.1: Re-sequencing buffer optimal timeout value and increase on end-to-end jitter and delay upper bound (with respect to the baseline with no re-sequencing buffer) for the four placement strategies in Section 6.5.1. T_i is the timeout of re-sequencing buffer placed at point i. We see that placing re-sequencing buffers at 1 and 2 provides better end-to-end delay and jitter comparing to the placement at 1 and 3.

Re-sequencing	Timeout			Increase to end-to-end jitter and delay with respect to no re-sequencing buffer at all.	
	T_1	T_2	T_3		
	Lossless and lossy			Lossless	Lossy
Only at 3	-	-	$\lambda_1 + V_2 + V_3$	0	$\lambda_1 + V_2 + V_3$
Only at 2	-	$\lambda_1 + V_2$	-	0	$\lambda_1 + V_2 + \Delta V_3$
At 1 and 2	λ_1	λ_2	-	0	$\lambda_1 + \lambda_2 + \Delta V_2 + \Delta V_3$
At 1 and 3	λ_1	-	$\lambda_2 + V_3 (+\Delta V_3$ for lossy$)$	0	$\lambda_1 + \lambda_2 + V_3 + \Delta V_2 + 2\Delta V_3$

and ΔV_3 the increase on delay jitter at subnetworks 2 and 3 with respect to the baseline. In the following subsection we estimate these increases numerically on two industrial cases. The re-sequencing buffer at 1 has timeout $T_1 = \lambda_1$, given by the RTO of subnetwork 1. For the RTO at point 3, observe that the subnetwork 1 combined with the re-sequencing buffer at point 1 is an order-preserving element; this causes the RTO at point 3 to be independent of subnetwork 1 (Theorem 6.7). We obtain the timeout value $T_3 = \lambda_2 + V_3 + \Delta V_3$. Finally, the end-to-end delay jitter is increased by T_1 at point 1 and T_3 at point 3, thus it is equal to $V_1 + T_1 + V_2 + \Delta V_2 + V_3 + \Delta V_3 + T_3$, which gives an increase with respect to the baseline equal to $\lambda_1 + \lambda_2 + V_3 + \Delta V_2 + 2\Delta V_3$.

The reasoning is similar for the two other placements. The results are given in Table 6.1. A similar line of reasoning can be used to compute the required sizes of re-sequencing buffers.

We observe the following. In this scenario, we expect λ_i to be much smaller than V_i, so if ΔV_i is small, it is beneficial to place an intermediate re-sequencing buffer, and it is also beneficial to place one at the edge-node rather than at the destination. This is because intermediate re-sequencing reduces the downstream RTO and avoids the RTO amplification pattern. However, if ΔV_i is large this benefit may be lost due to the burstiness increase caused by re-sequencing under lossy operation. In the numerical examples of the next sections, we find that, except in one case, the former effect is largely dominant. Also note that if per-flow regulation were performed at every hop, the latter effect would disappear and intermediate re-sequencing would reduce the worst-case delay and jitter under lossy operation.

6.5.2 Case Study 1: Automotive Network

We apply the methodology in Section 6.5.1 to the double star automotive network of [3] depicted in Fig. 6.4. To obtain the re-sequencing buffer size and timeout, as well as the jitter and delay upper bounds, we used FP-TFA [133, 132] (details of computations are in Appendix C.3).

The network consists of two switches and eight hosts and the rates of the links are $c = 1$ Gbps.

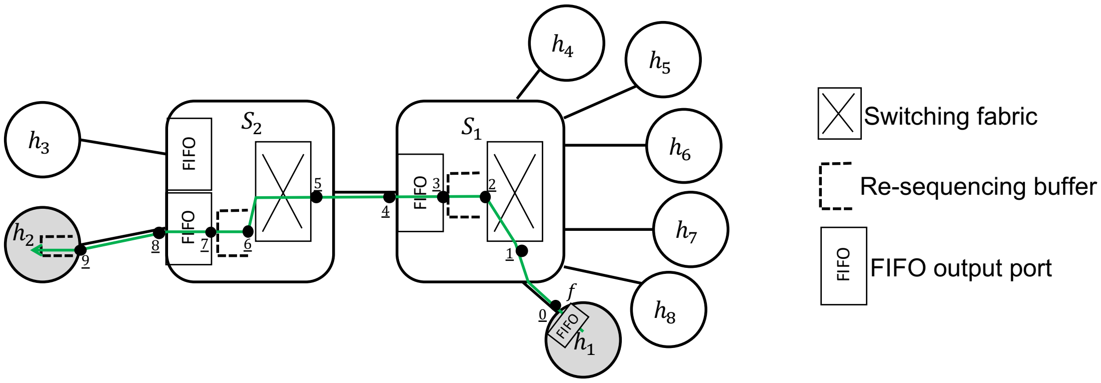


Figure 6.4: Double star automotive network [3].

Table 6.2: Bounds on end-to-end jitter and worst-case delay for case study 1 with the four placement strategies in Section 6.5.1, under both lossless and lossy network conditions, followed by timeout value and size of the re-sequencing buffers.

Re-sequencing buffers placement	Lossless		Lossy	
	Delay (μs)	Jitter (μs)	Delay (μs)	Jitter (μs)
Only at h_2	95.22	92.69	124.72	122.19
Only at S_2	95.22	92.69	127.22	124.69
At S_1 and h_2	95.22	92.69	111.72	109.19
At S_1 and S_2	95.22	92.69	99.22	96.69

Re-sequencing buffers placement	Lossless and Lossy			Lossless			Lossy		
	Timeout T (μs)			Size B (bytes)			Size B (bytes)		
	S_1	S_2	h_2	S_2	S_2	h_2	S_1	S_2	h_2
Only at h_2	–	–	29.49	–	–	6336	–	–	6400
Only at S_2	–	15.99	–	–	6336	–	–	6400	–
At S_1 and h_2	0.98	–	14.49	6336	–	6336	6400	–	6400
At S_1 and S_2	0.98	0.98	–	6336	6336	–	6400	6400	–

The output ports are FIFO and the scheduling mechanism is non-preemptive strict priority. The output ports of h_1, S_1 and S_2 offer the same service curve $\beta(t) = 125e6[t - 12\mu]^+$ bytes to the highest priority queue. In each switch, the switching fabric is implemented in parallel stages, i.e., reordering of packets may occur. The delay of switching fabrics is between $0.5\mu s$ to $2\mu s$ [163].

According to [3], the traffic is made of various flows with different priorities. In our example, we focus on ControlData flow as the only highest priority flow; it is shown as flow f in Fig. 6.4 and the path is taken from [3]; the flow is initiated by Control Data Unit (h_1) and destined to Control Unit (h_2) in the network. The source arrival curve for flow f is leaky bucket with rate 6400 bytes per second and burstiness 6400 bytes. All packets are the same size and equal to 64 bytes.

We implemented the four placement strategies in Section 6.5.1. The results are in Table 6.2. We see that re-sequencing at every switch fabric significantly reduces delay and jitter bounds. In contrast, re-sequencing at edge node (S_2 only) is not beneficial: this is an instance where the burstiness increase due to re-sequencing does have an impact. We also see that the required size of the re-sequencing buffer is independent of the placement strategy.

6.5.3 Case study 2: Orion network

We now consider the Orion crew exploration vehicle network, as described in [164] and depicted in Fig. 6.5. For the delay and jitter analysis, we used FP-TFA [133, 132] as there are cyclic dependencies in the placement of flows. The output ports in the hosts and switches are connected to the links with a rate of 1 Gbps. The output ports use the non-preemptive TSN scheduler with CBSs with per-class queuing as in Chapter 5; from highest to lowest priority, the classes are CDT, A, B, and Best Effort). The CBSs are used separately for classes A and B. The CBS parameters *idleslopes* are set to 50% and 25% of the link rate respectively for classes A and B [103]. In each switch, the switching fabric is implemented in parallel stages, i.e., reordering of packets may occur. The delay of switching fabrics is between $0.5\mu s$ to $2\mu s$ [163]. The CDT traffic has a leaky bucket arrival curve with rate 6.4 kilobytes per second and burst 64 bytes. The maximum packet length of classes B and BE is 1500 bytes. We focus on class A. Using Theorem 5.2, a rate-latency service curve offered to class A is $\beta(t) = 62.49e6[t - t_0]^+$ bytes with $t_0 = 12.5\mu$s.

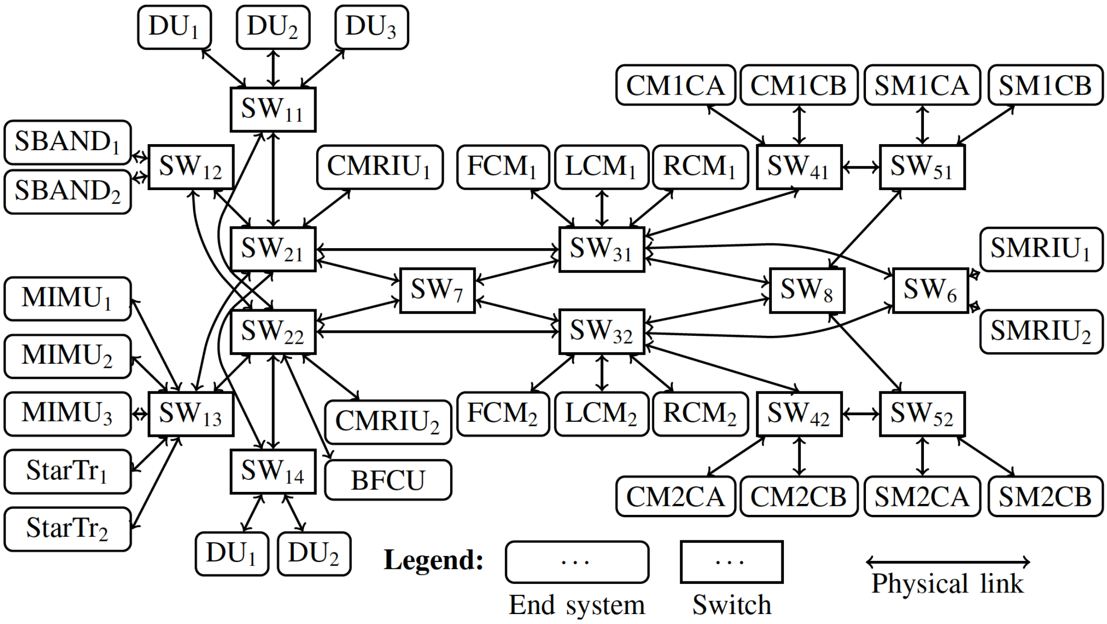


Figure 6.5: The Orion crew exploration vehicle network.

Class A contains 30 flows with constant packet size 147 bytes, which transmit 3 packets every 8 ms. Among these flows, 10 require in-order packet delivery. The flows traverse between 2 to 7 hops. We apply two placement strategies for re-sequencing buffers: at destinations only, and at every switch (immediately after the switching fabrics).

Fig. 6.6 shows the end-to-end delay and jitter bounds for the flows with in-order delivery requirement, for both strategies under lossy condition. The figure also shows the delay and jitter under lossless conditions, which are the same for both strategies and, as we know from Theorem 6.4, are also equal to the values when there is no re-sequencing buffer. First, we

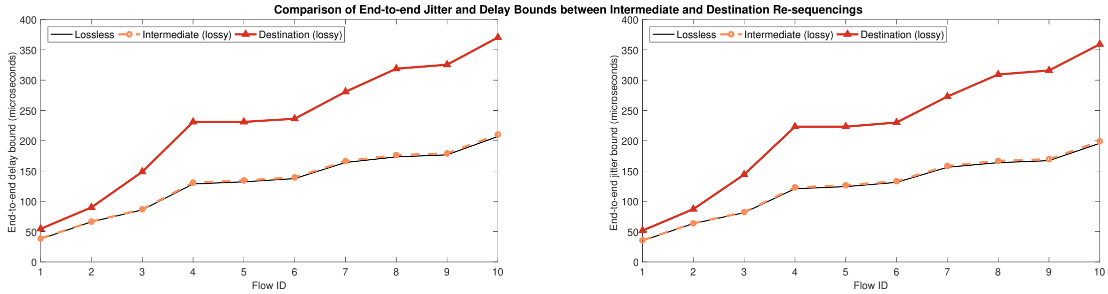


Figure 6.6: The end-to-end delay bounds (left) and jitter bounds (right) for the flows with in-order delivery requirement for the two strategies, i.e., placing re-sequencing buffers at the destinations or at every switch. In the lossless condition, the delay and jitter bounds do not depend on the strategy.

see that if re-sequencing is at destinations only, the effect on delay and jitter under lossy conditions is large: for more than half of the flows, the re-sequencing buffer doubles the delay and jitter, the increase being of the order of 100μs. This occurs even though the amount of reordering late time offset at every switching fabric is minuscule: every flow has at most 7 hops and the switching fabric re-orders packet by at most 324 ns at every hop (by Theorem 6.5 this is less than the jitter of the switching fabric). This illustrates the pattern of RTO amplification by downstream jitter. Second, we see that if re-sequencing is performed at every switch, the increase in delay and jitter under lossy conditions is negligible, as expected from Section 6.5.1, because such a strategy prevents amplification of RTO.

We also find that the size of re-sequencing buffers are the same for the two strategies; this implies that intermediate re-sequencing does not provide any benefit in terms of buffer size. It is equal to two packet size, i.e., 294 bytes, under lossless network condition; and it is equal to three packet size, i.e., 441 bytes, under lossy network conditions.

6.6 Conclusion

We have developed a theory of packet reordering in the context of time-sensitive networks, i.e. in networks where worst-cases are more relevant than averages. We showed that, if the network can safely be assumed lossless, re-sequencing does not modify worst-case delay nor delay jitter. In contrast, if performance under lossy operation is relevant, then re-sequencing comes with a penalty on delay equal at least to the RTO of the flow being re-sequenced. We showed that the RTO may be very large even though the RTO of every individual non order-preserving element is very small, due to amplification by downstream jitter. We provided a calculus to capture the RTO and RBO of a flow, given its arrival curve and simple properties of the network elements that are on its path. We applied the theory to evaluate the performance of re-sequencing strategies in industrial networks without flow regulation.

6.7 Notation

Packet n is the packet with sequence number n if the input point of observation is the source of the flow, otherwise it is the nth packet in chronological order at the input of the system of interest.

Term	Description
A_n	The time at which packet n is released by its source.
B	The size of re-sequencing buffer.
D_n	The departure time of packet n from a re-sequencing buffer.
E_n	The exit time of packet n from a non order-preserving or order-preserving element.
$L^{\max}$	The maximum packet length of a flow, in bytes.
$L^{\min}$	The minimum packet length of a flow, in bytes.
l_n	The length of a packet n, in bytes.
T	The timeout value of a re-sequencing buffer.
λ	The reordering late time offset (RTO) of a flow.
π	The reordering byte offset (RBO) of a flow.
$w^{\downarrow}$	The lower pseudo-inverse of function w (Section 3.1.1).
w^{+}	The right limit of function w (Section 3.1.3).
$\lceil x \rceil$	The ceiling of x
$[x]^{+}$	$\max(0, x)$

7 Analysis of Dampers in Time-Sensitive Networks with Non-ideal Clocks

Patience is bitter, but its fruit is sweet.
— Jean-Jacques Rousseau

As mentioned in Chapter 1, the emergence of applications with low jitter requirement in large-scale time-sensitive networks, such as industrial Internet of Things [8] and electricity distribution[9], questions the performance of existing queuing and shaping mechanisms such as credit-based shaper, IEEE 802.1Qch Cyclic Queuing and Forwarding [46], and deficit round robin [1]. This issue can be addressed with dampers, which are asynchronous mechanisms to reduce delay jitter in time-sensitive networks [42, 43, 44].

A damper delays every time-sensitive packet by an amount written in a packet header field, called damper header, which carries an estimate of the earliness of this packet with respect to a known delay upper-bound of upstream systems. This ideally leads to zero jitter; in practice, there is still some small residual jitter, due to errors in acquiring timestamps and in computing and implementing delays. As a positive side effect, dampers create packet timings that are almost the same as at the source, with small errors due to residual jitter, and thus cancel most of the burstiness increase imposed by the network. [64, Lemma 1]. The residual burstiness increase that remains when dampers are used is not influenced by the burstiness of cross-traffic. Thus, dampers solve the burstiness cascade issue [165] described in Chapter 1: individual flows that share a resource dedicated to a class may see their burstiness increase, which may in turn increase the burstiness of other downstream flows. Furthermore, dampers are stateless, unlike some TSN regulation mechanisms, e.g., Asynchronous Traffic Shaping [41]. Solving the burstiness cascade in a stateless manner makes the dampers of interest for large-scale time-sensitive networks.

Several implementations of dampers have been proposed; the older ones are associated with specific schedulers such as earliest-deadline-first [42, 44] and static priority [43]; the recent implementations can coexist with any scheduling mechanism [166, 167, 168]. Some of these implementations enforce dampers to behave in a FIFO manner [44, 168, 166] and some do not

[43, 167]. Analysis of damper is crucial to provide guarantees for applications in the context of time-sensitive networks. In the existing works, [167, 168] did not provide any analysis; others analyze only their implementation and under limited assumptions on the network settings. Also, existing analyses assume that the network operates with one ideal clock; in practice, this assumption does not hold and may have non-negligible side effects. Recently, the effect of non-ideal clocks on regulators was analyzed and a clock model was proposed in the context of time-sensitive networks [45], which we use in this chapter.

We first present a taxonomy of dampers that classifies the existing implementations into dampers with or without FIFO constraint. Then, under general network configuration with non-ideal clocks, we provide formulas to compute tight delay and jitter bounds for dampers without FIFO constraint (Theorems 7.1 and 7.2); we see that the impact of non-ideal clocks can be non-negligible in cases with low jitter requirements. As a result of this analysis, we derive conditions in which clock synchronization throughout a network does not affect the performance of dampers. Moreover, we capture the propagation of arrival curve at the output of dampers and see how this can solve the burstiness cascade issue. Next, we show that existing implementations of dampers without FIFO constraints may cause undesired packet reordering due to clock non-idealities, even in synchronized networks. This problem is avoided with dampers that enforce FIFO constraints; however, the effect on their timing properties was not analyzed in the literature and we bridge this gap in this chapter. We model two classes of dampers with FIFO constraint: re-sequencing dampers and head-of-line dampers. For the former class, we show that when all network elements are FIFO, the delay and jitter bounds are not affected by the re-sequencing operation (Theorem 7.3). For the latter class, there is a small penalty due to head-of-line queuing, which we quantify exactly (Theorem 7.4). In contrast, if some network elements are non FIFO, the jitter bounds for dampers with FIFO constraint can be considerably larger (Theorems 7.5 and 7.6). We finally evaluate our results in an industrial case-study.

The rest of this chapter is as follows. Section 7.1 presents the state-of-the-art. Section 7.2 describes the system model, terminology, clock model and all assumptions. Section 7.3 presents a taxonomy of the existing dampers. The analysis of dampers without FIFO constraint is presented in Section 7.4. Packet reordering scenarios due to non-ideal clocks are presented in Section 7.5. The analysis of dampers with FIFO constraint is given in Section 7.6. Section 7.7 provides a numerical evaluation for an industrial case-study and Section 7.8 concludes this chapter. For the reader's convenience, Section 7.9 gives the notation used throughout this chapter. Appendix D gives the proofs.

7.1 Related Works

The concept of dampers was introduced by Verma et. al [42], under the name *delay-jitter regulator*, in combination with earliest-deadline-first (EDF) scheduling. In this scheme, a per-flow regulator is placed at every node to delay a packet as much as its earliness in the previous node; the earliness is the time difference between the delay that a packet was supposed to experience and the actual delay that is measured by time-stamping. Later, Zhang et. al. [43] proposed rate-control static priority (RCSP) scheduling to avoid coupling of delay and bandwidth allocation in the EDF schedulers mentioned in [42]. We describe RCSP in Section 7.3.

The term *damper* was first used by Rene Cruz [44] as a conceptual network element that slows down the traffic passing through it. In [44], dampers are used in relationship with service-curve earliest deadline (SCED) scheduling to avoid extra queuing as was proposed by [43]. With this scheme, called SCED+, a flow traverses a few virtual paths (each is a sequence of switches) with guaranteed service curves and damper curves. Then, at the entrance of each virtual path, for every packet of the flow and every switch in the virtual path, initial and terminal eligibility times are computed using the service and damper curves; a packet is released from a switch within its initial and terminal eligibility times.

Recently, a few implementations of damper are proposed that can be used in combination with any scheduling mechanism. Grigorjew et. al. [166] implement damper as a shaper in relation with IEEE 802.1 Qcr Asynchronous Traffic Shaping [41]; we refer to their scheme as jitter-control ATS. It is assumed in [166] that the input flows are constrained by leaky-bucket arrival curve and all the elements inside the network, including the switching fabrics, output port queues and the ATS, are FIFO for the packets that share the same queues inside ATS. Rotated gate-control-queues (RGCQ) [167] is an implementation of a damper integrated with the queuing system of a scheduler. Flow-order preserving latency-equalizer (FOPLEQ) [168] is an extension of RGCQ to preserve the per-flow order of the packets according to its entrance to FOPLEQ. Section 7.3 describes the details of these implementations. These previous works do not provide delay analysis or do it in restricted settings. In particular, clock non-idealities are ignored. In [45] clock non-idealities are modelled in the context of time-sensitive networks and the impact on timing analyses is explained in detail. In this chapter, we apply this clock model to networks with dampers of various kinds.

7.2 System Model

We consider a network that contains a set of switches or routers, hosts and links with fixed capacity. Every flow follows a fixed path, has a finite lifetime and emits a finite, but arbitrary, number of packets. We consider unicast flows with known arrival curves at their sources (i.e. there are known bounds on the number of bits or packets that can be emitted by a flow within any period of time).

7.2.1 Terminology

We call *jitter-compensated system* (JCS) any delay element or aggregate of delay elements with known delay and jitter bounds, for which we want to compensate jitter by means of dampers.

This is typically the queuing system on the output port of a switch or router used in time-sensitive networks. It can also be a switching fabric or an input port processing unit, or even a larger system. For time-sensitive flows, a JCS should be able to time stamp packet arrivals and departures using the available local times. It should also increment the damper header field in every time-sensitive packet (if one is present) by an amount equal to an estimate of the earliness of this packet with respect to the known delay upper-bound at the JCS for the class of traffic that this packet belongs to. If no damper header is present, it inserts one, with a value equal to the estimated earliness.[1] The operation of the damper header update (DHU) unit is described in Section 7.3.3. When a time-sensitive flow crosses a JCS, for actual jitter removal to occur, there must be a downstream damper on the path of the flow. For example, if the JCS is a switch output port, the next downstream damper is typically located on the output port of the next downstream switch.

It is generally not possible, or required, to remove delay jitter in all network elements, because time stamping and DHU come with a cost. Therefore, it is required, for our timing analysis, to consider what we call *bounded-delay systems* (BDSs), defined as any delay element or aggregate of delay elements with known delay and jitter bounds, and for which we do not compensate jitter. Constant delay elements (e.g. an output link propagation delay), variable delay elements with very low jitter (e.g., very high speed backbone network) and other delay elements without DHU unit are examples of BDSs.

A *damper* is a system that delays every time-sensitive packet, using its local clock, for a duration approximately equal to the damper header (if any, else the damper does not delay the packet). Such a damper header was inserted/updated in the upstream JCSs between this damper and the previous upstream damper or the source of the flow. The damper also resets the damper header, so that the next downstream damper will see only the earliness accumulated downstream of this damper. Designing a stand-alone damper is a challenge, because such a damper may need to release a large number of packets instantly or within a very short time, which might not be feasible. This is why damper implementations are often associated with queuing systems; then, the time at which a damper releases a packet is simply the time at which the packet becomes visible to the queuing system. We classify and model existing designs of dampers in Section 7.3.

Example 1. Fig. 7.1 shows an example flow path within a local-area time-sensitive network where we want to compensate the jitter imposed by the output queuing systems and switching fabrics by means of dampers. A DHU is placed after every switching fabric and every queuing system. For example, the first DHU, at the queuing system of the source node, writes into the damper header the earliness of the packet at the queuing system; the second DHU, at the first switch, increments the damper header by the earliness at the switching fabric. The multiple DHUs allow to accumulate earliness with non-synchronized local clocks, as packet arrival

---

[1] We choose this method of carrying earliness in packet headers for ease of presentation. Another method consists in each JCS inserting a separate damper header: a packet then has as many damper headers as JCSs between dampers, and the earliness to be compensated at a damper is the sum of all these values. The discussion of such methods is out of the scope of this chapter, as it does not affect the timing analysis presented here.

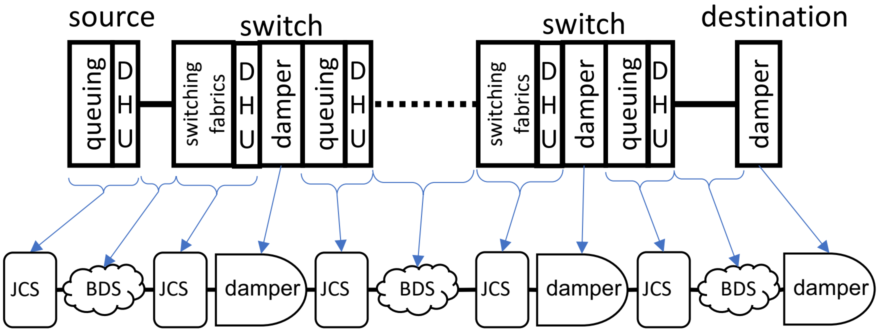


Figure 7.1: A local-area time-sensitive network example.

and departure times are measured locally. A damper is placed before each queuing system to remove the delay jitter imposed by the upstream switching fabric and queuing system. For example, the damper in the first switch compensates the jitter imposed by the queuing system of the source and the switching fabric of the first switch. Note that the propagation delay is constant and seen as a BDS. Here, the different clocks need not be synchronized.

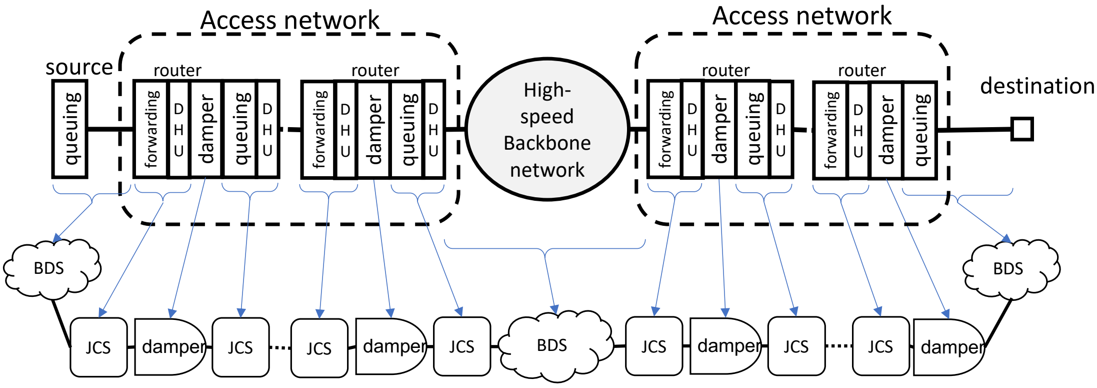


Figure 7.2: A large-scale time-sensitive network example with no jitter removal at backbone network.

Example 2. Fig. 7.2 shows an example flow path within a large-scale time-sensitive network. Assume that the backbone network has relatively low delay (because of high-speed links, e.g. 100 Gbps or more, worst-case delays tend to shrink [169]) and then the main source of jitter is the access network. For a given class of traffic, we want to remove the jitter imposed by the access network, in particular the forwarding plane and output queuing of each access router (each is treated as a JCS); therefore, each of these should have a DHU and a damper upstream of the output queuing system. In this example, the backbone network is modelled as a BDS; also, the source is unaware of any downstream damper and does not have a DHU and hence treated as a BDS. The jitter imposed by the access network is removed, but not the jitter caused by the backbone. The different clocks need not be synchronized and the backbone nodes are unmodified.

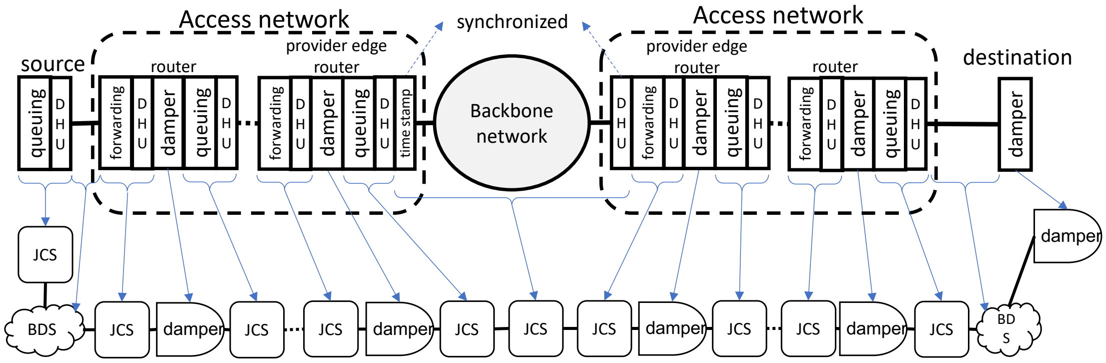


Figure 7.3: A large-scale time-sensitive network example with jitter removal at backbone network.

Example 3. We continue in Fig. 7.3 with the previous example but assume now that, for some class of traffic with very low jitter requirement, the jitter induced by the backbone should be compensated. In such a case, we need to treat the backbone network as a JCS, i.e., we need to time stamp the arrival of each time-sensitive packet to the backbone and modify their damper header at the departure from the backbone. This can be done as in Fig. 7.3, where, at the upstream provider edge (PE) router, a time stamping unit should be added that inserts a field in the packet header equal to the departure time of each time-sensitive packet from the PE router in its local time (this operation can be done within the upstream DHU unit to avoid placement of a time-stamping unit); then at the egress of the downstream PE router, a DHU is placed that reads the departure time of the packet from packet header, removes it from packet header, computes earliness with its local clock and finally modifies the damper header. In this case, differently from previous examples, the time stamping and DHU are performed with different clocks; therefore, the PE routers should be time-synchronized, as otherwise the computation of earliness is impossible (time synchronization is never absolute and, in sections 7.2.2 and 7.3.3, we analyze how to account for clock non-idealities). The jitter induced by the backbone network is compensated in the damper placed in the downstream PE router and hence removed. The PE routers must be time-synchronized (in provider networks, they typically are); backbone nodes are unmodified but time-sensitive packets carry an additional header for timestamps.

7.2.2 Assumptions on the Clocks

We call $\mathscr{H}_{\text{TAI}}$ the perfect clock, i.e. the international atomic time (TAI[2]). In practice, the local clock of a system deviates from the perfect clock [45]. Typically the JCSs upstream of a damper operate with different clocks than the damper itself, and this can affect the performance of the damper as we see in Section 7.4. In time-sensitive networks, clocks can be synchronized or non-synchronized. Non-synchronized clocks are independently configured

---

[2] Temps Atomique International

and do not interact with each other; this corresponds to the *free-running* mode in [170, Section 4.4.1]. When clocks are synchronized, using methods like Network Time Protocol (NTP) [171], Precision Time Protocol (PTP) [172], WhiteRabbit [173], Global Positioning System (GPS) [174], the occurrence of an event, when measured with different clocks, is bounded by the *time error bound* (~ 1μs or less in PTP, WhiteRabbit, and GPS; ~ 100 ms in NTP).

We follow the clock model in [45], which applies to time-sensitive networks. Consider a clock $\mathcal{H}_i$ that is either synchronized with time error bound ω, or not synchronized (in which case we set $\omega = +\infty$). Let $d^{\mathcal{H}_i}$ [resp. $d^{\mathcal{H}_{\mathrm{TAI}}}$] be a delay measurement done with clock $\mathcal{H}_i$ [resp. in TAI], then [45]:

$$\begin{aligned} d^{\mathcal{H}_{\mathrm{TAI}}} - d^{\mathcal{H}_i} &\leq \min\left((\rho-1)d^{\mathcal{H}_i} + \eta, 2\omega\right), \\ d^{\mathcal{H}_{\mathrm{TAI}}} - d^{\mathcal{H}_i} &\geq -\min\left(\left(1-\frac{1}{\rho}\right)d^{\mathcal{H}_i} + \frac{\eta}{\rho}, 2\omega\right), \end{aligned} \tag{7.1}$$

where ρ is the *stability bound* and η the *timing-jitter bound* of the clock $\mathcal{H}_i$. Note that this set of bounds is symmetric, i.e. we can exchange the roles of $\mathcal{H}_i$ and $\mathcal{H}_{\mathrm{TAI}}$ in (7.1). We assume that the parameters ω, ρ, η are valid for all clocks in the network, i.e. we consider network-wide time-error, stability and time-jitter bounds. When a flow has $\alpha^{\mathcal{H}_i}$ as arrival curve with clock $\mathcal{H}_i$, then an arrival curve in TAI is [45]:

$$\alpha^{\mathcal{H}_{\mathrm{TAI}}} : t \to \alpha^{\mathcal{H}_i}\left(\min\{\rho t + \eta, t + 2\omega\}\right). \tag{7.2}$$

In a TSN network, $\rho - 1 = 10^{-4}$ [175, Annex B.1.1] and $\eta = 2$ns [175, Annex B.1.3.1]]; if the network is synchronized with gPTP (generic PTP) then $\omega = 1\mu$s [175, Section B.3] and if it is not synchronized then $\omega = +\infty$.

7.3 Taxonomy of Dampers

As mentioned earlier, designing a damper is a challenge and there exist very different implementations. In this section we classify such implementations in a manner that will be useful for our timing analysis. In the rest of this chapter, we call “eligibility time" the time at which a damper releases a packet, as in most implementations the packet is not actually moved, but simply made visible to the next processing element.

7.3.1 Dampers without FIFO constraint

An *ideal* damper delays a packet by exactly the amount required by the damper header. Consider a packet n with damper header H_n that arrives at local time Q_n to a damper. The theoretical eligibility time $\tilde{E}_n$ for the packet is:

$$\tilde{E}_n = Q_n + H_n, \tag{7.3}$$

and the ideal damper releases the packet at time $\tilde{E}_n$. Jitter-control EDF [42] is an ideal damper, used in combination with an EDF scheduler.

Many other implementations of dampers use some tolerance for the packet release times, due to the difficulty of implementing exact timings. We call damper *with tolerances* $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$ a damper such that the actual eligibility time E_n, of packet n, in local time, satisfies:

$$\tilde{E}_n - \Delta^{\mathrm{L}} \le E_n \le \tilde{E}_n + \Delta^{\mathrm{U}}. \tag{7.4}$$

The tolerances can vary from hundreds of nanosecond to a few microsecond based on implementation. Hereafter, we study two instances of dampers with tolerances $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$, namely, RCSP [43] and RGCQ [167].

RCSP is an implementation of a damper in relation with static-priority scheduler; each queue of the scheduler is implemented as a linked list and the damper is implemented as a set of linked lists and a calendar queue [176]. A calendar queue contains a clock and a calendar where each calendar entry points to an array of linked lists (each for one priority queue). The clock ticks at every fix interval Δ. On each clock tick, the linked list that the current clock time of the calendar points is appended to the corresponding priority queue of the scheduler. Whenever a packet n arrives, its theoretical eligibility time is computed based on (7.3); then the actual eligibility time of the packet, $E_{n,\mathrm{RCSP}}$, is computed by rounding down the theoretical eligibility time, $E_{n,\mathrm{RCSP}} = \Delta\lfloor\frac{\tilde{E}_n}{\Delta}\rfloor$. Then, if $E_{n,\mathrm{RCSP}}$ is equal to the current clock time, it is appended to the corresponding priority queue of the scheduler; otherwise, it is appended to the linked list that the entry $E_{n,\mathrm{RCSP}}$ of the calendar points to. The computation of theoretical eligibility time is done with some errors in acquiring true local-time on packet arrival and in computation due to finite precision arithmetic that is bounded by ε (typically, in the order of a few nanoseconds). We can see that E_{RCSP} satisfies (7.4) when $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}) = (\Delta + \varepsilon, \varepsilon)$.

RGCQ, inspired by the idea of Carousel [177], is an implementation of a damper combined with a queuing system of a scheduler; in other words, each queue of a scheduler is replaced with an RGCQ. An RGCQ consists in a timer and a set of gate-control queues (GCQs). By default, the GCQs are closed and are assigned unique increasing *openTime*s with interspacing of Δ. A GCQ is opened whenever the timer reaches its openTime, and is closed after it is emptied or being opened for a fixed amount of *expiration* time; when a GCQ is closed, its openTime is set as the largest openTime+Δ. The scheduler selects a packet for transmission from an open GCQ with smallest openTime. Whenever a packet n arrives, its theoretical eligibility time is computed based on (7.3); then the actual eligibility time of the packet, $E_{n,\mathrm{RGCQ}}$, is computed by rounding up the theoretical eligibility time, i.e., $E_{n,\mathrm{RGCQ}} = \Delta\lceil\frac{\tilde{E}_n}{\Delta}\rceil$. Then, the packet is enqueued to the GCQ whose openTime is $E_{n,\mathrm{RGCQ}}$. Similarly to RCSP, due to timing acquisitions and arithmetic rounding bounded by ε, we see that E_{RGCQ} satisfies (7.4) when $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}) = (\varepsilon, \Delta + \varepsilon)$.

7.3.2 Dampers with FIFO constraint

The definition of damper with tolerance given in the previous section does not mention whether the damper preserves packet order, and the satisfaction of (7.4) does not preclude packet misordering. Indeed, we show in Section 7.5 that our two examples of dampers with tolerance, namely RCSP and RGCQ, can cause packet misordering due to clock non-idealities. Such a behavior is not possible with a class of proposed damper designs, which enforce the FIFO constraint, and which we now cover.

Re-sequencing damper

We call *re-sequencing damper* with tolerances $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$ a system that behaves as the concatenation of a damper with same tolerances and a re-sequencing buffer that, if needed, re-orders packets based on the packet order at the entrance of the damper. The packet order is with respect to a flow of interest.

Formally, a system is a re-sequencing damper if there exists a sequence $\bar{E}_n$ such that the release times for the flow of interest, in local time, satisfy:

$$\begin{aligned} &\tilde{E}_n - \Delta^{\mathrm{L}} \le \bar{E}_n \le \tilde{E}_n + \Delta^{\mathrm{U}}, \\ &E_1 = \bar{E}_1, \;\; E_n = \max\left\{\bar{E}_n, E_{n-1}\right\}, \end{aligned} \tag{7.5}$$

where $\tilde{E}_n$ is the theoretical eligibility time defined in (7.3) and packet numbers $n = 1, 2, \ldots$ are in order of arrival at the damper.

It follows that such a damper is FIFO for the flow of interest and that[3]:

$$\max_{i \le n}\left\{\tilde{E}_i\right\} - \Delta^{\mathrm{L}} \le E_n \le \max_{i \le n}\left\{\tilde{E}_i\right\} + \Delta^{\mathrm{U}}. \tag{7.6}$$

We say that a re-sequencing damper is *ideal* if has zero tolerances. Hereafter, we describe two instances of re-sequencing dampers, namely, SCED+ [44] and FOPLEQ [168].

SCED+ is an implementation of a damper in combination with SCED scheduling. The damper in [44] is defined as a conceptual element with tolerance Δ. Accordingly, each packet is assigned an initial eligibility time and a terminal eligibility time where the difference between the two is Δ. In SCED+, the damper ensures that the damper is released after the initial eligibility time and before the terminal eligibility time. In fact, the dampers assigns a tentative eligibility time, $\bar{E}_n$ to a packet n, where:

$$\tilde{E}_n - \Delta \le \bar{E}_n \le \tilde{E}_n. \tag{7.7}$$

---

[3] The converse does not hold, i.e., any system that is FIFO for the flow of interest and satisfies (7.6) is not necessarily a re-sequencing damper.

SCED+ assumes that the damper serves packets in FIFO manner; then, the actual eligibility time of the packet n is:

$$E_1 = \bar{E}_1, \;\; E_n = \max\left\{\bar{E}_n, E_{n-1}\right\}; \;\; n \geq 2. \tag{7.8}$$

so that SCED+ is a re-sequencing damper with tolerances $(\Delta+\varepsilon,\varepsilon)$, where ε is a bound on the errors on timing acquisition and arithmetic rounding.

FOPLEQ, similarly to RGCQ, is inspired by the architecture of Carousel [177]. Accordingly, it has a set of time-based queues along with a table, called eligibility time table (ETT), for the purpose of preserving the order of packets inside FOPLEQ. Each row in ETT belongs to a flow that has a packet in the Carousel and stores a tentative eligibility time of the latest packet belonging to the corresponding flow. The tentative eligibility time of a packet is obtained by dividing its theoretical eligibility time by Δ and rounding down the computed value. Consider a packet n of the flow of interest, where number is in the order of arrival at the FOPLEQ. First a theoretical eligibility time is computed using (7.3); second, a tentative eligibility time is obtained by rounding down to a multiple of Δ, i.e. $\bar{E}_n = \Delta\lfloor\frac{\tilde{E}_n}{\Delta}\rfloor$; then, the actual eligibility time of the packet is the maximum of its tentative eligibility time and the stored tentative eligibility time of the flow of interest in the ETT. The tentative eligibility times correspond to a damper with tolerances $(\Delta+\varepsilon,\varepsilon)$, where ε is a bound on the errors on timing acquisition and arithmetic rounding, and therefore FOPLEQ is a re-sequencing damper with tolerances $(\Delta+\varepsilon,\varepsilon)$.

Head-of-line (HoL) damper

The idea is introduced in [166]. An HoL damper is implemented as a FIFO queue. When a packet arrives, its arrival time is collected and the packet is stored at the tail of the queue. Only the packet at the head of the queue is examined; if its eligibility time is passed, it is immediately released, otherwise it is delayed and released at its eligibility time. When the head packet is released, it is removed from the damper queue and the next packet (if any) becomes the head of the queue and is examined. When an arriving packet finds an empty queue, it is immediately examined. By construction, packet ordering is preserved.

As before, the model should incorporate some tolerance to account for the timing inaccuracy and for processing times. Unlike with previous damper models, these two things cannot be aggregated because the head-of-line property has the effect that processing times may have an effect over subsequent packets (this is visible in Theorem 7.4).

Formally, we model a head-of-line damper as follows. It has tolerance parameters $\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}$ that account for the accuracy of timings, as well as processing bounds $\phi^{\min}, \phi^{\max}$ that account for non-zero processing times. We must have $\Delta^{\mathrm{L}} \geq 0, \Delta^{\mathrm{U}} \geq 0$ and $0 \leq \phi^{\min} \leq \phi^{\max}$. Packet numbering is with respect to the order of arrivals at the damper and is global for this damper (not per-flow). We say that a system is a head-of-line damper if the release times E_n, in local time, satisfy:

$$\begin{aligned}\tilde{E}_1 - \Delta^{\mathrm{L}} + \phi^{\min} \le E_1 \le \tilde{E}_1 + \Delta^{\mathrm{U}} + \phi^{\max},\\ \max(\tilde{E}_n - \Delta^{\mathrm{L}}, E_{n-1}) + \phi^{\min} \le E_n \le \max(\tilde{E}_n + \Delta^{\mathrm{U}}, E_{n-1}) + \phi^{\max},\end{aligned} \tag{7.9}$$

where $\tilde{E}_n$ is the theoretical eligibility time as in (7.3).

The definition in (7.9) can be explained as follows. First, the eligibility times are obtained with some errors due to timing acquisition and arithmetic rounding. Let $\bar{E}_n$ be the resulting tentative eligibility times, so that

$$\tilde{E}_n - \Delta^{\mathrm{L}} \le \bar{E}_n \le \tilde{E}_n + \Delta^{\mathrm{U}}. \tag{7.10}$$

Second, packet n is examined only when packet $n-1$ is released, and this action takes a processing time $\phi_n \in [\phi^{\min}, \phi^{\max}]$. The actual release time is therefore

$$E_n = \max(\bar{E}_n, E_{n-1}) + \phi_n. \tag{7.11}$$

Using Lemma D.2 in Appendix D.1 with $a = E_{n-1}$, $x^{\min} = \phi^{\min}$, $x^{\max} = \phi^{\max}$, $y^{\min} = \tilde{E}_n - \Delta^{\mathrm{L}}$, $y^{\max} = \tilde{E}_n + \Delta^{\mathrm{U}}$, $x = \phi_n$, $y = \bar{E}_n$ and $z = E_n$ we obtain that (7.10) and (7.11) imply (7.9); conversely, if (7.9) holds, there exists sequences $\bar{E}_n$ and $\phi_n \in [\phi^{\min}, \phi^{\max}]$ such that (7.10) and (7.11) hold.

If the tolerances and processing bounds are all equal to 0, then the HoL damper is called *ideal*. It follows immediately from (7.9) and (7.5) that an ideal HoL damper is the same as an ideal re-sequencing damper.

In [166], jitter-control ATS is presented as an ideal head-of-line damper in combination with ATS [41] within a switch where each FIFO queue is shared among all time-sensitive flows that come from the same input port, have the same class, and go to the same output port. In [166], the authors implicitly assume that the tolerances and processing times are zero and therefore ignore them in their analysis. This assumption might not hold in practical cases, specifically when a large number of packets become eligible at the same time in a jitter-control ATS. The effect of non-zero tolerances and processing times appear in Theorem 7.4 and is illustrated numerically in Section 7.7.

7.3.3 Damper Header Computation

In this subsection, we first describe the operation of damper header update unit. Then, we discuss the possible sources of error in the computation.

DHU unit operation

The DHU unit of a JCS computes the earliness of a packet and updates the damper header. A classical approach to compute the earliness is to first measure the actual delay of the packet in the JCS with the clock of DHU unit; then set the earliness as the difference between the known delay bound δ of the system for this class of traffic and the actual delay of the packet [42, 43]. More precisely, for a packet n, its arrival time is time stamped with local clock $\mathscr{H}_{\text{TS}}$ and stored locally (Examples 1 and 2 in Section 7.2.1) or delivered by the packet (Example 3 in Section 7.2.1); let $\tilde{A}_n^{\mathscr{H}_{\text{TS}}}$ denote the stored/delivered value. Then the DHU unit time stamps the departure time of the packet with its local clock $\mathscr{H}_{\text{DHU}}$; let $\tilde{W}_n^{\mathscr{H}_{\text{DHU}}}$ denote the departure time. Then, the DHU unit computes the earliness of the packet as

$$\text{earliness}_n = \delta - \left(\tilde{W}_n^{\mathscr{H}_{\text{DHU}}} - \tilde{A}_n^{\mathscr{H}_{\text{TS}}}\right). \tag{7.12}$$

The last step for the DHU unit is to update the damper header that is equal to the current damper header incremented by the computed earliness, and write the result in the damper header field. Then the packet leaves the JCS. In the case that the JCS is connected to an output link, the departure time of a packet is the complete packet transmission and thus the packet header is accessible to write the damper header just before packet transmission. Therefore, the start of transmission time of the packet is time stamped ($T_n^{\mathscr{H}_{\text{DHU}}}$) and the transmission time is inferred as $\tilde{\tau}_n^{\mathscr{H}_{\text{DHU}}} = \left\{\frac{l_n}{c}\right\}^{\mathscr{H}_{\text{DHU}}}$ with l_n as the packet length and c as the transmission rate. Then, we set the departure time to $\tilde{W}_n^{\mathscr{H}_{\text{DHU}}} = T_n^{\mathscr{H}_{\text{DHU}}} + \tilde{\tau}_n^{\mathscr{H}_{\text{DHU}}}$ and compute the earliness using (7.12). This method of damper header computation is used in most of the existing damper variants like RCSP [43], FOPLEQ [168] and jitter-control ATS [166]. We call this the *default* method of damper header computation.

Recently, [167] proposed a subtle change in the computation of earliness when a JCS comes immediately after a damper with tolerances $(\Delta^{\text{L}}, \Delta^{\text{U}})$. In particular, they suggest to time stamp the theoretical eligibility time $\tilde{E}_n$ of packet from the damper instead of the arrival time to the JCS; as a consequence, the jitter imposed by the tolerance of the damper is compensated by the next downstream damper. In such proposal, note that the delay upper-bound between the theoretical eligibility time to the arrival time to the JCS (i.e., the actual eligibility time from the damper) should be added to the earliness; by (7.4), this upper bound is Δ^{U}. Hence, the earliness for theoretical eligibility time stamping is:

$$\text{earliness}_n = \delta + \Delta^{\text{U}} - \left(\tilde{W}_n^{\mathscr{H}_{\text{DHU}}} - \tilde{A}_n^{\mathscr{H}_{\text{TS}}}\right). \tag{7.13}$$

We call this method of damper header computation *TE time-stamping*.

Errors in damper header computation

The DHU unit computes a damper header equal to the current damper header incremented by the computed earliness, and write the result in the damper header field. This step is imperfect due to finite precision arithmetic and finite resolution of the damper header. The corresponding error is $e_{\text{update},n} = H_n - \tilde{H}_n$, where $\tilde{H}_n$ is the theoretical value of the damper header and H_n is the actual value written in the packet.

In the computation of earliness ((7.12) and (7.13)), when $\tilde{A}_n^{\mathscr{H}_{\text{TS}}}$ is delivered within a packet header field (Example 3 in Section 7.2.1), there is some error induced due to the finite resolution of the header field. The corresponding error is $e_{\text{ts},n} = A'^{\mathscr{H}_{\text{TS}}}_n - \tilde{A}_n^{\mathscr{H}_{\text{TS}}}$, where $A'^{\mathscr{H}_{\text{TS}}}_n$ is the time stamped value at the packet arrival to the JCS.

As discussed earlier, when the JCS is connected to a transmission link, the DHU unit can infer the transmission time by dividing the packet length over the nominal transmission rate of the link. Due to transmission of preamble and inexact knowledge of actual transmission rate, the inference of transmission time is done with some error $e_{\text{tran},n} = \tilde{\tau}_n^{\mathscr{H}_{\text{DHU}}} - \tau_n^{\mathscr{H}_{\text{DHU}}}$, where $\tau_n^{\mathscr{H}_{\text{DHU}}}$ and $\tilde{\tau}_n^{\mathscr{H}_{\text{DHU}}}$ are the actual and inferred transmission times. The error $e_{\text{tran},n}$ can go up to tens of nanoseconds [178, 179].

Acquiring the true local-time on packet arrival and within the DHU unit usually comes with an error. We define the clock acquisition error as: $e_{\text{acq},n} = \left(W_n^{\mathscr{H}_{\text{DHU}}} - \tilde{W}_n^{\mathscr{H}_{\text{DHU}}}\right) + \left(A_n^{\mathscr{H}_{\text{TS}}} - A'^{\mathscr{H}_{\text{TS}}}_n\right)$, where $A_n^{\mathscr{H}_{\text{TS}}}$ and $W_n^{\mathscr{H}_{\text{DHU}}}$ are the true local times on packet arrival and departures.

The two clocks $\mathscr{H}_{\text{DHU}}$ and $\mathscr{H}_{\text{TS}}$ are often the same (Examples 1 and 2 in Section 7.2.1), but not always (Example 3 in Section 7.2.1). We select $\mathscr{H}_{\text{DHU}}$ as the reference clock of a JCS to compute damper header; then we define the error with respect to the reference clock as: $e_{\text{clk},n} = A_n^{\mathscr{H}_{\text{DHU}}} - A_n^{\mathscr{H}_{\text{TS}}}$, where $A_n^{\mathscr{H}_{\text{DHU}}}$ is the time that would be displayed at packet arrival if $\mathscr{H}_{\text{DHU}}$ would be used. If the clocks are the same, $e_{\text{clk}} = 0$; if the clocks are synchronized with error ω with respect to TAI, $|e_{\text{clk}}| \leq 2\omega$; and finally if the clocks are not synchronized, e_{clk} can get arbitrary large, which is incompatible with the goal of removing jitter. Therefore, in this chapter, we assume that both clocks $\mathscr{H}_{\text{TS}}$ and $\mathscr{H}_{\text{DHU}}$ are either one and the same, or are synchronized.

To summarize, the value of a damper header, as written in a packet n, suffers from some error e_n equal to: $e_n = e_{\text{update},n} + e_{\text{ts},n} + e_{\text{tran},n} + e_{\text{acq},n} + e_{\text{clk},n}$. Each of these sources of error can be bounded, depending on the technology used by the routers and switches. The variable ϵ denotes an upper bound on the error e_n, i.e., $|e_n| \leq \epsilon$. When $\mathscr{H}_{\text{DHU}}$ and $\mathscr{H}_{\text{TS}}$ are the same, ϵ is typically of the order of tens of nanoseconds; if they are synchronized, ϵ is mainly dominated by the time error bound (e.g. $\epsilon = 2\ \mu$s for gPTP).

7.4 Delay Analysis of Dampers without FIFO constraints

In this section we study the end-to-end delay and jitter of a flow when dampers without FIFO constraint are used. The first step is to decompose a flow path into a set of blocks that can be analyzed separately. Every block is as in Fig. 7.4; it contains a number of JCSs and BDSs and ends in a damper with tolerance. The second step, given in the rest of this section, is to give delay and jitter bounds for a flow through such a block. The bounds are valid whether the JCSs and the BDSs of the block are FIFO or not. The last step, to obtain end-to-end results, simply consists in summing up the delays and jitters of every block and, possibly, of remaining BDSs. For example, in Fig. 7.2, from source to the first router and from the queuing system of the last router to the destination are BDSs and the rest are decomposed in blocks as in Fig. 7.4.

In the following, we give delay and jitter bounds for a block as in Fig. 7.4. Theorem 7.1 gives the result for dampers without FIFO constraint when the default mode of header computation is used (as explained in Section 7.3.3) and Theorem 7.2 when TE time-stamping is used. In both cases, we capture the effect of errors and non-ideal clocks. We also illustrate cases where the errors and non-ideal clocks make a major contribution to the jitter bound.

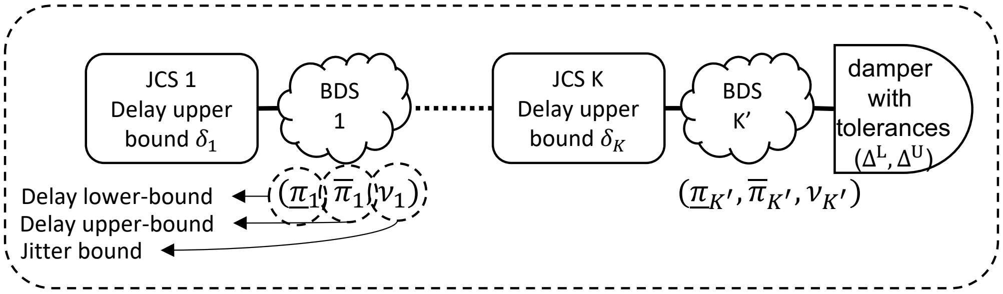

JCS 1
Delay upper bound δ_1
BDS 1
JCS K
Delay upper bound δ_K
BDS K'
damper with tolerances (Δ^L, Δ^U)
Delay lower-bound
Delay upper-bound
Jitter bound
$(\underline{\pi}_1, \overline{\pi}_1, v_1)$
$(\underline{\pi}_{K'}, \overline{\pi}_{K'}, v_{K'})$

Figure 7.4: Structure of a block whose delay and jitter bounds are computed in Theorem 7.1.

Theorem 7.1. *Consider a flow of interest that traverses the block in Fig. 7.4. The block contains a sequence of JCSs and BDSs and terminates in a damper with tolerances* (Δ^L, Δ^U)*. Assume that the clock of every system has stability bound* ρ*, timing-jitter bound* η *and time-error bound* ω *with respect to TAI (Section 7.2.2). Assume that JCS* i *has a delay upper bound* δ_i*, which is used for damper header computation, in its local time. Also, assume that the BDS* j *has delay lower and upper bounds* $\underline{\pi}_j^{\mathcal{H}_{\text{TAI}}}$ *and* $\overline{\pi}_j^{\mathcal{H}_{\text{TAI}}}$ *and jitter bound* $v_j^{\mathcal{H}_{\text{TAI}}}$*, all in TAI. Then, the delay of a packet from entrance to the exit of the block, in TAI, is upper-bounded by* $\overline{D}$*, lower-bounded by* $\underline{D}$ *and has jitter bound* V*, with*

$$\overline{D} = \sum_{j=1}^{K} \delta_j + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathcal{H}_{\text{TAI}}} + \Delta^U + K\epsilon + \overline{\psi},$$

$$\underline{D} = \sum_{j=1}^{K} \delta_j + \sum_{j=1}^{K'} \underline{\pi}_j^{\mathcal{H}_{\text{TAI}}} - \Delta^L - K\epsilon - \underline{\psi},$$

$$V = \sum_{j=1}^{K'} v_j^{\mathcal{H}_{\text{TAI}}} + \Delta^U + \Delta^L + 2K\epsilon + \overline{\psi} + \underline{\psi},$$

where $\overline{\psi}$ *and* $\underline{\psi}$ *are due to clock non-idealities,*

$$
\begin{aligned}
\overline{\psi} &= \min\Big((\rho-1)\big(\Delta^{\mathrm{U}} + \sum_{j=1}^{K}(\delta_j+\epsilon)\big) + (K+1)\eta, 2(K+1)\omega\Big),\\
\underline{\psi} &= \min\Big(\big(1-\frac{1}{\rho}\big)\big(-\Delta^{\mathrm{L}} + \sum_{j=1}^{K}\big(\delta_j-\epsilon\big)\big) + \frac{(K+1)\eta}{\rho}, 2(K+1)\omega\Big).
\end{aligned}
\tag{7.14}
$$

The bounds are tight, i.e., for any tolerances $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$*, every* δ_i*, any* $\underline{\pi}_j^{\mathcal{H}_{\mathrm{TAI}}}$, $\overline{\pi}_j^{\mathcal{H}_{\mathrm{TAI}}}$, $\nu_j^{\mathcal{H}_{\mathrm{TAI}}}$*, there is a system and two individual execution traces such that in one of them a packet experiences a delay of* $\overline{D}$ *and in the other one a packet experiences a delay of* $\underline{D}$*.*

The proof is in Appendix D.2.

Remark. The second arguments of the min(,) functions in (7.14) capture the impact of clock time error bounds when all the $K+1$ clocks (K JCSs and one damper with tolerance) are different from each other. If some systems share a common clock, so that there are $X \le K$ different clocks in total, the second argument of the min(,) functions should be replaced by $2(X+1)\omega$.

Hereafter, we provide an application of Theorem 7.1 to obtain delay and jitter bounds for the three examples of Section 7.2.1. Then we compare the bounds with the *basic* bounds obtained when assuming that clocks are perfect, i.e. by summing the tolerances of dampers and the jitters of BDSs.

Example 1. Consider Example 1 in Section 7.2.1 for a flow that traverses 6 switches. Assume that the switching fabrics (as JCSs) have a delay upper-bound of 2 μs and the delay bound at each queuing system (as a JCS) is 250 μs. Suppose that the error $\epsilon = 50$ ns and all the dampers are RCSP with $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}) = (1\ \mu\text{s}, 2\ \text{ns})$. Assume the propagation delay (as a BDS) is fixed and equal to 5 μs for all links. Assume first that the clocks of the switches are not synchronized, i.e. we set the time-error bound ω to ∞ in (7.14). Then, by applying Theorem 7.1 from source to the output of the first damper, the delay upper-bound is $\overline{D} = 257.13\ \mu$s; the delay jitter is $V = 1.264\ \mu$s of which 200 ns is due to errors and 62 ns is due to non-ideal clocks. The basic jitter bound is 1.002 μs and is due to the tolerance of the first damper. We can see that the error and non-ideal clocks add 262 ns (26%) to the basic jitter bound. The end-to-end delay and jitter bounds are computed by summing up the delay and jitter bounds from the output of one damper to the output of the next downstream damper until the destination; this gives an end-to-end delay upper-bound of 1.799 ms and end-to-end jitter bound of 8.834 μs. The values remain the same if we assume next that the clocks of the switches are synchronized with a time-error bound of 1 μs, as is typical in IEEE TSN systems.

Example 2. Consider Example 2 in Section 7.2.1 for a flow that traverses four access routers to reach the backbone and traverses four other access routers to reach to the destination. Assume that 1) the queuing delay at source has a delay upper-bound of 100 μs and the jitter bound of 80 μs, 2) the delay upper-bound at the output queuing and packet forwarding of each access router are 500 μs and 5 μs and the output queuing has jitter of 100 μs, 3) the backbone network has a delay upper-bound of 30 ms and jitter bound of 1 ms, 4) the propagation delay is 10 μs, 5) the error $\epsilon = 50$ ns and 6) all dampers are RCSP with $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}) = (1\ \mu\mathrm{s}, 2\ \mathrm{ns})$. Then, by using Theorem 7.1, the end-to-end delay upper-bound is $\overline{D}_{\mathrm{e2e}} = 34.211$ ms; delay jitter is $V_{\mathrm{e2e}} = 1.190$ ms of which 1.5 μs is due to the errors and 800 ns is due to non-ideal clocks. The basic jitter bound is 1.188 ms that is due to the tolerance of the dampers, as well as the BDSs, i.e., the backbone network, the source output queuing and the output queuing of the last access router (before the destination). We can see that here the effect of the errors and non-ideal clocks is negligible due to the jitter of the BDSs captured by the basic jitter bound.

Example 3. Consider Example 3 in Section 7.2.1 for a flow that traverses four access routers to reach the backbone and then traverses four other access routers to reach to the destination. Also, consider the same numerical assumptions as the previous example. We want to remove jitter of the backbone network using time stamping at the upstream PE router. Assume that the PE routers are synchronized with time-error bound of 1 μs; hence the error damper header computation at the downstream PE router of the backbone network is bounded by 2.05 μs (the error at the other JCSs is bounded by 50 ns). Then, by using Theorem 7.1, the end-to-end delay upper-bound is $\overline{D}_{\mathrm{e2e}} = 34.216$ ms; delay jitter is $V_{\mathrm{e2e}} = 21.74$ μs of which 5.8 μs is due to the errors and 6.92 μs is due to non-ideal clocks. Comparing to the previous example, the basic jitter bound is reduced to 9.02 μs as the jitter of the backbone network, queuing at source and the queuing at the last access router are removed. We can see that the errors and clock non-idealities add 12.72 μs (141%) to the basic jitter bound.

We see from these examples that, when the remaining end-to-end delay-jitter is still large after applying dampers (ms or more, Example 2) then the timing errors and clock non-idealities do not play a significant role and can be ignored. In contrast, for very small residual delay-jitter (Examples 1 and 2, 10 μs or less), ignoring timing errors and clock non-idealities can lead to significant under-estimation.

Remark. In Example 1, we see that the delay-jitter bound is not affected by the time-error bound, i.e., here, time synchronization does not improve the performance of dampers. We can easily analyze when this is the case, by comparing the terms in the min(.) functions in (7.14). We find that time synchronization does not improve the performance of dampers if and only if

$$\sum_{j=1}^{K} \delta_j \leq \frac{K+1}{\rho-1}\left(2\omega-\eta\right) - \Delta^{\mathrm{U}} - K\epsilon. \tag{7.15}$$

It follows that if $\sum_{j=1}^{K} \delta_j > \frac{2}{\rho-1}\left(2\omega-\eta\right)$, the time error bound affects the delay and jitter bounds of Theorem 7.1; i.e., it is the relation between the delay (not delay-jitter) bound and the time-error bound that matters (see Table 7.1).

TSN networks are typically synchronized with gPTP ($\omega = 1$ μs) [175, Section B.3]. Three main delay sensitive classes are CDT, class A for audio traffic and class B for video traffic. According

Table 7.1: Minimum values of the sum of delay bounds for the JCSs within a block such that clock synchronization improves the delay and jitter bounds in Theorem 7.1.

Synchronization method	Time-error bound (ω)	Minimum value of $\sum_{j=1}^{K}\delta_j$
White Rabbit	100ns	3.96ms
gPTP	1μs	39.96ms
NTP	100ms	3.99s

to TSN documents [180, 181, 9], the end-to-end delay requirement for CDT, classes A and B are respectively 100 μs in 5 hops, 2 ms and 50 ms in 7 hops. According to Table 7.1, the gPTP synchronization does not impact the obtained delay bound using Theorem 7.2 for CDT and class A. For class B, if we consider that for each block the sum of JCS delay bounds is less than 39.96 ms, similarly the gPTP synchronization does not play a role. This implies when all switches and the destinations in a TSN network implement dampers with tolerances and the source performs time stamping (as a JCS), then without gPTP synchronization the same performance is achieved.

In order to provide delay and delay-jitter guarantees to time-sensitive flows, it is often required to bound the burstiness of flows inside the network, which is typically larger than at the source. Finding such bounds may be difficult, and worst-case bounds may be large when there are cyclic dependencies [133]. Here, dampers can help a lot, as shown by the following Corollary, which comes by direct application of the jitter bound in Theorem 7.1 and [64, Lemma 1].

Corollary 7.1. *Consider Fig. 7.4. Suppose that a flow has* $\alpha^{\mathcal{H}_{\text{TAI}}}$ *as arrival curve at the entrance of the block. Then an arrival curve at the output of the block, in TAI, is given by* $\alpha_h^{\mathcal{H}_{\text{TAI}}}(t) = \alpha^{\mathcal{H}_{\text{TAI}}}(t+V)$ *where* V *is the jitter bound of the block defined in Theorem 7.1.*

Example. In Example 1 of Section 7.2.1, suppose that the flow has leaky-bucket arrival curve with rate 16 Mbps, in TAI, and burstiness 10 KBytes at source. We computed that the jitter bound is 1.262 μs from source the output of the first damper. Then the arrival curve at the output of the first damper has the same rate and the burstiness is increased by 3 Bytes. Without damper, the burstiness increase would be 515 Bytes: we see that the burstiness increase due to multiplexing is almost entirely removed.

Remark. The arrival curve constraint at source may be available in its local time rather than in TAI. Then, we can apply (7.2), to obtain an arrival curve in TAI and then use the result of Corollary 7.1.

When dampers use TE time-stamping for damper header computation rather than the default method, the delay and jitter bounds computation with TE time-stamping are slightly different than in Theorem 7.1. A JCS is affected only when the upstream damper uses TE time-stamping (otherwise, the bounds are the same). The next theorem gives end-to-end delay and jitter bounds when DHU unit uses TE time-stamping.

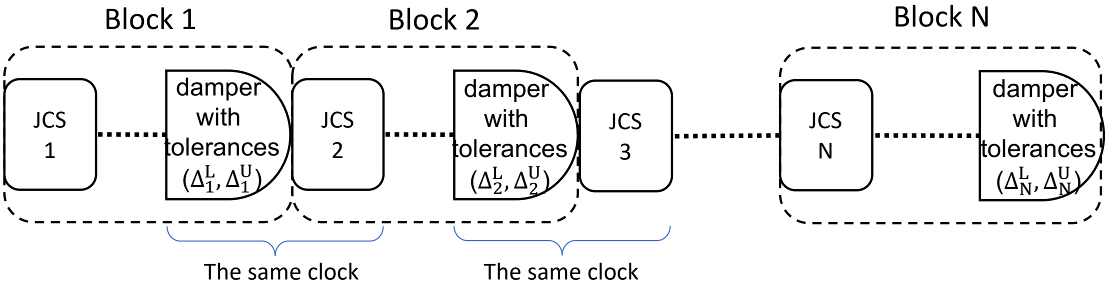

Block 1
Block 2
Block N
JCS 1
damper with tolerances $(\Delta_1^{\mathrm{L}}, \Delta_1^{\mathrm{U}})$
JCS 2
damper with tolerances $(\Delta_2^{\mathrm{L}}, \Delta_2^{\mathrm{U}})$
JCS 3
JCS N
damper with tolerances $(\Delta_N^{\mathrm{L}}, \Delta_N^{\mathrm{U}})$
The same clock
The same clock

Figure 7.5: The notation used in Theorem 7.2.

Theorem 7.2. *Consider Fig. 7.5 where a sequence of N blocks are concatenated and TE time-stamping is used for damper header computation. Assume that clocks follow the description in Section 7.2.2 and damper with tolerances at block i* $(i = 1,\dots,N-1)$ *and JCS* $i+1$ *operate with the same clock. Assume that there are M JCSs in total. Let us call the sum of the delay bounds of the JCSs* δ_{e2e} *and the sum of delay lower and upper bounds and jitter bound of the BDSs* $\underline{\pi}_{\mathrm{e2e}}, \overline{\pi}_{\mathrm{e2e}}$ *and* v_{e2e} *respectively. Then,*

$$\begin{aligned}
\overline{D}_{\mathrm{TE}} &= \delta_{\mathrm{e2e}} + \overline{\pi}_{\mathrm{e2e}} + \sum_{j=1}^{N} \Delta_j^{\mathrm{U}} + M\epsilon + \overline{\Psi}_{\mathrm{TE}},\\
\underline{D}_{\mathrm{TE}} &= \delta_{\mathrm{e2e}} + \underline{\pi}_{\mathrm{e2e}} + \sum_{j=1}^{N-1} \Delta_j^{\mathrm{U}} - \Delta_N^{\mathrm{L}} - M\epsilon - \underline{\Psi}_{\mathrm{TE}},\\
V_{\mathrm{TE}} &= v_{\mathrm{e2e}} + \Delta_N^{\mathrm{U}} + \Delta_N^{\mathrm{L}} + 2M\epsilon + \overline{\Psi}_{\mathrm{TE}} + \underline{\Psi}_{\mathrm{TE}},
\end{aligned} \tag{7.16}$$

where $\overline{\Psi}_{\mathrm{TE}}$ *and* $\underline{\Psi}_{\mathrm{TE}}$ *are the errors due to non-ideal clocks:*

$$\begin{aligned}
\overline{\Psi}_{\mathrm{TE}} &= \min\Big((\rho-1)\big(\delta_{\mathrm{e2e}} + \sum_{i=1}^{N} \Delta_i^{\mathrm{U}} + M\epsilon\big) + (M+N)\eta, 2(M+N)\omega\Big),\\
\underline{\Psi}_{\mathrm{TE}} &= \min\Big(\big(1-\frac{1}{\rho}\big)\big(\delta_{\mathrm{e2e}} - \Delta_N^{\mathrm{L}} + \sum_{i=1}^{N-1} \Delta_i^{\mathrm{U}} + M\epsilon\big) + \frac{(M+N)\eta}{\rho}, 2(M+N)\omega\Big).
\end{aligned} \tag{7.17}$$

The proof is available in Appendix D.3.

Example. Let us redo the end-to-end delay and jitter bounds computation for the three examples of Section 7.2.1 using Theorem 7.2 and compare them with the ones obtained with Theorem 7.1. Let us consider the same assumptions made when applying Theorem 7.1. We can see that the delay upper-bounds obtained by Theorem 7.2 are the same as the ones computed after Theorem 7.1; however, the end-to-end jitter is reduced. Using Theorem 7.2, the end-to-end jitter bound for Example 1 is $V_{\mathrm{Ex1}} = 2.834\ \mu$s, Example 2 is $V_{\mathrm{Ex2}} = 1.183$ ms, Example 3 is $V_{\mathrm{Ex3}} = 13.74\ \mu$s. The reason for jitter bound reduction by Theorem 7.2 is the elimination of the jitter imposed by the tolerances of all the intermediate dampers by the next downstream dampers. In examples 1 and 3, the jitter bounds are considerably reduced, by 68% and 36%; however, this is not the case for Example 2 as the main sources of jitter are the BDSs. Furthermore, the jitter imposed by the errors and the non-ideal clocks incorporate 65% and 92% of the the end-to-end jitter bounds computed for examples 1 and 3, which are respectively 2.8 and 13.74 times the basic jitter bounds.

7.5 Packet Reordering in Dampers without FIFO Constraints

In this Section we show that dampers without FIFO constraint can cause packet misordering, and we quantify the corresponding reordering metrics.

Obviously, a damper modifies packet order if the sequence of theoretical eligibility times is not monotonic. Since the theoretical eligibility time is equal to the arrival time at the JCS plus a constant, this may occur only if the packet order at the entrance to the damper is not the same as at the entrance to the JCS, i.e. this requires the JCS to be non FIFO. But, as we show next, this may occur even if the JCS is FIFO, due to timing inaccuracies.

RGCQ and RCSP are two instances of dampers with tolerance; by design, they avoid packet reordering due to the tolerances by enforcing FIFO behavior after computation of theoretical eligibility times. However, as we show next, packet reordering may still occur within RGCQ and RCSP due to the errors of damper header computation and non-ideal clocks.

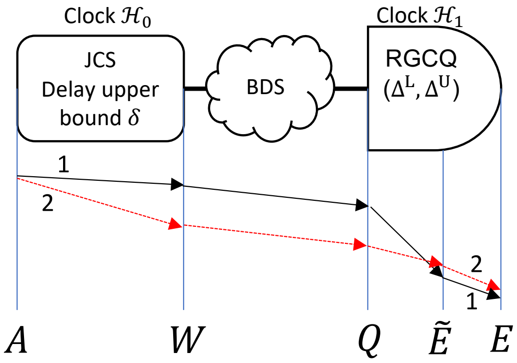


Figure 7.6: Packet reordering scenario when two packets enter back-to-back to a JCS.

Reordering example with RGCQ. Consider Fig. 7.6 where the damper is RGCQ with tolerances (Δ^L, Δ^U) and clocks are not synchronized. Assume that the JCS represents a FIFO queue connected to a transmission line with a fixed rate and the BDS has zero jitter and represents constant propagation delay (similar to the first hop in Example 1 of Section 7.2.1). Suppose that two packets 1 and 2 enter the JCS at the same time while 1 is prior to 2. Then packet 1 leaves before packet 2. The damper headers in the packets are:

$$\begin{aligned} H_1 &= \delta - \tilde{\tau}_1^{\mathscr{H}_0}, \\ H_2 &= \delta - \tilde{\tau}_1^{\mathscr{H}_0} - \tilde{\tau}_2^{\mathscr{H}_0} = H_1 - \tilde{\tau}_2^{\mathscr{H}_0}, \end{aligned} \tag{7.18}$$

where $\tilde{\tau}_i^{\mathscr{H}_0}$ is the inferred transmission times of packets $i \in \{1,2\}$ measured with $\mathscr{H}_0$. Then, the interspacing of the two packet at the output of the JCS is:

$$W_2^{\mathscr{H}_0} - W_1^{\mathscr{H}_0} = \tau_2^{\mathscr{H}_0}, \tag{7.19}$$

where $\tau_2^{\mathscr{H}_0}$ is the actual transmission time of packet 2 and $W_i^{\mathscr{H}_0}$ is the departure time of packet $i \in \{1,2\}$ from the JCS. Both packets experience the same delay in the BDS. Therefore, the interspacing between the two packets at the entrance of RGCQ when seen with clock $\mathscr{H}_1$ is:

$$Q_2^{\mathscr{H}_1} - Q_1^{\mathscr{H}_1} = \tau_2^{\mathscr{H}_1} = \tau_2^{\mathscr{H}_0} + \left(\tau_2^{\mathscr{H}_1} - \tau_2^{\mathscr{H}_0}\right), \tag{7.20}$$

where $Q_i^{\mathscr{H}_1}$ is the arrival time of packet $i \in \{1,2\}$ to RGCQ. Then, by (7.18) and (7.3), the difference between the theoretical eligibility times of packets 2 and 1 is:

$$\begin{aligned} \tilde{E}_2^{\mathscr{H}_1} - \tilde{E}_1^{\mathscr{H}_1} &= Q_2^{\mathscr{H}_1} - Q_1^{\mathscr{H}_1} + H_2 - H_1 \\ &= \left(\tau_2^{\mathscr{H}_0} - \tilde{\tau}_2^{\mathscr{H}_0}\right) + \left(\tau_2^{\mathscr{H}_1} - \tau_2^{\mathscr{H}_0}\right). \end{aligned} \tag{7.21}$$

The difference between the theoretical eligibility times is the sum of the error between actual and inferred transmission time and the measurement difference of packet 2 transmission time seen from clocks $\mathscr{H}_1$ and $\mathscr{H}_0$. Therefore, if it happens that clock $\mathscr{H}_1$ is faster than $\mathscr{H}_0$ during the transmission time of packet 2 from the JCS, then $\tau_2^{\mathscr{H}_1} < \tau_2^{\mathscr{H}_0}$, hence $\tilde{E}_2^{\mathscr{H}_1} - \tilde{E}_1^{\mathscr{H}_1} < 0$, i.e. packet 2 has smaller theoretical eligible time than packet 1. Then, by implementation of RGCQ discussed in Section 7.3, packet 2 leaves RGCQ before packet 1.

Remark. In this scenario, reordering occurs because of the difference of speed between the two clocks $\mathscr{H}_0$ and $\mathscr{H}_1$ at the microscopic scale and the error in inferring the transmission time. The earliness of a packet written in the header is measured using the local clock $\mathscr{H}_0$ while the delay imposed to the packet is measured in the RGCQ using the local clock $\mathscr{H}_1$. Even if both systems are time synchronized, there still remains a small difference in the time measurements performed by the two clocks. Over the transmission time of a packet, there is equal chance that one clocks ticks slightly faster than the other, i.e. there is 50% chance that the change of order described in this scenario occurs.

Reordering example with RCSP. Consider Fig. 7.7 where the dampers are RCSP. Assume that the first JCS represents a router, the second JCS is a FIFO queue connected to a transmission line with a fixed rate and the BDS has zero jitter and represents a constant propagation delay; this resembles the first and second access routers in Example 2 of Section 7.2.1. Now, focus on the first JCS and the first RCSP. Suppose two packets 1 and 2 enter the JCS with interspacing $\tau^{\mathscr{H}_{\text{TAI}}}$, measured in TAI (e.g. transmission time of packet 2 from source, when packets are sent

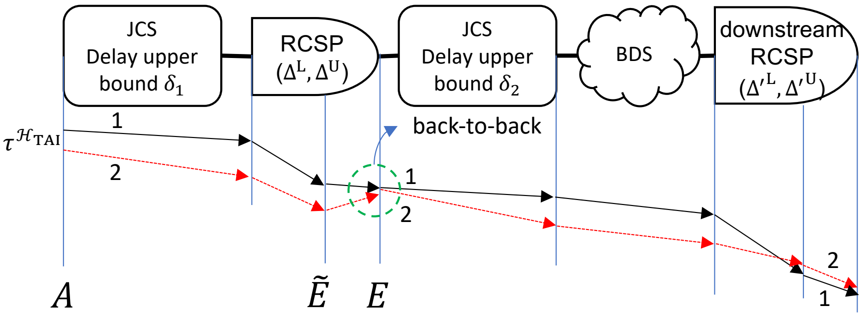

JCS
Delay upper bound δ_1
RCSP
$(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$
JCS
Delay upper bound δ_2
BDS
downstream RCSP
$(\Delta'^{\mathrm{L}}, \Delta'^{\mathrm{U}})$
back-to-back
$\tau^{\mathcal{H}_{\mathrm{TAI}}}$
1
2
1
2
2
1
A
$\tilde{E}$
E

Figure 7.7: A scenario that two packets become back-to-back after an RCSP and reordering occurs in the downstream RCSP.

back-to-back from source), i.e.,

$$A_2^{\mathcal{H}_{\mathrm{TAI}}} - A_1^{\mathcal{H}_{\mathrm{TAI}}} = \tau^{\mathcal{H}_{\mathrm{TAI}}}. \tag{7.22}$$

Let κ denote the delay difference of two packets from entrance of the JCS to the theoretical eligibility time of the RCSP in TAI, i.e.,

$$\kappa^{\mathcal{H}_{\mathrm{TAI}}} \triangleq \left(\tilde{E}_1^{\mathcal{H}_{\mathrm{TAI}}} - A_1^{\mathcal{H}_{\mathrm{TAI}}}\right) - \left(\tilde{E}_2^{\mathcal{H}_{\mathrm{TAI}}} - A_2^{\mathcal{H}_{\mathrm{TAI}}}\right). \tag{7.23}$$

$|\kappa^{\mathcal{H}_{\mathrm{TAI}}}|$ is less than the jitter bound of Theorem 7.1 when $\Delta^{\mathrm{L}} = \Delta^{\mathrm{U}} = 0$; $|\kappa^{\mathcal{H}_{\mathrm{TAI}}}| \leq V^0 = \overline{\psi} + \underline{\psi} + 2\epsilon$. Then by (7.22), we have:

$$\tilde{E}_1^{\mathcal{H}_{\mathrm{TAI}}} - \tilde{E}_2^{\mathcal{H}_{\mathrm{TAI}}} = \tau^{\mathcal{H}_{\mathrm{TAI}}} + \kappa^{\mathcal{H}_{\mathrm{TAI}}}, \tag{7.24}$$

which shows the interspacing between the theoretical eligibility times of two packets at the RCSP in TAI. Observing this interspacing with $\mathcal{H}_2$, we obtain

$$\tilde{E}_1^{\mathcal{H}_2} - \tilde{E}_2^{\mathcal{H}_2} = \tau^{\mathcal{H}_{\mathrm{TAI}}} + \kappa^{\mathcal{H}_{\mathrm{TAI}}} + \gamma, \tag{7.25}$$

where γ is the difference in the measurement of the interspacing between $\mathcal{H}_{\mathrm{TAI}}$ and $\mathcal{H}_2$, bounded by (7.1). The actual eligibility times of the packets from RCSP are obtained by getting the floor of the theoretical eligibility times divided by Δ (Section 7.3). Hence, if $\tau^{\mathcal{H}_{\mathrm{TAI}}} + \kappa^{\mathcal{H}_{\mathrm{TAI}}} + \gamma < \Delta^{\mathrm{L}}$, the two packets may have the same actual eligibility times (leave RCSP back-to-back). The probability of this phenomenon is

$$\mathbb{P}\,[\text{backToBack}] = 1 - \frac{\tau^{\mathcal{H}_{\mathrm{TAI}}} + \kappa^{\mathcal{H}_{\mathrm{TAI}}} + \gamma}{\Delta^{\mathrm{L}}} \approx 1 - \frac{\tau^{\mathcal{H}_{\mathrm{TAI}}}}{\Delta^{\mathrm{L}}},$$

which implies that, the smaller the interspacing of the two packets when entering the JCS, the larger is the probability of having the two packets with the same actual eligibility time and leave the RSCP back-to-back. When two packets are back-to-back from one RCSP, then similar to scenario 1, there is 50% chance of reordering for the two packets at the output of the next downstream RCSP. Due to the independence of the two events (being back-to-back at the output of the RCSP and reordering at next downstream RCSP), the chance of reordering is $0.5\left(1-\tau^{\mathscr{H}_{\mathrm{TAI}}}/\Delta^{\mathrm{L}}\right)$.

One approach to tackle the reordering issue of dampers with tolerance is to place re-sequencing buffers after the dampers to correct the reordering that they cause. With this approach, it is crucial to find proper time-out value and size for the re-sequencing buffers. As shown in Chapter 6, two reordering metrics, namely reordering late-time offset (RTO) and reordering byte offset (RBO) respectively give the time-out value and size of a re-sequencing buffer. We obtain these metrics for dampers as a direct application of theorems 6.5 and 6.6:

Corollary 7.2. *Consider Fig. 7.4 and a flow that has arrival curve* $\alpha^{\mathscr{H}_{\mathrm{TAI}}}$ *at the entrance of the block. Then, the RTO for the flow from the entrance of the block to the output of the damper with tolerance, measured in TAI, is* λ^{TAI} *and the corresponding RBO is* ζ*:*

$$\lambda^{\mathrm{TAI}}=\left[V-\left(\alpha^{\mathscr{H}_{\mathrm{TAI}}}\right)^{\downarrow}(2L^{\min})\right]^{+}, \tag{7.26}$$

$$\zeta=\alpha^{\mathscr{H}_{\mathrm{TAI}}}(V)-L^{\min}, \tag{7.27}$$

where V *is the jitter bound of the block, computed in Theorem 7.1,* $L^{\min}$ *is the minimum packet length of the flow and* $\left(\alpha^{\mathscr{H}_{\mathrm{TAI}}}\right)^{\downarrow}$ *is the lower pseudo-inverse function defined as*

$$\left(\alpha^{\mathscr{H}_{\mathrm{TAI}}}\right)^{\downarrow}(x)=\inf\left\{t\geq 0|\alpha^{\mathscr{H}_{\mathrm{TAI}}}(t)\geq x\right\}.$$

Example. Consider Example 1 of Section 7.2.1 with the same assumptions made after Theorem 7.1. Suppose that a flow has leaky-bucket arrival curve with rate 16 Mbps, in TAI, burstiness 10 KBytes at source and minimum packet length 100 Bytes. We computed that the jitter bound is 1.262 μs from the source to the output of the first damper. Then the RTO (time-out value) is 1.262 μs and the RBO (required buffer size) is 10003 Bytes.

Another approach to tackle reordering is to use dampers with FIFO constraint, as discussed in Section 7.3 and analyzed in the next section.

7.6 Analysis of Dampers with FIFO Constraints

As mentioned earlier, one way to avoid packet reordering within dampers with tolerance is to replace them with dampers with FIFO constraint, namely, re-sequencing and HoL dampers. The goal of this section is to provide delay and jitter bounds when dampers with FIFO constraint are used. In this context, “FIFO" and “re-sequencing" are per flow. When all the BDSs and JCSs within a flow path are FIFO, using re-sequencing or HoL dampers, in contrary to dampers with tolerances, can provide end-to-end in-order packet delivery. However, this might impact the delay and jitter bounds computed in Theorem 7.1. To this end, we capture the impact of using re-sequencing or HoL dampers instead of dampers with tolerances in terms of delay and jitter bounds in Theorem 7.3 and Theorem 7.4 when all systems are FIFO. Then, we see in Theorem 7.5 and Theorem 7.6 that the presence of a non-FIFO system (BDS or JCS) in the flow path considerably worsens the delay and jitter bounds obtained when all systems are FIFO. This phenomenon does not occur with dampers without FIFO constraint because the results in Section 7.4 hold whether the JCSs and BDSs are FIFO or not.

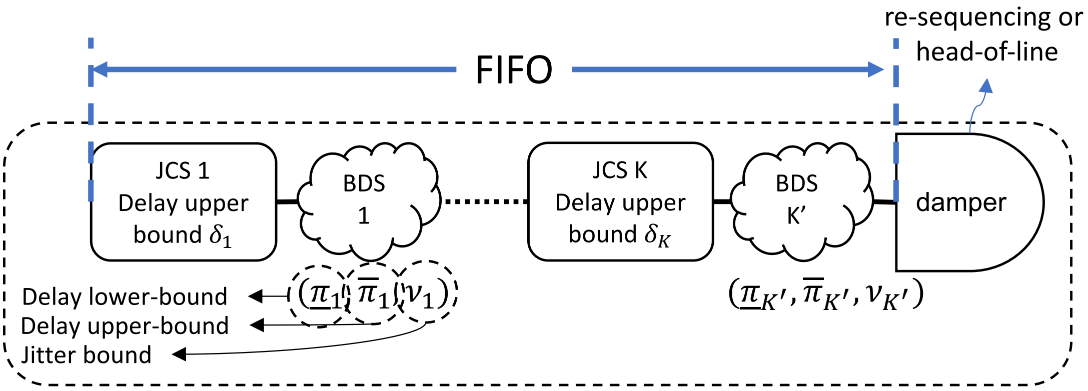

re-sequencing or
head-of-line
FIFO
JCS 1
Delay upper bound δ_1
BDS 1
JCS K
Delay upper bound δ_K
BDS K'
damper
$(\underline{\pi}_1, \overline{\pi}_1, \nu_1)$
$(\underline{\pi}_{K'}, \overline{\pi}_{K'}, \nu_{K'})$
Delay lower-bound
Delay upper-bound
Jitter bound

Figure 7.8: The notation used in Theorem 7.3 and Theorem 7.4.

Theorem 7.3. *Consider the block of systems in Fig. 7.8 where all the JCSs and BDSs are FIFO and the damper is an instance of re-sequencing dampers with tolerances* $\left(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}\right)$. *Assume that the clocks follow the description in Section 7.2.2. Then, the delay and jitter bounds of the block, in TAI, is the same as the bounds in Theorem 7.1.*

The proof is in Appendix D.4. It consists in two steps. First, we use an abstraction of a re-sequencing damper with tolerances $\left(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}\right)$ as a damper with tolerances $\left(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}\right)$ followed by a re-sequencing buffer that preserve the order of packet at their entrance to the damper with tolerances. Second, by the re-sequencing-for-free property of the re-sequencing buffers, discussed in Chapter 6, we obtain the bounds.

Remark. We have seen in the previous section that even if all BDSs and JCSs are FIFO in a flow path, dampers with tolerance may cause packet reordering due to the tolerances, non-ideal clocks and errors in packet header computation. Theorem 7.3 indicates that in such a case, placing a re-sequencing damper avoids packet reordering with the same delay and jitter bounds as if dampers with tolerances are used.

Theorem 7.4. *Consider the block of systems in Fig. 7.8 where all the JCSs and BDSs are FIFO and the damper is an instance of head-of-line dampers with tolerances* ($\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}$) *and processing-time bounds* ($\phi^{\min}, \phi^{\max}$)*. Assume that the clocks follow the description in Section 7.2.2. Then if* $\phi^{\max} = 0$*, the delay and jitter bounds are the same as the bounds in Theorem 7.1. Otherwise, for a flow with per-packet arrival curve* α *at the entrance of the block,*

1. *the delay upper-bound is increased by* θ*,*
2. *the delay lower-bound is increased by* $\phi^{\min}$*,*

3. *the jitter bound is increased by* $\theta - \phi^{\min}$,

where θ *is a delay upper-bound of a single-server FIFO queue with maximum processing time of* $\phi^{\max}$*, computed as*

$$\theta = \max_{k \in \mathbb{N}} \left\{ k\phi^{\max} - \alpha^{\downarrow}(k) + V \right\}, \tag{7.28}$$

where V *is the jitter bound computed in Theorem 7.1 and*

$$\alpha^{\downarrow}(k) = \inf\{t \geq 0 | \alpha(t) \geq k\}.$$

The proof is in Appendix D.5. The proof consists in two steps. First, we prove that an HoL damper is equivalent to re-sequencing damper with tolerances $(\Delta^{\text{L}}, \Delta^{\text{U}})$ followed by a single-server FIFO queue with service times within $(\phi^{\min}, \phi^{\max})$. Second, using the bounds of Theorem 7.3 and obtaining delay and jitter bounds on the single-server queue, the theorem is proven.

Remark. HoL dampers, contrary to re-sequencing dampers, impose some queuing delay, captured by θ in (7.28). The queuing delay is maximized for the last packet of a packet sequence when all become eligible at the same time; then since the HoL damper examines only the packet at the head of the queue, the last packet of the sequence is delayed as much as the processing delay of all the preceding packets.

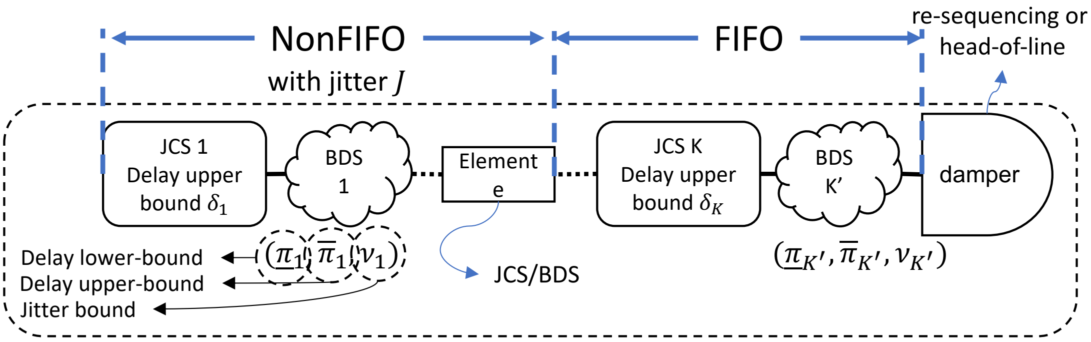

NonFIFO
with jitter J
FIFO
re-sequencing or
head-of-line
JCS 1
Delay upper bound δ_1
BDS 1
Element e
JCS K
Delay upper bound δ_K
BDS K'
damper
$(\underline{\pi}_1, \overline{\pi}_1, v_1)$
Delay lower-bound
Delay upper-bound
Jitter bound
JCS/BDS
$(\underline{\pi}_{K'}, \overline{\pi}_{K'}, v_{K'})$

Figure 7.9: The notation used in Theorem 7.5 and Theorem 7.6.

So far we provide delay and jitter bounds when all the systems are FIFO; however, the FIFO condition might not be always met for all systems such as multi-stage switching fabrics, multi-path routing of packets or packet duplication [48, 49, 50]. Hence, in the following theorems we capture the impact of non-FIFO behavior of systems on the delay and jitter bounds when re-sequencing and HoL dampers are used.

Theorem 7.5. *Consider Fig. 7.9 where system e (a BDS or a JCS) is the last non-FIFO system in the block and the damper is an instance of re-sequencing damper. Let us call J as the jitter from JCS 1 to system e (included), in TAI. Then, the delay upper-bound and jitter bound of the block, in TAI, are increased by J comparing to the bounds in Theorem 7.3.*

The bounds are tight, i.e., for any packet that experiences the delay equal to $\overline{D}$*, there is system and an execution trace that another packet experiences a delay equal to* $\overline{D}+J$.

The proof is in Appendix D.6. The proof has two parts. First, we show that delay upper-bound is increased by J while the delay lower-bound remains unchanged. Second, we provide a scenario where two packets with interspacing J enter the block and leave the element e back-to-back while their order is changed. We show that when the second packet experiences a delay $\overline{D}$, the first packet experiences a delay of $\overline{D}+J$ and the second packet leaves the re-sequencing damper before the first packet.

Theorem 7.6. *Consider Fig. 7.9 where system e (a BDS or a JCS) is the last non-FIFO system in the block and the damper is an instance of HoL damper. Let us call J as the jitter from JCS 1 to system e (included), in TAI. Then, comparing to Theorem 7.4, the delay upper-bound and jitter bound of the block, in TAI, are increased by J if* $\phi^{\max}=0$*, and are increased by* $2J$ *if* $\phi^{\max}>0$.

The proof is in Appendix D.7. The proof consists in two steps. First, similarly to the proof of Theorem 7.3, we abstract an HoL damper as a re-sequencing damper followed by a single-server FIFO queue. Second, by summing the bounds obtained in Theorem 7.5 and the bounds on the FIFO queue, the statement is proven. In the case $\phi^{\max}>0$, the bounds are increased once by J within the re-sequencing damper and once within the FIFO queue as a result of propagated arrival curve at the output of the re-sequencing damper.

Remark. Similarly to Corollary 7.1, propagated arrival curve of a flow, with arrival curve $\alpha^{\mathscr{H}_{\text{TAI}}}(t)$ at the entrance of a block, is $\alpha^{\mathscr{H}_{\text{TAI}}}(t+\bar{V})$ at the output of re-sequencing or HoL damper, where $\bar{V}$ is the jitter of the block computed by applying the corresponding theorem.

Remark. Theorem 7.5 and Theorem 7.6 show that when there is a non-FIFO system in a block, placement of a damper with FIFO constraint is counterproductive. First, comparing to placement of dampers with tolerances, the jitter is increased; in result, it leads to an increase in the burstiness of the propagated arrival curve. Second, the damper with FIFO constraint preserves the wrong order of the packets, which occurred within the non-FIFO system.

7.7 Numerical Evaluation

We illustrate our theoretical results on the Orion crew exploration vehicle network, as described in [164] and depicted in Fig. 6.5. For the delay and jitter analysis, we used Fixed-Point TFA [133, 132] as there are cyclic dependencies. The device clocks are not synchronized. The link rates are 1 Gbps. The output ports use a non-preemptive TSN scheduler with CBSs and per-class FIFO queuing [63, 103]; from highest to lowest priority, the classes are CDT, A, B, and Best Effort (BE). The CBSs are used separately for classes A and B. The CBS parameters *idleslope*s are set to 50% and 25% of the link rate respectively for classes A and B [103]. In each switch, the switching fabric has a delay between 0.5 μs to 2 μs [163]. The CDT traffic has a leaky-bucket arrival curve with rate 6.4 kilobytes per second and burst 64 bytes. The maximum packet length of classes B and BE is 1500 bytes. We focus on class A. Using Theorem 5.2, a rate-latency service curve offered to class A is $\beta(t) = 62.49e6[t - t_0]^+$ bytes with $t_0 = 12.5$ μs.

Class A contains 40 flows with constant packet size 147 bytes, which transmit 10 packets every 8 ms. The flows traverse between 2 to 9 hops. We assume all switching fabrics and output queuing systems implement DHU unit and therefore are JCSs; the propagation delays are considered as BDSs with zero jitter. We examine the case where no damper is placed and the case where dampers are placed at every switch and the destinations. For the choice of dampers, we considered individually the full deployment of RCSP ($\Delta^{\text{L}} = 1$ μs, $\Delta^{\text{U}} = 2$ ns), RGCQ ($\Delta^{\text{L}} = 2$ ns, $\Delta^{\text{U}} = 1$ μs) with TE time-stamping, FOPLEQ ($\Delta^{\text{L}} = 1$ μs, $\Delta^{\text{U}} = 2$ ns) and a head-of-line damper ($\phi^{\min} = 0, \phi^{\max} = 5$ ns, $\Delta^{\text{L}} = \Delta^{\text{U}} = 2$ ns).

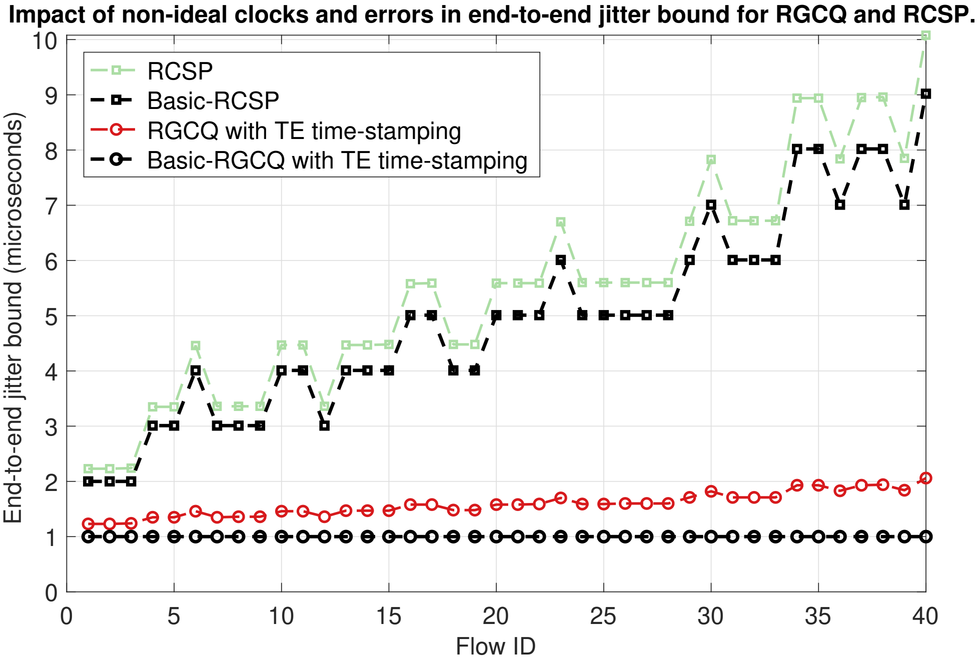


Figure 7.10: The end-to-end jitter bounds for RCSP and RGCQ with TE time-stamping using basic jitter computation and using theorems 7.1 and 7.2.

Fig. 7.10 shows the end-to-end jitter bounds of the flows for full deployment of RCSP and RGCQ with TE time-stamping. For each of the cases, the basic jitter computation only considers the jitter imposed by the tolerances of the dampers and ignore the impact of non-ideal clocks and errors in the computation of damper header. Fig. 7.10 also shows the true jitter bound for the case of RCSP, using Theorem 7.1, and for the case of RGCQ with TE time-stamping using Theorem 7.2. We see that non-ideal clocks and errors can increase jitter by 11% in the case of RCSP and 106% in the case of RGCQ with TE time-stamping. We also see that the TE time-stamping used with RGCQ can significantly reduce the end-to-end jitter comparing to the default time-stamping used with RCSP. Fig. 7.11 shows the end-to-end delay and jitter bounds of the flows when no damper is used and when there is a full deployment as

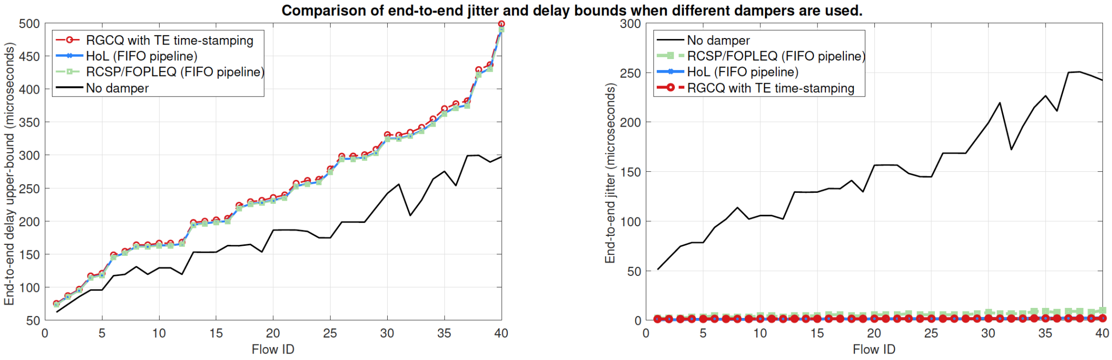


Figure 7.11: The end-to-end delay bounds (left) and jitter bounds (right) for full deployment of dampers and in the absence of dampers. For FOPLEQ and HoL dampers, we assume that all the elements are FIFO.

above. All switching fabrics are FIFO. We see that without damper, the delay upper-bound is smaller compared to full damper deployments; this is due to the line-shaping effect when computing the queuing delay bounds in the absence of dampers. However, as expected, the full deployment of dampers significantly reduces the jitter bounds. In this computation, the HoL damper provides quasi similar jitter bound as RGCQ with TE time-stamping and FOPLEQ gives the exact same jitter bound as RCSP as seen in Theorem 7.3.

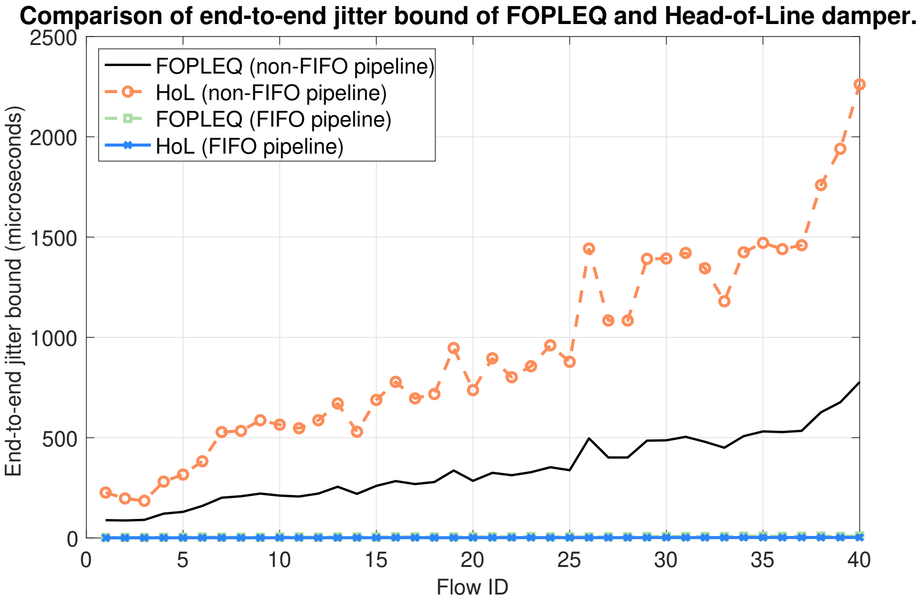


Figure 7.12: The end-to-end jitter bounds for FOPLEQ and head-of-line dampers when switching fabrics are FIFO or not.

Fig. 7.12 shows the end-to-end jitter bounds of the flows for FOPLEQ and HoL damper considering the switching fabrics are FIFO and are not FIFO. The figure shows that with FOPLEQ the jitter is significantly increased that is due to the jitter imposed by the output queuing, as seen in Theorem 7.5. It also shows that jitter bounds are worse in the case of HoL damper as discussed in Theorem 7.6.

7.8 Conclusion

We have presented a theory to compute delay and jitter bounds in a network that implements dampers with non-ideal clocks. We have shown that dampers without FIFO constraint can cause packet reordering even if all network elements are FIFO; re-sequencing dampers and head-of-line dampers avoid the problem; the former come with no jitter or delay penalty, and the latter with a small, quantified penalty. However, when a flow path contains non-FIFO elements, re-sequencing dampers and head-of-line dampers do not perform well.

7.9 Notation

Term	Description
$d^{\mathscr{H}_i}$	The delay measurement with clock $\mathscr{H}_i$
E_n	The actual release time of packet n from a damper
$\tilde{E}_n$	The theoretical eligibility time of packet n from a damper
H_n	The damper header of packet n
$\mathscr{H}_i$	The clock of a device i or the true time (TAI)
Q_n	The arrival time of packet n to a damper
$\alpha^{\mathscr{H}_i}$	The arrival curve of a flow seen with clock $\mathscr{H}_i$
δ_i	The delay upper-bound for JCS i
ϵ	An upper bound on the damper-header computation error
η	The clock timing-jitter bound
v_j	The delay-jitter bound of BDS j
$\underline{\pi}_j, \overline{\pi}_j$	The delay lower-bound and upper-bound for BDS j
ρ	The clock stability bound
$\phi^{\min}, \phi^{\max}$	The minimum/maximum processing times of HoL damper
ω	The clock synchronization time-error bound
$\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}}$	Damper tolerances
$w^{\downarrow}$	The lower pseudo-inverse of function w (Section 3.1.1)
$1_{\{C\}}$	It is equal to 1 when the condition C is true and is equal to 0 otherwise
$\lfloor x \rfloor$	The floor of x
$[x]^+$	$\max(0, x)$

8 Conclusion and Future Work

What we know is a drop, what we don't know is an ocean.
— Isaac Newton

In this thesis, in order to obtain bounds on delay, backlog, and delay jitter, we have provided a network-calculus-based analysis for asynchronous mechanisms in time-sensitive networks.

The first set of our results is novel delay-bounds based on network calculus, as presented in Chapter 4. Our obtained bounds supersede the classic network-calculus bounds for scheduling mechanisms and can be used in the analysis of time-sensitive networks. The bounds were obtained by decomposing the delay into queuing and transmission delays; this enabled us to exploit the information on the physical-line rate, hence the improvements in the delay bounds. Using this method of proof, together with a novel modelling of a packet-level arrival curve with g-regularity and bit-level arrival curve, we obtain a delay bound for flows with packet-level arrival curves. This delay bound improves the state-of-the-art approach that is obtained by only arrival-curve modelling of a packet-level arrival curve. For a proof of concept for our improved delay bounds, in Chapter 5, we have studied an asynchronous configuration of TSN with CBSs and ATSs, for which we obtained end-to-end delay and backlog for class A and class B traffic. The bounds are based on the obtainment of CBS service curves for classes A and B.

The second set of our results is the analysis of re-sequencing buffers that are used to address the in-order packet-delivery requirement in time-sensitive networks. Our analysis, presented in Chapter 6, is the first one performed on this topic. We have shown that the RTO of a flow determines the timeout value, and that the RBO is used to quantify the size of a re-sequencing buffer. We have proven that when the network can be assumed lossless (due to a proper buffer allocation in the systems and redundancy mechanisms), re-sequencing does not modify worst-case delay or delay jitter. However, if the network is lossy and its performance is of interest, the re-sequencing comes with a penalty, on the worst-case delay and jitter. The penalty is equal at least to the RTO of the flow being in-ordered. Furthermore, we have provided a calculus for capturing the RTO and RBO of a flow as the main ingredients of our analysis. We have shown that the RTO can be very large, even though the RTO of every individual non-order-preserving element is very small, due to amplification by downstream jitter.

Finally, we have studied dampers that are asynchronous mechanisms for providing low end-to-end delay jitter. The idea of dampers was proposed in the 1990s and recently caught attention as a solution to be used in time-sensitive networks. Prior to this thesis, a proper analysis of dampers was missing, i.e., in practice, unlike the literature assumption in the analysis, dampers operate with clocks that differ from the clocks of other network elements. This makes the analysis of dampers non-trivial, and we address this in Chapter 7. We have presented a theory for computing delay and jitter bounds in a network that implements dampers with non-ideal clocks. We have seen that dampers can be categorized into dampers with FIFO constraints and those without. We have shown that dampers without FIFO constraints can experience packet reordering, due to non-ideal clocks and to the errors in the damper-header computation, even if all the network elements are FIFO. This problem can be avoided by using re-sequencing dampers and head-of-line dampers: The re-sequencing dampers do not come with either jitter or delay penalties, and the head-of-line dampers come with only a small penalty. On the contrary, when a flow path contains non-FIFO elements, re-sequencing dampers and head-of-line dampers do not perform well, in the sense that they provide a worse jitter guarantee compared to dampers without FIFO constraints, and that they also preserve the wrong packet order that occurs in the non-FIFO elements.

Hereafter, we mention a number of open problems and directions for future studies.

(1) A valid question regarding our improved delay bounds in Chapter 4 is about how they can be used to provide good end-to-end bounds. When all systems implement dampers or regulators, the question is trivial, as all flows recreate their source constraints. However, in the absence of these mechanisms, the flow constraints are changed at intermediate systems due to the input-line shaping. This especially affects the packet-level arrival curves, as the shaping occurs at the bit level. Analyzing the delay bound by incorporating the input-line shaping effect with our results (that exploit the information on the output transmission-line) would be an interesting direction for future research.

(2) As mentioned in Chapter 2, there is also a set of synchronous mechanisms that we have not studied in this thesis and that can be combined with asynchronous ones (e.g., the considered TSN configuration in Chapter 5). An example of such a combination is a configuration of IRs, CBSs, TASs and SP to enable the transmission of time-triggered traffic. The coexistence of asynchronous and synchronous mechanisms adds more complexity to the computation of the delay, jitter, and the backlog bounds, and is a challenging research direction.

(3) We have mentioned that the placement of dampers comes with hardware costs. Also, in Chapter 7, we have shown that dampers can be partially deployed to reduce end-to-end jitter in time-sensitive networks. This inspires us with the idea of placing dampers at certain intermediate systems. The immediate question is how to choose these intermediate systems.

In other words, *"What is a proper placement of dampers in a network such that the flows meet their required performance guarantees?"*. A similar question regarding the IR placement was already addressed in [133].

A Appendix (Chapter 4)

A.1 Technical Prerequisites

Consider a function $f \in \mathscr{F}_{\text{inc}}$, where $\mathscr{F}_{\text{inc}}$ is the set of wide-sense increasing functions $w : [0,+\infty) \to [0,+\infty]$. Then by [124, Section 10.1][123],

$$(f^+)^\downarrow = f^\downarrow, \tag{A.1}$$

$$(f^\downarrow)^+ = f^\uparrow, \tag{A.2}$$

$$(f^\uparrow)^- = f^\downarrow, \tag{A.3}$$

$$(f^-)^\uparrow = f^\uparrow, \tag{A.4}$$

$$\forall y \in \mathbb{R}^+ \ : \ f(x) \leq y \implies x \leq f^\uparrow(y), \tag{A.5}$$

$$\forall y \in \mathbb{R}^+ \ : \ f(x) \geq y \implies x \geq f^\downarrow(y), \tag{A.6}$$

where f^+ is the right limit of the function f. Furthermore, by [124, Section 10.1][182], if f is right continuous:

$$f = (f^\downarrow)^\uparrow, \tag{A.7}$$

$$\forall w \in \mathscr{F}_{\text{inc}} \ : \ (f \circ w)^\downarrow(x) = (w^\downarrow \circ f^\downarrow)(x), \tag{A.8}$$

and if f is left continuous:

$$f = (f^\uparrow)^\downarrow, \tag{A.9}$$

$$\forall w \in \mathscr{F}_{\text{inc}} \ : \ (f \circ w)^\uparrow(x) = (w^\uparrow \circ f^\uparrow)(x). \tag{A.10}$$

Note that $\circ$ is the composition operator, i.e., $(f \circ w)(x) = f(w(x))$.

Lemma A.1. *Consider a left-continuous function $f \in \mathscr{F}_{\text{inc}}$. Then $\left((f^\downarrow)^\uparrow\right)^- = f$.*

Proof. By (A.3), we have:

$$\left(\left(f^{\downarrow}\right)^{\uparrow}\right)^{-} = \left(\left(\left(f^{\uparrow}\right)^{-}\right)^{\uparrow}\right)^{-}. \tag{A.11}$$

Then, by (A.4):

$$\left(\left(\left(f^{\uparrow}\right)^{-}\right)^{\uparrow}\right)^{-} = \left(\left(f^{\uparrow}\right)^{\uparrow}\right)^{-}. \tag{A.12}$$

Then, by (A.3):

$$\left(\left(f^{\uparrow}\right)^{\uparrow}\right)^{-} = \left(f^{\uparrow}\right)^{\downarrow} = f, \tag{A.13}$$

where the last equality is by (A.9) and left-continuity of f. □

Lemma A.2. *Let f be a wide-sense increasing and c-Lipschitz function. Then, for $x' \geq x$, we have:*

$$f^{\downarrow}\left(x'\right) - f^{\downarrow}(x) \geq \frac{x'-x}{c}. \tag{A.14}$$

Proof. According to the definition of Lipschitz continuity and because f is wide-sense increasing, we have for $t' \geq t$:

$$f\left(t'\right) - f(t) \geq c(t'-t). \tag{A.15}$$

Assume that $f^{\downarrow}\left(x'\right) = t'$ and $f^{\downarrow}(x) = t$. Due to Lipschitz continuity, f is continuous. Therefore, from (3.1), $t = f^{\downarrow}(x) = \inf\{s \geq 0 | f(s) \geq x\} = \sup\{s \geq 0 | f(s) < x\} = \sup\{s \geq 0 | f(s) \leq x\}$. Thus, $f(t) = x$. Similarly, we can show that $f\left(t'\right) = x'$. Therefore, we obtain

$$f^{\downarrow}\left(x'\right) - f^{\downarrow}(x) = t' - t \geq \frac{f\left(t'\right) - f(t)}{c} = \frac{x'-x}{c}, \tag{A.16}$$

which completes the proof. □

Lemma A.3. *A flow with fixed interval (τ, K) conforms to a packet-level arrival-curve α_{pkt} with*

$$\alpha_{\text{pkt}}(0) = 0, \;\; \alpha_{\text{pkt}}(t) = K\lceil\frac{t}{\tau}\rceil + K : t > 0. \tag{A.17}$$

Proof. First, since $N(s) - N(s) = 0; \forall s \geq 0$, by (4.4) $\alpha_{\text{pkt}}(0) = 0$.

Next we prove the statement for $t > 0$. For all $t > 0$, there exists some $0 \leq t' < \tau$ such that

$$t = i\tau + t', \;\; i = \lceil\frac{t}{\tau}\rceil. \tag{A.18}$$

Now, consider a time instant s. We cover the two cases $s \leq \theta$ and $s > \theta$ separately.

- $0 \leq s \leq \theta$. Then by (4.9) $N(s) = 0$. Therefore:

  $$N(s+t) - N(s) = N(s+t) \leq N(\theta + t). \tag{A.19}$$

  By (A.18):

  $$\begin{aligned} N(\theta + t) = N(\theta + i\tau + t') = \big[N(\theta + i\tau + t') - N(\theta + i\tau)\big] \\ + [N(\theta + i\tau) - N(\theta + (i-1)\tau)] + \cdots + [N(\theta + \tau) - N(\theta)] + N(\theta) \end{aligned}$$

  Using the definition of fixed interval constraint in (4.9), we have:

  $$N(\theta + t) \leq (i+1)K + N(\theta) = (i+1)K \tag{A.20}$$

  Since $i = \lceil \frac{t}{\tau} \rceil$, by (A.19) we have:

  $$N(s+t) - N(s) \leq K\lceil \frac{t}{\tau} \rceil + K. \tag{A.21}$$

- $s > \theta$. Then, there exists some $n \in \mathbb{N}$ such that:

  $$\theta + n\tau \leq s < \theta + (n+1)\tau. \tag{A.22}$$

  Therefore,

  $$N(s+t) - N(s) \leq N(\theta + (n+1)\tau + t) - N(\theta + n\tau). \tag{A.23}$$

  By (A.18), the above equation gives:

  $$\begin{aligned} N(s+t) - N(s) &\leq N(\theta + (n+1)\tau + i\tau + t') - N(\theta + n\tau) \\ &= \big[N(\theta + (n+i+1)\tau + t') - N(\theta + (n+i+1)\tau)\big] + \\ &\big[N(\theta + (n+i+1)\tau) - N(\theta + (n+i)\tau)\big] \\ &+ \cdots + \big[N(\theta + (n+1)\tau) - N(\theta + n\tau)\big]. \end{aligned} \tag{A.24}$$

  Since $i = \lceil \frac{t}{\tau} \rceil$, (A.24) gives:

  $$N(s+t) - N(s) \leq (i+1)K = K\lceil \frac{t}{\tau} \rceil + K. \tag{A.25}$$

By (A.21) and (A.21), for all $s \geq 0$:

$$N(s+t) - N(s) \leq K\lceil \frac{t}{\tau} \rceil + K = \alpha_{\text{pkt}}(t), \tag{A.26}$$

which proves the lemma. □

A.2 Proofs

A.2.1 Proof of Proposition 4.1

1) Since the flow has an arrival curve α, by (3.20) for any packet index m, n such that $m \leq n$, we have:

$$\sum_{i=m}^{n} l_i \leq \alpha^+(A_n - A_m). \tag{A.27}$$

By excluding the last packet from the left hand-side of the inequality, we have:

$$\sum_{i=m}^{n-1} l_i \leq \alpha^+(A_n - A_m) - l_n \leq \alpha^+(A_n - A_m) - L^{\min}. \tag{A.28}$$

We define $h(t) = [t - L^{\min}]^+$ and $x = \sum_{i=m}^{n-1} l_i$. Then, (A.28) can be rewritten as:

$$x \leq (h \circ \alpha^+)(A_n - A_m). \tag{A.29}$$

Applying (A.6) to (A.29):

$$A_n - A_m \geq (h \circ \alpha^+)^{\downarrow}(x). \tag{A.30}$$

Since h is continuous, by (A.8) we have:

$$A_n - A_m \geq ((\alpha^+)^{\downarrow} \circ h^{\downarrow})(x). \tag{A.31}$$

Applying (A.1), we have:

$$A_n - A_m \geq (\alpha^{\downarrow} \circ h^{\downarrow})(x). \tag{A.32}$$

Finally, by (3.1), we have $h^{\downarrow}(x) = x + L^{\min}, x > 0; h^{\downarrow}(0) = 0$. Then, for $x > 0$:

$$A_n - A_m \geq \alpha^{\downarrow}(x + L^{\min}) = g(x), \tag{A.33}$$

and for $x = 0$:

$$A_n - A_m \geq \alpha^{\downarrow}(0) = 0 = g(0). \tag{A.34}$$

2) By [[126], Lemma 6.2.8], $g^{\uparrow}(t) + L^{\max}$ is an arrival curve for the flow. Since, the left limit of an arrival curve is also an arrival curve for the flow, we have:

$$\alpha(t) = \lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} g^{\uparrow}(t - \varepsilon) + L^{\max} = \left(g^{\uparrow}\right)^{-}(t) + L^{\max} = g^{\downarrow}(t) + L^{\max}. \tag{A.35}$$

The last equality is by (A.3).

3) Assume a flow with arrival curve α and let us first apply item 1 and then item 2. By item 1, we can find a g-regularity constraint:

$$g(x) = \alpha^{\downarrow}(x + L^{\min}) = (\alpha^{\downarrow} \circ h)(x), \tag{A.36}$$

where $h(x) = x + L^{\min}$. Now, we apply item 2 to the obtained g-regularity constraint in (A.36) and derive an arrival curve α':

$$\alpha'(t) = g^{\downarrow}(t) + L^{\max} = (\alpha^{\downarrow} \circ h)^{\downarrow}(t) + L^{\max} = \left((\alpha^{\downarrow} \circ h)^{\uparrow}\right)^{-}(t) + L^{\max}, \tag{A.37}$$

The last equality is by (A.3). Due to left continuity of $\alpha^{\downarrow}$, by (A.10) we have:

$$\alpha'(t) = \left(\left(h^{\uparrow} \circ \left(\alpha^{\downarrow}\right)^{\uparrow}\right)\right)^{-}(t) + L^{\max} = \left[\left(\left(\alpha^{\downarrow}\right)^{\uparrow}\right)^{-}(t) - L^{\min}\right]^{+} + L^{\max}. \tag{A.38}$$

Note that $h^{\uparrow}(t) = [t - L^{\min}]^{+} = \max(t - L^{\min}, 0)$. Since α is left continuous and $\alpha(t) \geq L^{\max}$, by Lemma A.1 we have:

$$\alpha'(t) = \alpha(t) - L^{\min} + L^{\max}. \tag{A.39}$$

Eq. (A.39) shows that by applying item 1 and then item 2, the obtained arrival curve is not the same as the initial one, i.e., $\alpha \neq \alpha'$, except from the case that $L^{\max} = L^{\min}$ (when all packets have the same length).

Let us now examine the opposite direction. Assume a flow has g-regularity constraint. By applying item 2, we can find an arrival curve, α:

$$\alpha(t) = g^{\downarrow}(t) + L^{\max} = (h \circ g^{\downarrow})(t), \tag{A.40}$$

where $h(t) = t + L^{\max}$. Now, we apply item 1 to the obtained arrival curve (A.40) and derive a g'-regularity constraint:

$$g'(x) = \alpha^{\downarrow}(x + L^{\min}) = (h \circ g^{\downarrow})^{\downarrow}(x + L^{\min}) = \left((h \circ g^{\downarrow})^{\uparrow}\right)^{-}(x + L^{\min}). \tag{A.41}$$

By (A.10) and since $h^{\uparrow}(x) = \left[x - L^{\max}\right]^{+}$, we have:

$$g'(x) = \left((g^{\downarrow})^{\uparrow} \circ h^{\uparrow}\right)^{-}(x + L^{\min}) = \left((g^{\downarrow})^{\uparrow}\right)^{-}([x - L^{\max}]^{+} + L^{\min}). \tag{A.42}$$

Since g is left continuous, by Lemma A.1, we have:

$$g'(x) = g([x - L^{\max}]^{+} + L^{\min}). \tag{A.43}$$

The above equation shows that applying item 2 and then item 1 does not give the same g-regularity as the initial one, i.e., $g \neq g'$, except from the case that $L^{\max} = L^{\min}$ (when all packets have the same length).

A.2.2 Proof of Proposition 4.2

1) According to the min-plus representation of packet-level arrival-curve in Eq. (4.5), for any packets m, n with $m \le n$, we have:

$$n - m + 1 \le \alpha^{+}_{\text{pkt}}(E_n - E_m). \tag{A.44}$$

Now let us multiply both sides of the inequality by $L^{\max}$:

$$(n - m + 1)L^{\max} \le L^{\max}\alpha^{+}_{\text{pkt}}(E_n - E_m). \tag{A.45}$$

For all packet indices i, it holds that $l_i \le L^{\max}$. Thus,

$$\sum_{i=m}^{n} l_i \le L^{\max}\alpha^{+}_{\text{pkt}}(E_n - E_m). \tag{A.46}$$

According to (3.20), the flow conforms to an arrival curve $\alpha = L^{\max}\alpha_{\text{pkt}}$.

To obtain g-regularity, we use Eq. (A.45) as well as the fact that $l_i \le L^{\max}$ for all packets i, and we have:

$$\sum_{i=m}^{n-1} l_i + L^{\max} \le L^{\max}\alpha^{+}_{\text{pkt}}(E_n - E_m). \tag{A.47}$$

Now, we divide the both sides by $L^{\max}$ and set $x := \sum_{i=m}^{n-1} l_i$. Then,

$$\frac{x}{L^{\max}} + 1 \le \alpha^{+}_{\text{pkt}}(E_n - E_m). \tag{A.48}$$

Using (A.6) and then (A.1):

$$E_n - E_m \ge \alpha^{\downarrow}_{\text{pkt}}(\frac{x}{L^{\max}} + 1) = g(x). \tag{A.49}$$

2) First we show that we can derive (4.14) using (4.13). By item 1, the flow with packet-level arrival-curve α_{pkt} conforms to g-regularity with $g = \alpha^{\downarrow}_{\text{pkt}} \circ f$, where $f(x) = \frac{x}{L^{\max}} + 1$. The flow also conforms to a bit-level arrival-curve α' that is derived by applying item 2 of Proposition 4.1 to (4.13), i.e., $\alpha'(t) = g^{\downarrow}(t) + L^{\max}$; then,

$$\alpha'(t) = g^{\downarrow}(t) + L^{\max} = \left(\alpha^{\downarrow}_{\text{pkt}} \circ f\right)^{\downarrow}(t) + L^{\max} = \left(\left(\alpha^{\downarrow}_{\text{pkt}} \circ f\right)^{\uparrow}\right)^{-}(t) + L^{\max} \tag{A.50}$$

where the last equality is obtained by (A.3). Now, using (A.10) and left continuity of $\alpha^{\downarrow}_{\text{pkt}}$:

$$\alpha'(t) = \left(f^{\uparrow} \circ \left(\alpha^{\downarrow}_{\text{pkt}}\right)^{\uparrow}\right)^{-}(t) + L^{\max}, \tag{A.51}$$

Since $f^{\uparrow}(t) = [tL^{\max} - L^{\max}]^+$, we have:

$$\alpha'(t) = \left[L^{\max}\left(\left(\alpha^{\downarrow}_{\text{pkt}}\right)^{\uparrow}\right)^{-}(t) - L^{\max}\right]^{+} + L^{\max}. \tag{A.52}$$

Since α_{pkt} is left continuous, by Lemma A.1:

$$\alpha'(t) = L^{\max}\alpha_{\text{pkt}}(t) - L^{\max} + L^{\max}. \tag{A.53}$$

Note that $\alpha^{+}_{\text{pkt}}(t) \geq 1$. Thus $\alpha'(t) = \alpha(t)$ given by (4.14).

Second we show that (4.14) does not give (4.13). By item 1, the flow with packet-level arrival-curve α_{pkt} conforms to bit-level arrival-curve with $\alpha = f' \circ \alpha_{\text{pkt}}$, where $f'(t) = tL^{\max}$. In addition, the flow also conforms to a g'-regularity constraint that derives by applying item of Proposition 4.1 to (4.14):

$$g'(x) = \alpha^{\downarrow}(x + L^{\min}) = \left(f' \circ \alpha_{\text{pkt}}\right)^{\downarrow}(x + L^{\min}) = \left(\alpha^{\downarrow}_{\text{pkt}} \circ f'^{\downarrow}\right)(x + L^{\min}), \tag{A.54}$$

where the last equality is obtained by using (A.8) and continuity of f'. Now, since $f'^{\downarrow}(x) = \frac{x}{L^{\max}}$, we have:

$$g'(x) = \alpha^{\downarrow}_{\text{pkt}}\left(\frac{x + L^{\min}}{L^{\max}}\right). \tag{A.55}$$

When all packets are of the same size ($L^{\max} = L^{\min}$), $g' = g$. When $L^{\min} < L^{\max}$, since $\alpha^{\downarrow}_{\text{pkt}}$ is a wide-sense increasing function, we have $g' \leq g$ and $g' \neq g$, i.e. g' is a weaker constraint than g given in (4.13).

A.2.3 Proof of Lemma 4.1

Let n be the index of the packet of interest with length l_n. Using Lemma A.4, there exists an $m \leq n$ such that:

$$\beta(Q_n - A_m) \leq \sum_{k=m}^{n-1} l_k, \tag{A.56}$$

where Q_n is the beginning of transmission of packet n. Using (A.5), Eq. (A.56) gives:

$$Q_n - A_m \leq \beta^{\uparrow}\Big(\sum_{k=m}^{n-1} l_k\Big). \tag{A.57}$$

Therefore Q_n satisfies,

$$Q_n \leq \max_{m \leq n}\left\{A_m + \beta^{\uparrow}\Big(\sum_{k=m}^{n-1} l_k\Big)\right\}. \tag{A.58}$$

Since $\sum_{k=m}^{n-1} l_k \leq w(A_n - A_m)$, we have:

$$Q_n \leq \max_{m\leq n}\left\{A_m + \beta^{\uparrow}\Big(w(A_n - A_m)\Big)\right\}. \tag{A.59}$$

By defining $t \overset{\text{def}}{=} A_n - A_m \geq 0$, we further obtain,

$$Q_n - A_n \leq \sup_{t\geq 0}\left\{-t + \beta^{\uparrow}\left(w\left(t\right)\right)\right\}. \tag{A.60}$$

Applying (A.2) to (A.60):

$$Q_n - A_n \leq \sup_{t\geq 0}\left\{(\beta^{\downarrow})^{+}\left(w\left(t\right)\right) - t\right\}. \tag{A.61}$$

Next, we use Lemma A.5 and set $\beta^{\downarrow} = f$; therefore:

$$Q_n - A_n \leq \sup_{t\geq 0}\left\{\beta^{\downarrow}\left(w\left(t\right)\right) - t\right\} = h(w,\beta), \tag{A.62}$$

which completes the proof.

Lemma A.4. *If a FIFO system has (i) β as a service curve, and (ii) packetized input, then for every packet n, there exists a packet index $m \leq n$ such that,*

$$\beta(Q_n - A_m) \leq \sum_{k=m}^{n-1} l_k,$$

where l_k is the length of the k^{th} packet, Q_n is the start of transmission of packet n and A_m is the arrival time of packet m.

Proof. Let $I(t)$ be the number of bits that have arrived up to (excluding) time t and $O(t)$ be the number of bits that have been served up to (excluding) time t. Then by definition of service curve in Eq. (3.23), the system is said to offer the flow a service curve β if for any $t \geq 0$, there exists an $s \in [0, t]$ such that

$$O(t) \;\geq\; I(s) \;+\; \beta(t-s). \tag{A.63}$$

Let n be some packet index. We have $O(Q_n) = \sum_{i=1}^{n-1} l_i$, since Q_n is the time at which packet n starts being transmitted and, by the FIFO property, all packets before n have been served by that time.

Now apply (A.63) with $t = Q_n$. For the resulting s, let m be the smallest packet index such that, $s \leq A_m$ and thus $I(s) = \sum_{i=1}^{m-1} l_i$, with the convention that an empty sum is equal to 0 (which occurs when $m = 1$). Note that, at this point, we do not know if $n \leq m$ or if $n > m$ (we know that $s \leq Q_n$, but, this does not imply that $m \leq n$, for example it is quite possible that $A_{n+1} < Q_n$ since packets may wait in the queue). But, in any case, since $s \leq A_m$ and β is wide-sense increasing:

$$\beta(Q_n - s) \geq \beta(Q_n - A_m). \tag{A.64}$$

By using (A.64) and the expressions of $O(Q_n)$ and $I(s)$ in A.63, we obtain:

$$\sum_{i=1}^{n-1} l_i \geq \sum_{i=1}^{m-1} l_i + \beta(Q_n - A_m). \tag{A.65}$$

If $m > n$, we obtain $\beta(Q_n - A_m) < 0$, which is a contradiction; therefore, $m \leq n$. □

Lemma A.5. *If $f(.)$ is a wide-sense increasing function and $f^+(.)$ is its right-limit, then for any $R > 0$:*

$$\sup_{t\geq 0}\left(f^+(t) - Rt\right) = \sup_{t\geq 0}\left(f(t) - Rt\right). \tag{A.66}$$

Proof. Let $K = \sup_{t\geq 0}\left(f(t) - Rt\right)$ and $K' = \sup_{t\geq 0}\left(f^+(t) - Rt\right)$. We want to prove that $K = K'$. To do so, first we show that $K \leq K'$; and second that $K \geq K'$.

- $K \leq K'$: The function f is wide-sense increasing; therefore for any $t \geq 0$, we have:

  $$f(t) \leq f^+(t) \implies f(t) - Rt \leq f^+(t) - Rt. \tag{A.67}$$

  Using (A.67), it is trivially shown that $K \leq K'$.

- $K \geq K'$: The function f is wide-sense increasing; therefore for any $t \geq 0$ and $\varepsilon > 0$, we have $f^+(t) \leq f(t+\varepsilon)$; thus:

  $$\begin{aligned} f^+(t) - Rt &\leq f(t+\varepsilon) - Rt = f(t+\varepsilon) - R(t+\varepsilon) + R\varepsilon \\ &\leq \sup_{u\geq 0}\left(f(u) - R(u)\right) + R\varepsilon = K + R\varepsilon, \end{aligned} \tag{A.68}$$

  i.e., $K + R\varepsilon$ is an upper bound on $f^+(t) - Rt$. By definition, K' is the lowest such upper bound. Thus $K' \leq K + R\varepsilon$. This holds for any $\varepsilon > 0$, thus $K' \leq K$.

□

A.2.4 Proof of Lemma 4.2

Let us remind that l_k and A_k are the length and the arrival time of the k^{th} packet with $k = 1, 2...$ (Section 4.2). Let n be the index of the packet of interest belonging to flow 1 with length l_n, $l_n = l$. The sum of all packets can be split in two parts, one with packets belonging to flow 1 and the one with packets belonging to flow 2. Let $F(k)$ be the flow id of k^{th} packet, then for any $0 \le m \le n$, we have:

$$\sum_{k=m}^{n-1} l_k = \sum_{k=m}^{n-1} 1_{\{F(k)=1\}} l_k + \sum_{k=m}^{n-1} 1_{\{F(k)=2\}} l_k. \tag{A.69}$$

For flow 1 with g-regularity constraint, by applying (A.5) to (4.1), we have:

$$\sum_{k=m}^{n-1} 1_{\{F(k)=1\}} l_k \le g^{\uparrow}(A_n - A_m), \tag{A.70}$$

For flow 2 with bit-level arrival-curve, using (3.20) we have:

$$\sum_{k=m}^{n} 1_{\{F(k)=2\}} l_k \le \alpha^+(A_n - A_m). \tag{A.71}$$

Note that since packet n belongs to flow 1, $\sum_{k=m}^{n} 1_{\{F(k)=2\}} l_k = \sum_{k=m}^{n-1} 1_{\{F(k)=2\}} l_k$. Therefore, together with (A.70) and (A.71):

$$\sum_{k=m}^{n-1} 1_{\{F(k)=1\}} l_k + \sum_{k=m}^{n-1} 1_{\{F(k)=2\}} l_k \le g^{\uparrow}(A_n - A_m) + \alpha^+(A_n - A_m). \tag{A.72}$$

Using (A.72) in (A.69):

$$\sum_{k=m}^{n-1} l_k \le w(A_n - A_m), \tag{A.73}$$

where the function $w : \mathbb{R}^+ \to \mathbb{R}^+$ is $w = g^{\uparrow} + \alpha^+$. Then by Lemma 4.1 for packet n, we have:

$$Q_n - A_n \le h(w, \beta), \tag{A.74}$$

where Q_n is start of transmission of packet n. Since the transmission time for the packet of interest, n, is $D_n - Q_n = \frac{l}{c}$, the delay bound is:

$$D_n - A_n = D_n - Q_n + Q_n - A_n \le h(w, \beta) + \frac{l}{c}, \tag{A.75}$$

which concludes the proof for item 1. Now, since $l \le L_1^{\max}$, $h(w, \beta) + \frac{L_1^{\max}}{c}$ is a delay bound for flow 1 which completes the proof.

A.2.5 Proof of Theorem 4.2

Let us first define the function w as:

$$w(t) = \sum_{u=1}^{M} L_u^{\max} \alpha_{\text{pkt},u}(t), \tag{A.76}$$

The proof is in two steps: first, we construct a simulation trace; second, we verify its properties.

Step 1. We start the construction of a simulation trace.

(a) We determine the smallest time instant t' with the following property:

$$\beta^{\downarrow}\left(w(t')-L_1^{\max}\right)-t'=\sup_{t\geq 0}\left\{\beta^{\downarrow}\left(w(t)-L_1^{\max}\right)-t\right\}=h(w-L_1^{\max},\beta). \tag{A.77}$$

In fact, the time t' will be the time at which the packet of interest arrives at the system and experiences the worst-case delay.

(b) Now, we generate the packet sequence for the M flows. Flow 1 has $n_1=\alpha_{\text{pkt},1}{}^{+}(t')$ packets presented as a pair $(A^1,\mathscr{L}^1)$ where $A^1=(A^1_1,A^1_2,\ldots,A^1_{n_1})$ is the sequence of packet arrival times and $\mathscr{L}^1=(l^1_1,l^1_2,\ldots,l^1_{n_1})$ is the packet length sequence, defined for all $i\in\{1,\ldots,n_1\}$ by:

$$\begin{aligned} l^1_i &= L_1^{\max}, \\ A^1_i &= \gamma+\inf\{s\geq 0 \mid \alpha_{\text{pkt},1}(s)\geq i\}=\gamma+\alpha^{\downarrow}_{\text{pkt},1}(i), \end{aligned} \tag{A.78}$$

where $\gamma=t'-\alpha^{\downarrow}_{\text{pkt},1}(n_1)$. Lemma A.6 shows that $\alpha^{\downarrow}_{\text{pkt},1}(n_1)\leq t'$ and hence $\gamma\geq 0$. The aforementioned packet sequence indicates that the packets have maximum length and the packet arrival is greedy, starting at time γ, and the last packet (packet of interest) arrives at $A^1_{n_1}=t'$.

Any other flow f, $f\neq 1$, has $n_f=\alpha_{\text{pkt},f}{}^{+}(t')$ packets presented as a pair $(A^f,\mathscr{L}^f)$ where $A^f=(A^f_1,A^f_2,\ldots,A^f_{n_f})$ is the packet arrival sequence and $\mathscr{L}^f=(l^f_1,l^f_2,\ldots,l^f_{n_f})$ is the packet length sequence, that are defined for all $j\in\{1,\ldots,n_f\}$ as:

$$\begin{aligned} l^f_j &= L_f^{\max}, \\ A^f_j &= \inf\{s\geq 0 \mid \alpha_{\text{pkt},f}(s)\geq j\}=\alpha_{\text{pkt},f}{}^{\downarrow}(j). \end{aligned} \tag{A.79}$$

The aforementioned packet sequence indicate that the packets have maximum length and their arrival is greedy starting at time $A^f_1=\alpha_{\text{pkt},f}{}^{\downarrow}(1)=0$. With the above arrival construction, we have the cumulative input packet-count function as:

$$N_u(t)=\begin{cases}\sum_{i=1}^{n_u}1_{A^u_i<t} & t\leq t'\\ n_u & t>t'\end{cases},\ \forall u=1,2,\ldots,M. \tag{A.80}$$

Now let us merge the two packet sequences to express the total traffic. We define the pair $(A,\mathscr{L})$ with arrival sequence $A=(A_1,A_2,\ldots,A_n)$ and lengths sequence $\mathscr{L}=(l_1,l_2,\ldots,l_n)$ of total packets and $n=\sum_{k=1}^{M}n_k$, i.e., $(A,\mathscr{L})=\bigcup_{k=1}^{M}(A^k,\mathscr{L}^k)$. Lemma A.6 indicates that for any $f\neq 1$:

$$A^f_{n_f}\leq t'=A^1_{n_1}.$$

In the case $A^f_{n_f}=A^1_{n_1}=t'$, assume that the last packet of flow 1 is enqueued after the last packet of other flows; hence, $A_n=A^1_{n_1}=t'$ and $l_n=L_1^{\max}$.

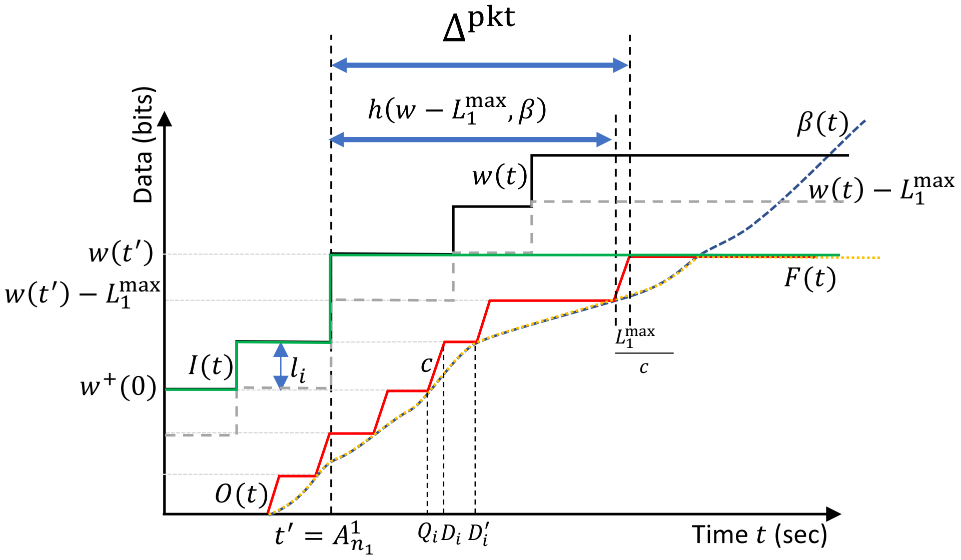


Figure A.1: The execution trace used in the proof of Theorem 4.2. The delay of the packet with length $L_1^{\max}$ that arrives at time t' is Δ^{pkt}.

The cumulative input function, $I(t)$ is shown as the green line in Fig. A.1 that is obtained as:

$$I(t) = \sum_{u=1}^{M} L_k^{\max} N_u(t) = \begin{cases} \sum_{k=1}^{\infty} L_k^{\max} 1_{\{A_k < t\}} & t \le t' \\ \sum_{u=1}^{M} n_u L_u^{\max} = w^+(t) & t > t' \end{cases}$$

(c) For the output, we first construct the fluid output curve $F(t)$ (orange dotted-line in Fig. A.1) given by

$$F(t) = \inf_{0 \le s \le t} \left\{ I(s) + \beta(t-s) \right\}, \tag{A.81}$$

so that the service curve property would be automatically satisfied if we would let the output cumulative function be $O(t) = F(t)$. However, we cannot take $O(t) = F(t)$ because $F(t)$ does not satisfy the condition that packet transmission is at rate c. In order to obtain the output function $O(t)$, we first observe the start and end of transmission time of a packet l_i, i.e., Q_i and D_i' as:

$$\begin{aligned} D_i' &= \inf\left\{s \ge 0 : F(s) \ge I^+(A_i)\right\} = F^{\downarrow}\left(I^+(A_i)\right), \\ Q_i &= \inf\left\{s \ge 0 : F(s) \ge I^+(A_i) - l_i\right\} = F^{\downarrow}\left(I^+(A_i) - l_i\right). \end{aligned}$$

In the above definitions, we have $D_i' = Q_{i+1}$ as $I^+(A_{i+1}) = I^+(A_i) + l_i$ by definition of I. For the output function $O(t)$, we keep the same time Q_i for the start of transmission of packet i, but, we let the transmission finish at time $D_i = Q_i + \frac{l_i}{c}$. Observe that:

$$D_i' - D_i = F^{\downarrow}\left(I^+(A_i)\right) - \left(F^{\downarrow}\left(I^+(A_i) - l_i\right) + \frac{l_i}{c}\right). \tag{A.82}$$

By Lemma A.7, $F(t)$ is c-Lipschitz. Then using Lemma A.2,

$$D_i' - D_i \geq \frac{l_i}{c} - \frac{l_i}{c} = 0. \tag{A.83}$$

Then, more precisely $\forall i = 1, \dots, n_1 + n_2$ and $\forall t \in [Q_i, D_i']$, $O(t)$ is:

$$O(t) = \begin{cases} c(t - Q_i) + F(Q_i) & Q_i \leq t < D_i, \\ F(Q_i) + l_i & D_i \leq t \leq D_i'. \end{cases} \tag{A.84}$$

$O(t)$ is shown with red line in Fig. A.1.

Step 2. We verify that all requirements in the theorem are satisfied. First we show that the service curve property holds; to do this, since F satisfies the service curve property, it is sufficient to show that $O(t) \geq F(t) : \forall t \in [Q_i, D_i']$ for any $i = 1, \dots, n$. Using Lipschitz continuity of F for $t \in [Q_i, D_i)$,

$$F(t) \leq F(Q_i) + c(t - Q_i) = O(t). \tag{A.85}$$

For $t \in [D_i, D_i']$, by construction:

$$F(t) \leq F(Q_i) + l_i = O(t). \tag{A.86}$$

The above equations imply $O(t) \geq F(t), \forall t \geq 0$; therefore, the service curve property is satisfied when the output is $O(t)$.

Moreover, by construction the system is FIFO, the input is packetized and packet transmission occurs at rate c. We need to prove that the input conforms to the packet-level arrival-curve. For any flow f, $f \neq 1$, consider two time instants $s, t \geq 0$, $s \leq t$.

If $t = s = 0$, then $N_f(t) - N_f(s) = \alpha_{\text{pkt},f}(t - s) = 0$.

If $0 < t \leq t'$, there exist a packet index m' where $t \in (A^f_{m'}, A^f_{m'+1}]$. Then by Lemma A.8 with $k = 0$, we have $m' = \alpha_{\text{pkt},f}(t)$. Now if $s = 0$, we have

$$N_f(t) - N_f(s) = N_f(A^f_{m'+1}) - N_f(0) = m' = \alpha_{\text{pkt},f}(t); \tag{A.87}$$

otherwise, there exists a packet index m ($m \leq m'$), where $s \in (A^f_m, A^f_{m+1}]$. Then:

$$N_f(t) - N_f(s) = N_f(A^f_{m'+1}) - N_f(A^f_{m+1}) = m' - m. \tag{A.88}$$

By Lemma A.8 with $k = 0$, $m' = \alpha_{\text{pkt},f}(t)$ and $m = \alpha_{\text{pkt},f}(s)$. Therefore:

$$N_f(t) - N_f(s) = \alpha_{\text{pkt},f}(t) - \alpha_{\text{pkt},f}(s) \le \alpha_{\text{pkt},f}(t-s), \tag{A.89}$$

where the last inequality is due to sub-additivity of $\alpha_{\text{pkt},f}$.

If $t > t'$, by construction, $N_f(t) = n_f = \alpha^+_{\text{pkt},f}(t')$. Now, for $s = 0$, we have

$$N_f(t) - N_f(s) = n_f = {\alpha_{\text{pkt},f}}^+(t') \le \alpha_{\text{pkt},f}(t); \forall t > t'. \tag{A.90}$$

For $0 < s \le t'$, there exists a packet index m ($m \le m'$), where $s \in (A^f_m, A^f_{m+1}]$. Then by Lemma A.8 with $k = 0$, we have $m = \alpha_{\text{pkt}}(s)$. Therefore,

$$N_f(t) - N_f(s) = n_f - m = {\alpha_{\text{pkt},f}}^+(t') - \alpha_{\text{pkt},f}(s) \le \alpha_{\text{pkt},f}(t-s); \forall t > t', \tag{A.91}$$

where the last inequality is due to sub-additivity of $\alpha_{\text{pkt},f}$.

For $t' < s \le t$, $N_f(s) = n_f$. Then,

$$N_f(t) - N_f(s) = n_f - n_f = 0 \le \alpha_{\text{pkt},f}(t-s), \tag{A.92}$$

which shows flow f conforms to the packet-level arrival-curve.

For flow 1, similarly to the above computation and using Lemma A.8 with $k = \gamma$, we obtain

$$N_1(t) - N_1(s) \le \alpha_{\text{pkt}}(t-s); \forall s \le t, \forall t \ge 0, \tag{A.93}$$

which shows flow 1 conforms to packet-level arrival-curve.

Last, we show that packet n achieves the delay bound. We have for packet n, $A_n = t'$; then:

$$Q_n = F^{\downarrow}(I^+(t') - l_n) = F^{\downarrow}(w^+(t') - L_1^{\max}). \tag{A.94}$$

Furthermore, $D_n = Q_n + \frac{l_n}{c} = Q_n + \frac{L_1^{\max}}{c}$, therefore,

$$D_n - A_n = F^{\downarrow}(w^+(t') - L_1^{\max}) - t' + \frac{L_1^{\max}}{c}. \tag{A.95}$$

By definition of F in (A.81), we have $F \le \beta$. Then by [124, Lemma 10.1], $F^{\downarrow} \ge \beta^{\downarrow}$. Hence:

$$D_n - A_n \ge \beta^{\downarrow}(w^+(t') - L_1^{\max}) - t' + \frac{L_1^{\max}}{c} = h(w^+ - L_1^{\max}, \beta) + \frac{L_1^{\max}}{c} = \Delta^{\text{pkt}}. \tag{A.96}$$

By Theorem 4.1, we have $D_n - A_n \le \Delta^{\text{pkt}}$; then together with the above equation, we have $D_n - A_n = \Delta^{\text{pkt}}$.

Lemma A.6. *Consider a wide-sense increasing function $f : \mathbb{R}^+ \to \mathbb{Z}^+$, a positive real value t_0, a positive integer $n = f^+(t_0)$, and a value $x = f^{\downarrow}(n)$; then we have $x \le t_0$.*

Proof. We have:

$$x = f^{\downarrow}(n) = f^{\downarrow}(f^{+}(t_0)). \tag{A.97}$$

By (A.1), $f^{\downarrow} = (f^{+})^{\downarrow}$. Therefore, $x = (f^{+})^{\downarrow}(f^{+}(t_0))$; then using Property (P1) of [124, Chapter 10.1] with $F = f^{+}$, we have $(f^{+})^{\downarrow}(f^{+}(t_0)) \leq t_0$ which concludes the proof. □

Lemma A.7. *Consider two functions f, g where f is L-Lipschitz and $g \geq 0$. Then*

$$z(t) = \inf_{0\leq s\leq t}\{g(s) + f(t-s)\}$$

is L-Lipschitz.

Proof. Let us define the set of functions $w_s(t)$ with constant $s \in \mathbb{R}^+$, as:

$$w_s(t) = g(s) + f(t-s). \tag{A.98}$$

Then, $z(t) = \inf_{0\leq s\leq t}\{w_s(t)\}$. First we prove that w_s is L-Lipschitz for any $s \in \mathbb{R}^+$. For any $t_1, t_2 \in \mathbb{R}^+$ and $t_2 \geq t_1$:

$$\begin{aligned} |w_s(t_2) - w_s(t_1)| &= |g(s) + f(t_2 - s) - g(s) - f(t_1 - s)| \\ &= |f(t_2 - s) - f(t_1 - s)| \leq L|t_2 - t_1|. \end{aligned} \tag{A.99}$$

The last inequality is obtained as f is L-Lipschitz. Now we prove the lemma. By (A.99), we have for any $s \in \mathbb{R}^+$:

$$w_s(t_1) - L|t_2 - t_1| \leq w_s(t_2) \leq w_s(t_1) + L|t_2 - t_1|. \tag{A.100}$$

Using the left inequality, we have:

$$\forall s \in \mathbb{R}^+ : \inf_{0\leq u\leq t_1}\{w_u(t_1) - L|t_2 - t_1|\} \leq w_s(t_2). \tag{A.101}$$

that gives $z(t_1) - L|t_2 - t_1| \leq w_s(t_2)$; therefore,

$$z(t_1) - L|t_2 - t_1| \leq \inf_{0\leq s\leq t_2}\{w_s(t_2)\} = z(t_2) \tag{A.102}$$

Using the right inequality in (A.100), we have:

$$\inf_{0\leq s\leq t_2}\{w_s(t_2)\} \leq \inf_{0\leq s\leq t_2}\{w_s(t_2) + L|t_2 - t_1|\} \leq \inf_{0\leq s\leq t_1}\{w_s(t_2)\} + L|t_2 - t_1|, \tag{A.103}$$

that gives $z(t_2) \leq z(t_1) + L|t_2 - t_1|$. Then, together with (A.102):

$$z(t_1) - L|t_2 - t_1| \leq z(t_2) \leq z(t_1) + L|t_2 - t_1|. \tag{A.104}$$

Hence, $|z(t_2) - z(t_1)| \leq L|t_2 - t_1|$, which concludes the proof. □

Lemma A.8. *Consider a left-continuous function $\alpha : \mathbb{R}^+ \rightarrow \mathbb{N}$ and a constant $k \geq 0$. Let $A = (A_1, A_2, \dots)$ be a sequence where $A_i = k + \alpha^{\downarrow}(i)$. Assume a positive integer m and a time instant t, such that $t \in (A_m, A_{m+1}]$; then we have $m = \alpha(t-k)$.*

Proof. We prove $\alpha(t-k) \geq m$ and $\alpha(t-k) \leq m$. We have $t > A_m = k + \alpha^{\downarrow}(m)$; therefore, $t - k > \alpha^{\downarrow}(m)$. Then, by [182, Proposition 6], we have $\alpha(t-k) \geq m$.

Also, since $t \leq A_{m+1}$, for any $\varepsilon > 0$ we have:

$$t - \varepsilon < A_{m+1} = k + \alpha^{\downarrow}(m+1). \tag{A.105}$$

Therefore,

$$t - k - \varepsilon < \alpha^{\downarrow}(m+1). \tag{A.106}$$

Then, by [182, Proposition 6], $\alpha(t-k-\varepsilon) < m+1$. Hence:

$$\lim_{\varepsilon \to 0} \alpha(t-k-\varepsilon) < m+1. \tag{A.107}$$

As α is left-continuous, we have $\alpha(t-k-\varepsilon) = \alpha(t-k) < m+1$. Since $\alpha(t-k) \in \mathbb{N}$, $\alpha(t-k) \leq m$. ☐

A.2.6 Proof of Theorem 4.3

As discussed in Section 4.2.3, the flows can have sliding-interval and fixed-interval regulation constraints; let F and S be respectively the sets of flows with fixed interval and sliding interval. By (4.7), if a flow i has the sliding interval (τ_i, K_i) constraint, this is equivalent to have the following packet-level arrival-curve:

$$\alpha_{\text{pkt},i}(0) = 0,\ \alpha_{\text{pkt},i}(t) = K_i \lceil \frac{t}{\tau_i} \rceil,\ t > 0. \tag{A.108}$$

By (4.10), if a flow i has the fixed interval (τ_i, K_i), the constraint implies the following packet-level arrival-curve:

$$\alpha_{\text{pkt},j}(0) = 0,\ \alpha_{\text{pkt},j}(t) = K_j \lceil \frac{t}{\tau_j} \rceil + K_j,\ t > 0. \tag{A.109}$$

Then the delay bound obtain by Theorem 4.1 for flow 1 is:

$$\Delta^{\text{pkt}} = h\Big(\sum_{u=1}^{N} L_u^{\max} K_u \lceil \frac{t}{\tau_u} \rceil + \sum_{u=1}^{N} L_u^{\max} K_u 1_{\{u \in F\}} - L_1^{\max}, \beta\Big) + \frac{L_1^{\max}}{c}. \tag{A.110}$$

We want to construct a simulation trace where a packet of flow 1 experiences a delay arbitrarily close to D. To this end, for flows in S, we use (A.108) to construct the input packet sequence as it is equal to the sliding interval interpretation. For any other flow j with fixed interval (τ_j, K_j), we use the following packet-level arrival-curve:

$$\begin{aligned}
&\alpha^{\varepsilon}_{\text{pkt},j}(0) = 0, \\
&\alpha^{\varepsilon}_{\text{pkt},j}(t) = K_j \lceil \frac{[t-\varepsilon]^+}{\tau_j} \rceil + K_j : t > 0, \forall \varepsilon \in (0, \min_{u \in F}\{\tau_u\}).
\end{aligned} \tag{A.111}$$

Lemma A.9 shows that a greedy packet-sequence of α^{j}_{ε}, starting at $t_0 = \max_{u\in F}\{\tau_u\} - \varepsilon$, conforms to Fixed interval (τ_j, K_j).

Now, let:

$$\begin{aligned}
w_\varepsilon(t) &= \sum_{u=1}^{N} L_u^{\max} \alpha^{u}_{\text{pkt}}(t) 1_{\{u\in S\}} + \sum_{u=1}^{N} L_u^{\max} \alpha^{\varepsilon}_{\text{pkt},u}(t) 1_{\{u\in F\}} \\
&= \sum_{u=1}^{N} L_u^{\max} K_u \lceil \frac{t}{\tau_u} \rceil 1_{\{u\in S\}} + \sum_{u=1}^{N} L_u^{\max} K_u \lceil \frac{[t-\varepsilon]^+}{\tau_u} \rceil 1_{\{u\in F\}} + \sum_{u=1}^{N} K_u 1_{\{u\in F\}}.
\end{aligned} \tag{A.112}$$

The tightness scenario follows the same as Theorem 4.2 by generating a greedy packet-sequence for every flow using the packet-level arrival-curves in (A.111) for fixed-interval flows and (A.108) for sliding-interval flows. To create a feasible greedy packet-sequence, we shift the start of the simulation by t_0 (as mentioned earlier, this guarantees the existence of greedy packet-sequence for the fixed-interval flows). Therefore, a packet of a flow of interest, i.e., flow 1, experiences a delay of $d_\varepsilon = h(w_\varepsilon - L_1^{\max}, \beta) + \frac{L_1^{\max}}{c}$ as shown in (A.96). By definition of w_ε in (A.112), d_ε is dependent on ε. We show next that $d_\varepsilon \geq \Delta^{\text{pkt}} - \varepsilon$.

To this end, let us define the following auxiliary functions:

$$\begin{aligned}
f(t) &= \sum_{u=1}^{N} L_u^{\max} K_u \lceil \frac{t}{\tau_u} \rceil 1_{\{u\in F\}},\ t > 0;\ f(t) = 0, t \leq 0, \\
g(t) &= \sum_{u=1}^{N} L_u^{\max} K_u \lceil \frac{t}{\tau_u} \rceil 1_{\{u\in S\}} + \sum_{u=1}^{N} K_u 1_{\{u\in F\}} - L_1^{\max}.
\end{aligned} \tag{A.113}$$

Then we have

$$\begin{aligned}
w_\varepsilon(t) &= f(t-\varepsilon) + g(t) + L_1^{\max} \\
\Delta^{\text{pkt}} &= h(f+g, \beta) + \frac{L_1^{\max}}{c}.
\end{aligned} \tag{A.114}$$

By Lemma A.10, $h(w_\varepsilon - L_1^{\max}, \beta) \geq h(f+g, \beta) - \varepsilon$; therefore,

$$d_\varepsilon \geq h(f+g, \beta) + \frac{L_1^{\max}}{c} - \varepsilon = \Delta^{\text{pkt}} - \varepsilon. \tag{A.115}$$

Moreover, by Theorem 4.1, $d_\varepsilon \leq \Delta^{\text{pkt}}$; hence,

$$\Delta^{\text{pkt}} - \varepsilon \leq d_\varepsilon \leq \Delta^{\text{pkt}}.$$

Finally, when $\varepsilon \to 0$, d_ε gets arbitrary close to Δ^{pkt}.

Lemma A.9. *Consider the following packet-level arrival-curve:*

$$\begin{aligned}
&\alpha_{\text{pkt}}(0) = 0, \\
&\alpha_{\text{pkt}}(t) = K\lceil \frac{[t-\epsilon]^+}{\tau} \rceil + K : t > 0, \forall \epsilon \in (0,\tau).
\end{aligned} \tag{A.116}$$

Then, a greedy packet-sequence of α_{pkt} *at any time* $t_0 \geq \tau - \epsilon$*, conforms to the fixed-interval* (τ, K) *in* (4.9).

Proof. Consider the cumulative packet function N defined as:

$$N(0) = N(t_0) = 0, \;\; N(t) = \alpha_{\text{pkt}}(t - t_0), \; t > t_0. \tag{A.117}$$

By definition, the above function is greedy at time t_0. Now, in (4.9), let $\theta = t_0 - \tau + \epsilon$. Since $t_0 \geq \tau - \epsilon$, we have $0 \leq \theta < t_0$; therefore, $N(\theta) = 0$. Next, we show that the greedy packet-sequence conforms to the fixed-interval constraint for any time $\geq \theta$. For $i = 0$ in (4.9):

$$N(\theta + \tau) - N(\theta) = N(t_0 + \epsilon) - 0 = \alpha_{\text{pkt}}(\epsilon) = K, \tag{A.118}$$

and for $i \geq 1$:

$$\begin{aligned}
N(\theta + (i+1)\tau) - N(\theta + i\tau) &= N(t_0 + \epsilon + i\tau) - N(t_0 + \epsilon + (i-1)\tau) \\
&= \alpha_{\text{pkt}}(\epsilon + i\tau) - \alpha_{\text{pkt}}(\epsilon + (i-1)\tau) = (iK + K) - ((i-1)K + K) = K.
\end{aligned}$$

Therefore, $\forall i \in \mathbb{N}$:

$$N(\theta) = 0, \; N(\theta + (i+1)\tau) - N(\theta + i\tau) \leq K, \tag{A.119}$$

which shows there exists θ $(= t_0 - \tau + \epsilon)$ where the cumulative function N, as a greedy packet-sequence of α_{pkt} at t_0, conforms to the fixed interval (τ, K) constraint. □

Lemma A.10. *Consider the functions* $f, g, \beta \in \mathscr{F}_{\text{inc}}$. *Let*

$$f_\epsilon(t) = \begin{cases} f(t-\epsilon) & t \geq \epsilon, \\ 0 & t < \epsilon. \end{cases} \tag{A.120}$$

Then $h(f_\epsilon + g, \beta) \geq h(f + g, \beta) - \epsilon$.

Proof. We have,

$$\begin{aligned}
h(f_\epsilon + g, \beta) &= \sup_{t \geq 0} \{\beta^{\downarrow}\left(f_\epsilon(t) + g(t)\right) - t\} \\
&\geq \sup_{t \geq \epsilon} \{\beta^{\downarrow}\left(f_\epsilon(t) + g(t)\right) - t\} = \sup_{t \geq \epsilon} \{\beta^{\downarrow}\left(f(t-\epsilon) + g(t)\right) - t\}.
\end{aligned} \tag{A.121}$$

Since $g(.)$ is wide-sense increasing, $g(t) \geq g(t-\epsilon)$, $\forall t \geq \epsilon$. Therefore:

$$\begin{aligned} h(f_\epsilon + g, \beta) &\geq \sup_{t \geq \epsilon}\{\beta^\downarrow\left(f(t-\epsilon)+g(t-\epsilon)\right) - t\} = \sup_{s\geq 0}\{\beta^\downarrow\left(f(s)+g(s)\right) - s - \epsilon\} \\ &= \sup_{s\geq 0}\{\beta^\downarrow\left(f(s)+g(s)\right) - s\} - \epsilon = h(f+g,\beta) - \epsilon, \end{aligned} \tag{A.122}$$

which concludes the proof. □

A.2.7 Proof of Theorem 4.5

(i) Let us remind that l_k and A_k are the length and the arrival time of the k^{th} packet with $k = 1,2...$ (Section 4.2). Let n be the index of the packet of interest belonging to flow 1 with length l_n, $l_n = l$. The sum of all packets can be split in two parts, one with packets belonging to flow 1 and the one with packets belonging flow 2. Let $F(k)$ be the flow id of k^{th} packet, then:

$$\sum_{k=m}^{n-1} l_k = \sum_{k=m}^{n-1} 1_{\{F(k)=1\}} l_k + \sum_{k=m}^{n-1} 1_{\{F(k)=2\}} l_k. \tag{A.123}$$

The flow of interest 1 has a bit-level arrival-curve; using (3.20), for any $0 \leq m \leq n$, we can write:

$$\sum_{k=m}^{n} 1_{\{F(k)=1\}} l_k \leq \alpha^+(A_n - A_m). \tag{A.124}$$

By excluding the last packet from the left hand-side of (A.124) (note that $l_n = l$), we obtain:

$$\sum_{k=m}^{n-1} 1_{\{F(k)=1\}} l_k \leq \alpha^+(A_n - A_m) - l. \tag{A.125}$$

Similarly, flow 2 has bit-level arrival-curve; thus, using (3.20) for any $0 \leq m \leq n$, we can write:

$$\sum_{k=m}^{n-1} 1_{\{F(k)=2\}} l_k \leq \alpha'^+(A_n - A_m). \tag{A.126}$$

Note that since packet n belongs to flow 1, the above equation conforms the min-plus representation of bit-level arrival-curve in (3.20) for flow 2. Now, we sum up (A.125) and (A.126):

$$\sum_{k=m}^{n-1} 1_{\{F(k)=1\}} l_k + \sum_{k=m}^{n-1} 1_{\{F(k)=2\}} l_k \leq \alpha^+(A_n - A_m) + \alpha'^+(A_n - A_m) - l. \tag{A.127}$$

Using (A.127) in (A.123):

$$\sum_{k=m}^{n-1} l_k \leq w(A_n - A_m), \tag{A.128}$$

where the function $w = \alpha^+ + \alpha_2^+ - l$. Then by Lemma 4.1 for packet n, we have:

$$Q_n - A_n \leq h(w, \beta), \tag{A.129}$$

where Q_n is start of transmission of packet n. Since the transmission time for the packet of interest, n, is $D_n - Q_n = \frac{l}{c}$, the delay bound is:

$$D_n - A_n = D_n - Q_n + Q_n - A_n \leq h(w, \beta) + \frac{l}{c}, \tag{A.130}$$

which completes the proof.

(ii) We want to compute

$$\begin{aligned} \sup_{l \in [L_1^{\min}, L_1^{\max}]} \{\Delta^A(l)\} &= \sup_l \left[\sup_t \left(\beta^{\downarrow}(\alpha^+ + \alpha'^+ - l) - t \right) + \frac{l}{c} \right] \\ &= \sup_t \left[\sup_l \left(\beta^{\downarrow}(\alpha^+ + \alpha'^+ - l) + \frac{l}{c} \right) - t \right]. \end{aligned} \tag{A.131}$$

Since β is c-Lipschitz, by Lemma A.2, it satisfies:

$$\beta^{\downarrow}(\alpha^+ + \alpha'^+ - L_1^{\min}) - \beta^{\downarrow}(\alpha^+ + \alpha'^+ - l) \geq \frac{1}{c}(l - L_1^{\min}). \tag{A.132}$$

This gives,

$$\beta^{\downarrow}(\alpha^+ + \alpha'^+ - l) \leq \beta^{\downarrow}(\alpha^+ + \alpha'^+ - L_1^{\min}) - \frac{1}{c}(l - L_1^{\min}). \tag{A.133}$$

By using the last relation in (A.131), we obtain,

$$\begin{aligned} \sup_{l \in [L_1^{\min}, L_1^{\max}]} \{\Delta^A(l)\} &= \sup_t \left[\beta^{\downarrow}(\alpha^+ + \alpha'^+ - L_1^{\min}) - t + \frac{L_1^{\min}}{c} \right] \\ &= h\left(\alpha^+ + \alpha'^+ - L_1^{\min}, \beta\right) + \frac{L_1^{\min}}{c}. \end{aligned} \tag{A.134}$$

Since β is c-Lipschitz continuous and α, α' are left continuous, by [2, Theorem 5.6], we have:

$$h\left(\alpha^+ + \alpha'^+ - L_1^{\min}, \beta\right) = h\left(\alpha + \alpha' - L_1^{\min}, \beta\right), \tag{A.135}$$

which together with (A.134) completes the proof.

A.2.8 Proof of Proposition 4.3

From Proposition 4.2, any flow u conform to a bit-level arrival-curve $\alpha_u(t) = \alpha_{\text{pkt},u}(t) L_u^{\max}$. Then by Theorem 4.5, the delay bound for a packet with size l of flow 1 is:

$$\Delta^A(l) = h\left(\sum_{u=1}^{U} \alpha_{\text{pkt},u} L_u^{\max} - l, \beta \right) + \frac{l}{c}. \tag{A.136}$$

By [2, Proposition 5.12], h, is monotonically increasing with respect to its first argument; therefore, $\Delta^{pkt}(l) \leq \Delta^{A}(l)$. As a result, $\sup_l \Delta^{pkt}(l) \leq \sup_l \left\{\Delta^{A}(l)\right\}$, i.e., $\Delta^{pkt} \leq \Delta^{A}$. Note that, if $L_1^{\max} = L_1^{\min}$ (all packets have the same length), then $\Delta^{pkt} = \Delta^{A}$. For the general statement, we show a case that when $L_1^{\min} < L_1^{\max}$, the per-flow bound in Theorem 4.1 strictly improves Δ^{A}.

First, for the ease of presentation, let us define $w(t) = \sum_{u=1}^{U} \alpha_{\text{pkt},u} L_u^{\max}$. Next, assume a rate-latency service-curve $\beta(t) = R[t-T]^+$, $R < c$. Then, as $\beta^{\downarrow}(x) = T + \frac{x}{R}$, Theorem 4.1 gives:

$$\begin{aligned}\Delta^{pkt} &= \sup_{t\geq 0}\left\{\beta^{\downarrow}\left(w(t) - L_1^{\max}\right) - t\right\} + \frac{L_1^{\max}}{c} = \sup_{t\geq 0}\left\{T + \frac{w(t) - L_1^{\max}}{R} - t\right\} + \frac{L_1^{\max}}{c}\\ &= T - L_1^{\max}\left(\frac{1}{R} - \frac{1}{c}\right) + \sup_{t\geq 0}\left\{\frac{w(t)}{R} - t\right\}. \end{aligned} \tag{A.137}$$

Using the derived bit-level arrival-curves of flows 1 and 2, Theorem 4.5 gives:

$$\begin{aligned}\Delta^{\mathrm{A}} &= \sup_{t\geq 0}\left\{\beta^{\downarrow}\left(w(t) - L_1^{\min}\right) - t\right\} + \frac{L_1^{\min}}{c} = \sup_{t\geq 0}\left\{T + \frac{w(t) - L_1^{\min}}{R} - t\right\} + \frac{L_1^{\min}}{c}\\ &= T - L_1^{\min}\left(\frac{1}{R} - \frac{1}{c}\right) + \sup_{t\geq 0}\left\{\frac{w(t)}{R} - t\right\}. \end{aligned} \tag{A.138}$$

By (A.137) and (A.138):

$$\Delta^{pkt} - \Delta^{\mathrm{A}} = -(L_1^{\max} - L_1^{\min})\left(\frac{1}{R} - \frac{1}{c}\right) < 0, \tag{A.139}$$

as $L_1^{\min} < L_1^{\max}$ and $R < c$.

A.2.9 Proof of Proposition 4.4

(i) Theorem 4.5 gives:

$$\Delta^{\mathrm{A}}(l) = h(\alpha^{+} + \alpha'^{+} - l, \beta) + \frac{l}{c}. \tag{A.140}$$

From item 1 of Proposition 4.1, $g_1(x) = \alpha^{\downarrow}(x + L_1^{\min})$ and $g_2(x) = \alpha'^{\downarrow}(x + L_2^{\min})$. By (A.1), we have:

$$g_1(x) = (\alpha^{+})^{\downarrow}(x + L_1^{\min}),\ g_2(x) = (\alpha'^{+})^{\downarrow}(x + L_2^{\min}). \tag{A.141}$$

Since $(\alpha^{+})^{\downarrow}$ and $(\alpha'^{+})^{\downarrow}$ are left continuous, by (A.10):

$$\begin{aligned} g_1^{\uparrow}(t) &= [((\alpha^{+})^{\downarrow})^{\uparrow}(t) - L_1^{\min}]^{+},\\ g_2^{\uparrow}(t) &= [((\alpha'^{+})^{\downarrow})^{\uparrow}(t) - L_2^{\min}]^{+}. \end{aligned} \tag{A.142}$$

As α^{+} and α'^{+} are right continuous and respectively larger than or equal to $L_1^{\max}$ and $L_2^{\max}$, by

(A.7) we have:

$$g_1^{\uparrow}(t) = \alpha^+(t) - L_1^{\min},\ g_2^{\uparrow}(t) = \alpha'^+(t) - L_2^{\min}. \tag{A.143}$$

Then, by applying Theorem 4.4, we obtain

$$\Delta^{\mathrm{G}}(l) = h(\alpha^+ + \alpha'^+ - L_1^{\min} - L_2^{\min} + L_2^{\max}, \beta) + \frac{l}{c}. \tag{A.144}$$

Let us compare (A.140) and (A.144). Since $l \ge L_1^{\min}$ and $L_2^{\max} \ge L_2^{\min}$, we have:

$$\alpha^+ + \alpha'^+ - l \le \alpha^+ + \alpha'^+ - L_1^{\min} - L_2^{\min} + L_2^{\max}. \tag{A.145}$$

By [2, Proposition 5.12], h, is monotonically increasing with respect to its first argument; hence, $\Delta^A(l) \le \Delta^G(l)$.

Since $\Delta^A(l) \le \Delta^G(l)$ holds for all packet sizes l, it also holds that $\sup_l \Delta^A(l) \le \sup_l \Delta^G(l)$, i.e., $\Delta^A \le \Delta^G$. If $L_1^{\max} = L_1^{\min}$ and $L_2^{\max} = L_2^{\min}$ (the packets of each flow have the same length), then $\Delta^A = \Delta^G$. For the general statement, we show a case that when $L_1^{\min} < L_1^{\max}$ or $L_2^{\min} < L_2^{\max}$, we have $\Delta^A < \Delta^{\mathrm{G}}$.

First, for the ease of presentation, let us define $w(t) = \alpha(t) + \alpha'(t)$. Next, assume a rate-latency service-curve $\beta(t) = R[t - T]^+$, $R < c$. Then, as $\beta^{\downarrow}(x) = T + \frac{x}{R}$, Theorem 4.5 gives:

$$\begin{aligned}\Delta^{\mathrm{A}} &= \sup_{t\ge 0}\left\{\beta^{\downarrow}\left(w(t) - L_1^{\min}\right) - t\right\} + \frac{L_1^{\min}}{c} = \sup_{t\ge 0}\left\{T + \frac{w(t) - L_1^{\min}}{R} - t\right\} + \frac{L_1^{\min}}{c}\\ &= T - \frac{L_1^{\min}}{R} + \sup_{t\ge 0}\left\{\frac{w(t)}{R} - t\right\} + \frac{L_1^{\min}}{c}.\end{aligned} \tag{A.146}$$

Using the derived g-regularity constraints of flows 1 and 2, Theorem 4.4 gives:

$$\begin{aligned}\Delta^{\mathrm{G}} &= \sup_{t\ge 0}\left\{\beta^{\downarrow}\left(w(t) - L_1^{\min} - L_2^{\min} + L_2^{\max}\right) - t\right\} + \frac{L_1^{\max}}{c}\\ &= \sup_{t\ge 0}\left\{T + \frac{w(t) - L_1^{\min} - L_2^{\min} + L_2^{\max}}{R} - t\right\} + \frac{L_1^{\max}}{c}\\ &= T - \frac{L_1^{\min}}{R} + \sup_{t\ge 0}\left\{\frac{w(t)}{R} - t\right\} + \frac{L_2^{\max} - L_2^{\min}}{R} + \frac{L_1^{\max}}{c}.\end{aligned} \tag{A.147}$$

By (A.146) and (A.147):

$$\Delta^{\mathrm{A}} - \Delta^{\mathrm{G}} = -\frac{L_2^{\max} - L_2^{\min}}{R} - \frac{L_1^{\max} - L_1^{\min}}{c} < 0, \tag{A.148}$$

as $L_1^{\min} < L_1^{\max}$ or $L_2^{\min} < L_2^{\max}$.

(ii) Theorem 4.4 gives:

$$\Delta^{\mathrm{G}}(l) = h(g_1^{\uparrow} + g_2^{\uparrow} + L_2^{\max}, \beta) + \frac{l}{c}. \tag{A.149}$$

From item 2 of Proposition 4.1, we obtain $\alpha(t) = g_1^{\downarrow}(t) + L_1^{\max}$ and $\alpha'(t) = g_2^{\downarrow}(t) + L_2^{\max}$. Using Theorem 4.5, we have:

$$\begin{aligned}\Delta^{\mathrm{A}}(l) &= h((g_1^{\downarrow})^{+} + (g_2^{\downarrow})^{+} + L_2^{\max} + L_1^{\max} - l, \beta) + \frac{l}{c} \\ &= h(g_1^{\uparrow} + g_2^{\uparrow} + L_2^{\max} + L_1^{\max} - l, \beta) + \frac{l}{c}.\end{aligned} \tag{A.150}$$

Note that by (A.2), $(g_1^{\downarrow})^{+} = g_1^{\uparrow}$ and $(g_2^{\downarrow})^{+} = g_2^{\uparrow}$. Similarly to the proof of (i), due to monotony of h with respect to its first argument, $\Delta^{\mathrm{G}}(l) \leq \Delta^{\mathrm{A}}(l)$.

Since $\Delta^{G}(l) \leq \Delta^{A}(l)$ holds for all packet sizes l, it also holds that $\sup_l \Delta^{G}(l) \leq \sup_l \Delta^{A}(l)$, i.e., $\Delta^{G} \leq \Delta^{A}$. If $L_1^{\max} = L_1^{\min}$ (the packets of the flow if interest have the same length), then $\Delta^{A} = \Delta^{G}$. For the general statement, similarly to the proof of item (i), we show a case that when $L_1^{\min} < L_1^{\max}$, we have $\Delta^{A} < \Delta^{\mathrm{G}}$. Considering the same service curve $\beta(t) = R[t-T]^{+}$, $R < c$ as proof of item (1), we obtain:

$$\Delta^{\mathrm{G}} - \Delta^{\mathrm{A}} = -(L_1^{\max} - L_1^{\min})\left(\frac{1}{R} - \frac{1}{c}\right) < 0, \tag{A.151}$$

as $L_1^{\min} < L_1^{\max}$ and $R < c$.

A.3 Example of non c-Lipschitz service curve

Consider the following function (FIFO residual service curve [2]), where θ and R are fixed positive numbers:

$$\beta(t) = \begin{cases} 0 & \text{if } t \leq \theta, \\ Rt & \text{if } t > \theta. \end{cases}$$

It is not c-Lipschitz as it is not continuous at $t = \theta$. Considering the assumptions of item (i) of Theorem 4.5, the response time of a packet with size l of flow 1 is

$$\Delta^{\mathrm{A}}(l) = \sup_{t \geq 0}\left\{\beta^{\downarrow}\left(\alpha^{+}(t) + \alpha'^{+}(t) - l\right) - t\right\} + \frac{l}{c}. \tag{A.152}$$

and the delay bound for flow 1 is:

$$\Delta^{\mathrm{A}} = \sup_{l \in [L_1^{\min}, L_1^{\max}]} \left\{\Delta^{\mathrm{A}}(l)\right\} \tag{A.153}$$

Given that $\alpha^+(t) \geq L_1^{\max}$, $\alpha'^+(t) \geq L_2^{\max}$ and $l \leq L_1^{\max}$, we have

$$\beta^{\downarrow}\left(\alpha^+(t) + \alpha'^+(t) - l\right) = \max(\theta, \frac{\alpha^+(t) + \alpha'^+(t) - l}{R}) \tag{A.154}$$

Therefore,

$$\begin{aligned}
\Delta^{\mathrm{A}}(l) &= \sup_{t\geq 0}\left\{\max(\theta, \frac{\alpha^+(t) + \alpha'^+(t) - l}{R}) - t\right\} + \frac{l}{c} \\
&= \sup_{t\geq 0}\left\{\max(\theta - t, \frac{\alpha^+(t) + \alpha'^+(t) - l}{R} - t)\right\} + \frac{l}{c} \\
&= \max\left(\sup_{t\geq 0}\{\theta - t\}, \sup_{t\geq 0}\{\frac{\alpha^+(t) + \alpha'^+(t) - l}{R} - t\}\right) + \frac{l}{c} \\
&= \max\left(\theta + \frac{l}{c}, \psi - \frac{l}{R} + \frac{l}{c}\right),
\end{aligned} \tag{A.155}$$

with

$$\psi = \sup_{t\geq 0}\left\{\frac{\alpha^+(t) + \alpha'^+(t)}{R} - t\right\}.$$

After examining all cases and some algebra, we find that

- if $R(\psi - \theta) \leq (1 - \frac{R}{c})L_1^{\min} + \frac{R}{c}L_1^{\max}$ then

  $$\Delta^{\mathrm{A}} = \sup_{l\in[L_1^{\min}, L_1^{\max}]}\left\{\Delta^{\mathrm{A}}(l)\right\} = \theta + \frac{L_1^{\max}}{c}$$

  and the supremum is attained at $l = L_1^{\max}$;

- else

  $$\Delta^{\mathrm{A}} = \psi - \frac{L_1^{\min}}{R} + \frac{L_1^{\min}}{c},$$

  and the supremum is attained at $l = L_1^{\min}$ and not at $l = L_1^{\max}$.

Therefore, the supremum over $[L_1^{\min}, L_1^{\max}]$ to obtain Δ^{A} can be achieved either at $l = L_1^{\max}$ or $l = L_1^{\min}$, depending on the parameter values.

B Appendix (Chapter 5)

B.1 Proof of Theorem 5.1

Consider some fixed time $t \geq 0$ and priority x; we show that $V_x(t) \leq V_x^{\max}$. If $V_x(t) \leq 0$, since $V_x^{\max} \geq 0$, the result follows trivially. Therefore we consider only the case $V_x(t) > 0$. Define time instant $s = \sup\{u \in [0,t] : V_x(u) = 0\}$. Based on the definition of s, $V_x(u) \neq 0,\ \forall u \in (s,t]$. This implies no credit reset in $(s,t]$, i.e., $V_x(\cdot)$ is continuous during this interval. Therefore, $\forall u \in (s,t] : V_x(u) > 0$. Also, CDT either finishes a transmission at s or is not transmitting at s. Indeed, otherwise, since $V_x(s) = 0$, and the credit of x is frozen during the transmission of CDT, it would be true that $V_x(s^+) = 0$ and thus, $s \neq \sup\{u \in [0,t] : V_x(u) = 0\}$.

The class x cannot start a transmission at time s, otherwise, by rule R2 in Section 5.1.2, since $V_x(s) = 0$, its credit would decrease to negative values, which contradicts our assumption that $\forall u \in (s,t] : V_x(u) > 0$. Note that since the credit of class x is positive in $(s,t]$, its backlog is also positive in $(s,t]$.

Since $\forall u \in (s,t] : V_x(u) > 0$ and due to rule R1, a class with lower priority than x cannot start a transmission in $(s,t]$. However, in order to consider non-preemptive AVB and BE classes, we must account for the case that a lower priority class has initiated a transmission the latest at s and is still transmitting at s. To do so, we define the time instant t_0, with $s \leq t_0$, as the end of the transmission of the residual of a lower priority packet after time s. The latter is denoted by $l^{LO} \leq \bar{L}^x$. If there is no transmission of a lower priority packet, then $l^{LO} = 0$. Let d^{L0} be the aggregated time period that the credit is frozen within $[s, t_0]$. Then, $t_0 = s + d^{L0} + \frac{l^{LO}}{c}$. If $t_0 > t$, due to rule R3, the credit of class x increases with rate I^x (except during transmission of CDT), therefore $V_x(t) = V_x(s) + I^x(t - s - d^{LO})$. By definition, $V_x(s) = 0$, thus, $V_x(t) = I^x(t - s - d^{LO}) \leq l^x \frac{\bar{L}^x}{c}$ and it follows that $V_x(t) \leq V_x^{\max}$, which ends the proof in this case. Therefore, in the rest of the proof we assume $t_0 \leq t$.

The interval $[t_0, t]$ can be split into a sequence of sub-intervals during which class x alternates between non-transmission and transmission. Let $[t_0, t_1], [t_1, t_2], \ldots, [t_{n-1}, t_n]$ be such a sequence, with $t_0 \leq t_1 < \ldots < t_n = t$. We allow $t_0 = t_1$ as this makes it possible to assume that class x does not transmit in the first interval $[t_0, t_1]$ (i.e., if class x starts transmission at time t_0 we set $t_1 = t_0$). It follows that for even intervals $[t_k, t_{k+1}]$, with $k \in \{0, 2, 4, ..., 2\lfloor \frac{n}{2} \rfloor\}$, we have $\frac{d}{dt} V_i(u) \geq 0$, $\forall u \in (t_k, t_{k+1})$. Indeed, during non-transmission, the credit either increases or remains constant, by rules R3 and R4. Conversely, for the odd intervals $[t_k, t_{k+1}]$, with $k \in \{1, 3, 5, ..., 2\lceil \frac{n}{2} \rceil - 1\}$, we have $\frac{d}{dt} V_x(u) < 0$, $\forall u \in (t_k, t_{k+1})$.

Let us define d_k as the aggregated time period that the credit is frozen within the even interval $[t_k, t_{k+1}]$. Next, we study the credit variation for all classes, starting with the interval $[s, t_0]$, then following with even and odd intervals in $(t_0, t]$. In $[s, t_0]$:

- Each class $j < x$ gains credit if it has backlog or negative credit (rule R3), except if CDT transmits, i.e.,

$$V_j(t_0) - V_j(s) \leq I^j (t_0 - s). \tag{B.1}$$

  By summing up for all $j < x$, we have

$$\sum_{j=1}^{x-1} \Big(V_j(t_0) - V_j(s) \Big) \leq \sum_{j=1}^{x-1} I^j (t_0 - s). \tag{B.2}$$

- Class x gains credit because it has backlog, as explained above, except if CDT transmits, i.e.,

$$V_x(t_0) - V_x(s) = I^x (t_0 - s) - I^x d^{LO}, \tag{B.3}$$

  and since $V_x(s) = 0$ and $t_0 = s + d^{LO} + \frac{l^{LO}}{c}$, we get

$$V_x(t_0) = I^x (t_0 - s - d^{LO}) \leq I^x \frac{l^{LO}}{c} \leq I^x \frac{\bar{L}^x}{c}. \tag{B.4}$$

For the odd intervals, $[t_{2k-1}, t_{2k}]$, $(1 \leq k \leq \lfloor \frac{n}{2} \rfloor)$:

- Since the credit of class x reduces, the higher priority classes do not transmit within $[t_{2k-1}, t_{2k}]$ and $\forall j < x : V_j(t_{2k-1}) \leq 0$. They gain credit if they have positive backlog or negative credit, therefore

$$V_j(t_{2k}) - V_j(t_{2k-1}) \leq I^j (t_{2k} - t_{2k-1}). \tag{B.5}$$

  Summing them up for all $j < i$:

$$\sum_{j=1}^{x-1} \Big(V_j(t_{2k}) - V_j(t_{2k-1}) \Big) \leq \sum_{j=1}^{x-1} I^j (t_{2k} - t_{2k-1}). \tag{B.6}$$

- The credit of class x reduces due to transmission (R2):

$$V_i(t_{2k}) - V_i(t_{2k-1}) = S_i(t_{2k} - t_{2k-1}). \tag{B.7}$$

For the even intervals, $[t_{2k}, t_{2k+1}]$, $(0 \leq k \leq \lfloor \frac{n-1}{2} \rfloor)$:

- There exists an AVB class $j < x$ that transmits or all AVB and BE classes wait for CDT (for an aggregated time d_{2k}). Define $a_{j,2k}$ as the aggregated period of time that class j transmits packets in $[t_{2k}, t_{2k+1}]$. Then, by using $I^j - S^j = c$, we obtain

$$V_j(t_{2k+1}) - V_j(t_{2k}) \leq I^j(t_{2k+1} - t_{2k}) - c a_{j,2k} - I^j d_{2k}. \tag{B.8}$$

  Summing up for all $j < x$, and considering that $t_{2k+1} - t_{2k} = d_{2k} + \sum_{j=1}^{x-1} a_{j,2k}$, we obtain

$$\sum_{j=1}^{x-1} \Big(V_j(t_{2k+1}) - V_j(t_{2k})\Big) \leq -\Big(c - \sum_{j=1}^{x-1} I^j\Big)(t_{2k+1} - t_{2k}) + (c - \sum_{j=1}^{x-1} I^j) d_{2k}. \tag{B.9}$$

- The credit of class x increases or is frozen for an aggregated time d_{2k}, i.e.,

$$V_x(t_{2k+1}) - V_i(t_{2k}) = I^x(t_{2k+1} - t_{2k}) - I^x d_{2k}. \tag{B.10}$$

Next, we study the credit variation within $[t_0, t_n]$. First we assume that n is odd. By summing up the credit variations for all intervals and all classes $j < x$, we have

$$\sum_{j=1}^{x-1} \Bigg[\Big(V_j(t_1) - V_j(t_0)\Big) + \Big(V_j(t_2) - V_j(t_1)\Big) + \ldots + \Big(V_j(t_{n-1}) - V_j(t_{n-2})\Big) + \Big(V_j(t_n) - V_j(t_{n-1})\Big) \Bigg] \leq$$
$$-\Big(c - \sum_{j=1}^{x-1} I^j\Big)(t_1 - t_0) + \ldots + \sum_{j=1}^{x-1} I^j (t_{n-1} - t_{n-2}) - \Big(c - \sum_{j=1}^{x-1} I^j\Big)(t_n - t_{n-1}) + \Big(c - \sum_{j=1}^{x-1} I^j\Big) \sum_{k=0}^{k=\lfloor \frac{n}{2} \rfloor} d_{2k}. \tag{B.11}$$

Therefore, by setting $\alpha = (t_2 - t_1) + (t_4 - t_3) + \ldots + (t_{n-1} - t_{n-2})$ and $\Delta t = t_n - t_0 - \sum_{k=0}^{k=\lfloor \frac{n}{2} \rfloor} d_{2k}$, we can write,

$$\sum_{j=1}^{x-1} \Big(V_j(t_n) - V_j(t_0)\Big) \leq -(c - \sum_{j=1}^{x-1} I_j)\Delta t + c\alpha. \tag{B.12}$$

Next, by summing up the credit variations for all intervals for class x and considering $S^x = I^x - c$,

$$V_x(t_n) - V_x(t_0) = I^x(t_1 - t_0) + (I^x - c)(t_2 - t_1) + \ldots + I^x(t_n - t_{n-1}) - I^x \sum_{k=0}^{k=\lfloor \frac{n}{2} \rfloor} d_{2k} = I^x \Delta t - c\alpha. \tag{B.13}$$

By [152, Theorem 7]; for $x = 1 ... p$:

$$\sum_{j=1}^{x} V_j(t) \leq \frac{\bar{L}^x}{c} \sum_{j=1}^{x} I^j. \tag{B.14}$$

Hence for $t = t_0$, we obtain

$$\sum_{j=1}^{x-1} V_j(t_0) \leq -V_x(t_0) + \frac{\bar{L}^x}{c} I^x + \frac{\bar{L}^x}{c} \sum_{j=1}^{x-1} I^j. \tag{B.15}$$

We lower bound the left-hand side of Eq. (B.12) using the lower bound of Eq. (5.3) and the bound of Eq. (B.15); therefore,

$$V_x(t_0) - K \leq c\alpha - (c - \sum_{j=1}^{x-1} I^j)\Delta t, \tag{B.16}$$

where $K = -\sum_{j=1}^{x-1} L^j \frac{S^j}{c} + \frac{\bar{L}^x}{c} I^x + \frac{\bar{L}^x}{c} \sum_{j=1}^{x-1} I^j \geq 0$. Eq. (B.16) gives an upper bound on Δt, i.e.,

$$\Delta t \leq \frac{c\alpha + K - V_x(t_0)}{c - \sum_{j=1}^{x-1} I^j}. \tag{B.17}$$

By using Eq. (B.17) in Eq. (B.13), we obtain

$$\begin{aligned} V_x(t_n) &\leq I^x \left(\frac{c\alpha + K - V_x(t_0)}{c - \sum_{j=1}^{x-1} I^j} \right) - c\alpha + V_x(t_0) \\ &= I^x \left(\frac{K}{c - \sum_{j=1}^{x-1} I^j} \right) + V_x(t_0) \left(\frac{c - \sum_{j=1}^{x-1} I^j - I^x}{c - \sum_{j=1}^{x-1} I_j} \right) - c\alpha \left(\frac{c - \sum_{j=1}^{x-1} I^j - I^x}{c - \sum_{j=1}^{x-1} I^j} \right). \end{aligned} \tag{B.18}$$

Next, considering $\sum_{j=1}^{i} I^j < c$, and since by Eq. (B.4) $V_x(t_0) \leq \frac{\bar{L}^x}{c} I^x$, we obtain

$$V_x(t_n) \leq \frac{I^x}{c\left(c - \sum_{j=1}^{x-1} I^j\right)} \left(cK + \bar{L}^x \left(c - \sum_{j=1}^{x-1} I^j - I^x \right) \right). \tag{B.19}$$

By replacing the value of K, the credit of class x at time t_n, where n is odd, is upper bounded by $V_x^{\max}$ given in the statement. If n is even, then

$$V_x(t_n) = V_x(t_{n-1}) + S^x(t_n - t_{n-1}). \tag{B.20}$$

Since $S^x(t_n - t_{n-1}) \leq 0$, then $V_x(t_n) \leq V_x(t_{n-1})$. As n is even, $n-1$ is odd. We have already found a bound, i.e. $V_x^{\max}$, for t_k when k is odd. Since $t = t_n$, and n is either odd or even, it follows that $V_x(t) \leq V_x^{\max}$.

B.2 Proof of Theorem 5.2

Proof. In this proof, for the ease of presentation, the indices i and j are neglected from the parameters. For a given class $x \in \{A, B\}$, $V^x(u)$ stands for the amount of CBS credit at time u, which is less than maximum credit, namely $V^{x,\max}$. Also, let $N^x(u)$, $O^x(u)$ be the cumulative input, output traffic, respectively, for class x at time u and $O^H(u)$ the cumulative output CDT traffic. Given a time t, we define the time s, as follows,

$$s = \sup\left\{u \le t \mid V^x(u) = 0 \text{ and } O^x(u) = N^x(u)\right\}. \tag{B.21}$$

Since s is the last time that the backlog is zero, there is no credit reset in the interval $[s, t]$. In the rest of the proof, we use $\Delta t = t - s$ as the time interval over which the system is analyzed. For a given class x, we also define the time intervals Δt^{x-}, Δt^{x+}, and Δt^{x0}, as the aggregated time periods within Δt, over which the credit is decreasing (due to packet transmission of this class), the credit is increasing (due to transmission of BE traffic or AVB traffic except class x in backlog period of this class or credit recovery), and the credit is frozen (due to transmission of CDT flows), respectively. Then,

$$V^x(t) - V^x(s) = V^x(t) = \Delta t^{x+} I^x + \Delta t^{x-} S^x. \tag{B.22}$$

As for a class $x \in \{A, B\}$, $\Delta t = \Delta t^{x+} + \Delta t^{x-} + \Delta t^{x0}$, then,

$$\Delta t^{x-} = \frac{I^x \Delta t - V^x(t) - I^x \Delta t^{x0}}{I^x - S^x}. \tag{B.23}$$

The number of bits served during Δt for class $x \in \{A, B\}$ is $c\Delta t^{x-}$. By utilizing Eq. (B.23),

$$O^x(t) - O^x(s) = \frac{c}{I^x - S^x}(I^x \Delta t - V^x(t) - I^x \Delta t^{x0}). \tag{B.24}$$

Δt^{x0} is the aggregated transmission time periods of CDT flows during Δt; therefore,

$$\Delta t^{x0} = \frac{O^H(t) - O^H(s)}{c}. \tag{B.25}$$

The service curve offered to an aggregate of CDT flows in CBFS at time $t > s$ is $\beta^H(t) = c[t - \bar{L}/c]^+$, where $\bar{L} = \max(L^A_{ij}, L^B_{ij}, L^E_{ij})$. As the CDT queue has an LB arrival curve (denoted as $\alpha^H = rt + b$), the following equation holds for the output arrival curve of the CDT queue:

$$O^H(t) - O^H(s) \le (\alpha^H \oslash \beta^H)(\Delta t) = b + r(\Delta t + \frac{\bar{L}}{c}). \tag{B.26}$$

Therefore, from Eqs. (B.25) and (B.26),

$$\Delta t^{x0} \le \frac{b + r(\Delta t + \frac{\bar{L}}{c})}{c}. \tag{B.27}$$

From Eqs. (B.24) and (B.27), and knowing that $O^x(s) = N^x(s)$ (from the assumption for s):

$$O^x(t) - N^x(s) \geq \frac{I^x(c-r)}{I^x - S^x}\left(\Delta t - \frac{V^x(t)c}{I^x(c-r)} - \frac{b + \frac{r\bar{L}}{c}}{c-r}\right). \tag{B.28}$$

As $V^x(t) \leq V^{x,\max}$, and using Eq. (B.28), we define the following rate-latency service curve for class $x \in \{A, B\}$:

$$\beta^x(t) = \frac{I^x(c-r)}{I^x - S^x}\left[t - \frac{V^{x,\max}c}{I^x(c-r)} - \frac{b + \frac{r\bar{L}}{c}}{c-r}\right]^+. \tag{B.29}$$

Finally, we replace in the last equation the credit upper bound given in Theorem 5.1 for each class $x \in \{A, B\}$. To simplify, we $S^A = I^A - c$, hence:

$$V^{A,\max} = \bar{L}^A \frac{I^A}{c}, \tag{B.30}$$

$$V^{B,\max} = \frac{I^B}{c}\left(L^A - \frac{c\bar{L}^B}{S^A}\right), \tag{B.31}$$

and we obtain the equations in the statement of Theorem 5.2. □

B.3 Proof of Theorem 5.3

Proof. $H(f, i, j, k, x)$ is an upper bound on $E_n - D'_n$ for all packets n of flow f. Now

$$E_n - D'_n = (E_n - A_n) - (D'_n - A_n), \tag{B.32}$$

and

$$E_n - A_n \leq C(i, j, k, x). \tag{B.33}$$

Therefore, for a packet n of flow f:

$$E_n - D'_n \leq C(i, j, k, x) - \inf_{n \in N_f} (D'_n - A_n), \tag{B.34}$$

now

$$D'_n - A_n \geq \frac{M_f}{c_{ij}} + T_{ij}^{\text{out,min}} + T_{ij}^{\text{proc, min}}. \tag{B.35}$$

□

C Appendix (Chapter 6)

C.1 Re-sequencing Buffer Operation

A re-sequencing buffer stores the packets of a flow until the packets with smaller sequence numbers arrive; then it delivers them in the increasing order of their sequence numbers. A re-sequencing buffer has two parameters, a size in bytes, B, and a timeout value, T. It is described in terms of:

- Shared variables, that are manipulated by the code routines. These are 1) a list (buffer), buf, containing the packets that are waiting for the packets with smaller sequence number; 2) an integer, N, expressing the next sequence number that the buffer is expecting to receive
- Timers: The re-sequencing buffer sets a timer for each packet stored in the buffer. The object TimerList is the list of the timers for the packets. It has two functions, start(pid,$deadline$) and stop(pid). The former, starts a timer for the packet with sequence number pid with expiration time $deadline$. The latter, stops the timer for the packet with sequence number pid.
- Events, which trigger the execution of code routines. The events are packet arrival (Algorithm 1) and timeout (Algorithm 2). We assume that the execution of the code routines is serialized, namely, a code routine can start only after the code routine triggered by the previous event has completed (to avoid race conditions with shared variables).

When a new packet p arrives, the packet arrival code routine in Algorithm 1 is executed. If the packet sequence number $p.id$ is smaller than N, then the packet is considered invalid and is discarded (line 13). This is an error-case: packet N is expected, which means that packet $N-1$ was delivered. This packet is either a duplicate of packet $N-1$ or a packet with a smaller sequence number, and delivering it would violate in-order delivery.

Algorithm 1 Packet arrival event code routine

```
Input: packet $p$
Shared variables: $buf$ and $N$
if $p.id \geq N$ then                                  ▷ if TRUE, $p$ is a valid packet
    if $p.id > N$ then
        if $buf$.len() + $p.len \leq B$ then
            TimerList.start($p.id$,Time() + $T$)
            $buf$.enqueue($p$)
        else   discard($p$)                            ▷ ERROR, OVERFLOW
        end if
    else
        $N \leftarrow p.id + 1$
        release($p$)
        CHECK_BUFFER()
    end if
else   discard($p$)                                    ▷ ERROR, INVALID PACKET
end if
function CHECK_BUFFER(void)
    if $buf$.contains($N$) then
        $p \leftarrow buf$.dequeue($N$)
        $N \leftarrow p.id + 1$
        TimerList.stop($p.id$)
        release($p$)
        CHECK_BUFFER()
    end if
end function
```

Algorithm 2 Timeout event code routine

```
Input: packet id $pid$
Shared variables: $buf$ and $N$
while $N \leq pid$ do
    while !$buf$.contains($N$) do
        $N \leftarrow N + 1$
    end while
    $p' \leftarrow buf$.dequeue($N$)
    $N \leftarrow p'.id + 1$
    TimerList.stop($p'.id$)
    release($p'$)
end while
CHECK_BUFFER()
```

If $p.id > N$ (line 2), the packet should wait in the buffer for the packets with smaller sequence numbers to arrive. Then, it checks the current length of buffer, buf.len(); if addition of packet p with length of $p.len$ does not exceed the size of the buffer, B, (line 3) a timer for this packet starts, which expires at time Time()+T (line 4); the function Time() returns the current time of the buffer. Then, it enqueues the packet in the buffer (line 5). Otherwise, if the buffer does not have enough capacity, buffer overflow occurs and the packet is discarded (line 6).

If $p.id == N$ (line 8), the packet is the expected one. Then, N is incremented by 1 (line 9) and the packet is released (line 10). When packet p departs, the buffer should be checked to release the packets that were only waiting for packet p; this is done by a recursive function CHECK_BUFFER() (line 15). Accordingly, if the buffer contains a packet with a sequence number equal to the new value of N (line 16), the packet is dequeued from the buffer (line 17). Then, N is increased to the next sequence number of the flow; the corresponding timer is stopped; and then the packet is released (lines 18 to 20). It recalls the functions recursively (line 21) and the value of N is increased every time the function executed; the recursion continues until the buffer does not have a packet with sequence number equal to the last updated value of N; all the packets with sequence numbers less than the value of N are already released from the buffer.

When the timer for a packet with sequence number pid expires, the timeout code routine in Algorithm 2 is executed. In this condition, the packet with sequence number pid should be released. To provide in-order delivery, the buffer first releases all the packets in the buffer with sequence number less than or equal to pid in increasing order. To do so, the loop at line 1 is executed to iterate the buffer for the packets with sequence number less than pid. The lines 2 to 4, increment N to the smallest sequence number for a packet stored in the buffer; the function buf.contains(N), returns TRUE if a packet with sequence number N is in the buffer, otherwise, it returns FALSE. In line 5, the buffer contains a packet p' with sequence number equal to N; then, it is dequeued from the buffer; the value of N is incremented by 1; the timer for this packet is stopped; and it is released from the buffer (lines 5 to 8). Whenever a packet is released from the buffer, its corresponding timer is stopped. The loop in line 1 is executed at least once, because when the timeout event occurs for a packet with sequence number pid, it is already in the buffer; this implies $N \leq p.id$. The loop in line 2 is executed at the latest when $N = pid$. In line 10, the buffer is checked to release the packets that were waiting for packets with id pid and smaller; this is done by the function CHECK_BUFFER().

Observe that a packet is released if any one of the following conditions hold: 1) all packets with smaller sequence numbers are received, 2) its timer expires or 3) the timer of a received packet with a larger sequence number expires.

By construction, the re-sequencing buffer delivers the packets that it does not discard in increasing sequence numbers. Furthermore, a packet is discarded by the re-sequencing buffer either when the buffer is full or when the sequence number of the arriving packet is less than N. The latter occurs when the timeouts of packets are too early, compared to the lateness of misordered packets. Therefore, to avoid discarding packets, the re-sequencing buffer size and the timeout value should be large enough.

C.2 Proofs

C.2.1 Proof of Lemma 6.1

Let $A_1 \leq A_2 \leq ... \leq A_n ...$ be the arrival times of packets at $\mathscr{S}$; here the packet indices are in order of arrival (and are not necessarily equal to the sequence numbers). Let D_n be the departure time of the packet with index n; the sequence D_n need not be monotonic. Let R, R' be the cumulative arrival and departure functions[123], defined by $R(t) = \sum_{n=1}^{+\infty} l_n 1_{A_n < t}$ and $R'(t) = \sum_{n=1}^{+\infty} l_n 1_{D_n < t}$. Here, l_n is the length of packet n and we allow $t \leq 0$ (in which case $R(t) = R'(t) = 0$). Let $\varphi(t)$ be the function $\mathbb{R} \to \mathbb{R}$ defined by $\varphi(t) = 0$ is $t \leq 0$ and $\varphi(t) = 1$ if $t > 0$, so that

$$R(t) = \sum_{n=1}^{+\infty} l_n \varphi(t - A_n), \quad R'(t) = \sum_{n=1}^{+\infty} l_n \varphi(t - D_n) \tag{C.1}$$

Let $0 \leq s \leq t$; the arrival curve constraint at the input means that $R(t) - R(s) \leq \alpha(t - s)$. This equation continues to hold if s or t is negative, with the convention that $\alpha(t) = 0$ whenever $t \leq 0$. Therefore

$$\forall s, t \in \mathbb{R}, \quad R(t) - R(s) \leq \alpha(t - s) \tag{C.2}$$

Furthermore,

$$R'(t) - R'(s) = \sum_{n=1}^{+\infty} l_n \left(\varphi(t - D_n) - \varphi(s - D_n)\right) \tag{C.3}$$

Let $d^{\min}$ be the best-case delay of the flow, so that the worst-case delay is $\leq d^{\min} + V$. For every packet index n, we have

$$A_n + d^{\min} \leq D_n \leq A_n + d^{\min} + V \tag{C.4}$$

thus

$$t - D_n \leq t - A_n - d^{\min} \tag{C.5}$$

$$s - D_n \geq s - A_n - d^{\min} - V \tag{C.6}$$

and, since φ is wide-sense increasing

$$\varphi(t - D_n) - \varphi(s - D_n) \leq \varphi(t - A_n - d^{\min}) - \varphi(s - A_n - d^{\min} - V) \tag{C.7}$$

Combining with (C.3) and (C.2):

$$\begin{aligned} R'(t) - R'(s) &\leq \sum_{n=1}^{+\infty} l_n \Big(\varphi(t - A_n - d^{\min}) - \varphi(s - A_n - d^{\min} - V) \Big) \\ &= R(t - d^{\min}) - R(s - d^{\min} - V) \leq \alpha(t - s + V) \end{aligned} \tag{C.8}$$

□

C.2.2 Proof of Lemma 6.2

With the same notation as in the proof of Lemma 6.1, the backlog of this flow at time t is $B(t) = R(t) - R'(t)$, so that

$$B(t) = \sum_{n=1}^{+\infty} l_n \left(\varphi(t - A_n) - \varphi(t - D_n) \right) \tag{C.9}$$

We have $t - D_n \geq t - A_n - U$ and, since φ is wide-sense increasing:

$$B(t) \leq \sum_{n=1}^{+\infty} l_n \left(\varphi(t - A_n) - \varphi(t - A_n - U) \right) = R(t) - R(t - U) \leq \alpha(U) \tag{C.10}$$

where the last inequality is by (C.2). □

C.2.3 Proof of Theorem 6.1

(1) We prove (6.4) by induction on n.

Base case $n = 1$. According to the description of re-sequencing buffer in Section 6.2, the initial value of next expected sequence number is $N = 1$; therefore, packet 1 is never stored in the buffer.

- If packet 1 arrives at some time $E_1 \neq +\infty$. Then, there are two cases for the timers at time E_1 of packets with index larger than 1:
  - No timer has expired at E_1. Then, $N = 1$ and packet 1 is released immediately after arrival ($D_1 = E_1$). Since no timer was expired, for any packet p in the buffer, $p > 1$, we have $E_1 \leq E_p + T$. Thus, based on (6.5), $I_1 = E_1$ and finally $D_1 = I_1$ as required.
  - A timer has expired, say for packet index p, at E_1. Then, $N > 1$ and packet 1 is discarded ($D_1 = +\infty$). We have $E_1 > E_p + T$, (6.5) gives $I_1 = +\infty$ and finally $D_1 = I_1$ as required.
- Else packet 1 is lost ($E_1 = +\infty$). Then it is not released from the buffer ($D_1 = +\infty$). By (6.5), $I_1 = E_1 = +\infty$ and finally $D_1 = I_1$ as required.

Induction step. Suppose that (6.4) holds for all $i, i \leq n - 1$.

- If packet n arrives at some time $E_n \neq +\infty$. Then, there are two cases for the value of N:
  - $N \leq n$. No timer for a packet j in the buffer, $j > n$, has been expired (otherwise, the value of N would be increased to the index of the packet with expired timer, i.e., $N > n$). Then, we have $E_n \leq E_j + T$, therefore, $E_n \leq T + \min_{j \geq n}\{E_j\}$. Then, based on (6.5), $I_n = E_n$. Now, the release time of packet n depends on the status of packet $n-1$; there are two possible cases:
    - $N = n$. This implies that packet $n-1$ is already released, $D_{n-1} \leq E_n$. Therefore, packet n is released immediately on arrival ($D_n = E_n$). Then, based on (6.4) and (6.5), we also have:

$$I_n = E_n, \tag{C.11}$$

$$G_n = \min\left\{D_{n-1}, T + \min_{j \geq n}\{E_j\}\right\} \leq D_{n-1} \leq E_n, \tag{C.12}$$

      and then $\max(G_n, I_n) = E_n$, which shows that the right hand-side of (6.4) is D_n as required.
    - $N < n$. This implies that packet $n-1$ is not yet released, $D_{n-1} > E_n$. Therefore, packet n is stored in the buffer. Packet n is released when:
      - The timer for a packet with sequence number larger than or equal to n is expired before packet $n-1$ is released ($D_{n-1} \geq \min_{j \geq n}\{E_j\} + T$). Let us call p, where $p \geq n$, as the packet of which the timer is expired before the others ($T + \min_{j \geq n}\{E_j\} = T + E_p$). Then, any packet j in the buffer $j \leq p$ are released in-order (Algorithm 2, line 8); therefore, $D_n = T + E_p = T + \min_{j \geq n}\{E_j\}$. Then, based on (6.4) and (6.5), we also have:

$$I_n = E_n, \tag{C.13}$$

$$G_n = \min\left\{D_{n-1}, T + \min_{j \geq n}\{E_j\}\right\} = T + \min_{j \geq n}\{E_j\}, \tag{C.14}$$

        and then we have:

$$\max\{G_n, I_n\} = \max\{T + \min_{j \geq n}\{E_j\}, E_n\} = T + \min_{j \geq n}\{E_j\} = D_n, \tag{C.15}$$

        which shows (6.4).
      - Or when packet $n-1$ is released before any timer of packets with sequence number larger than or equal to n is expired ($D_{n-1} < T + \min_{j \geq n}\{E_j\}$). Then packet n is released immediately after packet $n-1$ is released ($D_n = D_{n-1}$). Then, based on (6.4) and (6.5), we also have:

$$G_n = \min\left\{D_{n-1}, T + \min_{j \geq n}\{E_j\}\right\} = D_{n-1}. \tag{C.16}$$

Then,

$$\max\{G_n, I_n\} = \max\{G_n, E_n\} = D_{n-1} = D_n, \tag{C.17}$$

which shows (6.4).

- $N > n$. Then, packet n is discarded ($D_n = +\infty$). Since $N > n$, a timer should has been expired for a packet p in the buffer, $p > n$ such that $E_n > E_p + T$; then, based on (6.4) and (6.5), we also have:

$$I_n = +\infty, \max\{G_n, I_n\} = +\infty = D_n, \tag{C.18}$$

which shows (6.4).

- Else, packet n is lost ($E_n = +\infty$). Then it is not released from the buffer ($D_n = +\infty$). By (6.5), $I_n = +\infty$ as well and thus $\max\{G_n, I_n\} = +\infty$, i.e. the right-handside of (6.4) is equal to D_n as required.

(2) Consider Fig. 6.1 and a received packet n. Due to (6.2), for any packet $j \geq n$, we have $E_n \leq E_j + \lambda$; therefore, $E_n \leq \min_{j\geq n}\{E_j\} + \lambda$. Since $\lambda \leq T$:

$$E_n \leq \min_{j\geq n}\{E_j\} + T \implies I_n = E_n, \tag{C.19}$$

which proves (6.7).

(3) The proof of item (3) is by induction on $n \geq 1$.

Base case $n = 1$. Then, by (6.7), $D_1 = E_1$ and the statement is trivially proven.

Induction step. We assume that the statement holds for all packets i with $i \leq n-1$. Due to (6.2):

$$\begin{aligned}
& & \forall k < n, \forall p \geq n:\ & E_k \leq E_p + \lambda \\
&\text{thus} & \forall k < n:\ & E_k \leq T + \min_{j\geq n}\{E_j\}, \\
&\text{thus} & & \max_{k\leq n-1}\{E_k\} \leq T + \min_{j\geq n}\{E_j\} \\
&\text{thus} & & D_{n-1} \leq T + \min_{j\geq n}\{E_j\}.
\end{aligned} \tag{C.20}$$

By (6.6) and (6.7):

$$G_n = \min\left\{D_{n-1}, T + \min_{j\geq n}\{E_j\}\right\} = D_{n-1}, \tag{C.21}$$

$$D_n = \max(G_n, E_n) = \max(D_{n-1}, E_n) = \max\left(\max_{k\leq n-1}\{E_k\}, E_n\right) = \max_{k\leq n}\{E_k\}. \tag{C.22}$$

C.2.4 Proof of Theorem 6.2

Consider Fig. 6.1.

First, we prove by induction on $n \geq 1$ that

$$D_n \geq \max_{i<n|E_i \neq +\infty} \{D_i\} \tag{C.23}$$

Base case. By Theorem 6.1, $n = 1$, $D_1 = E_1$, and then (C.23) is obvious.

Induction step. We assume that (C.23) holds for all packet index $i < n$. According to Theorem 6.1, for packet $n \geq 2$, we have:

$$D_n = \max(G_n, E_n), \tag{C.24}$$

$$G_n = \min\left(D_{n-1}, T + \min_{j \geq n}\{E_j\}\right). \tag{C.25}$$

Consider the set $\mathscr{E}_n = \{i < n | E_i \neq +\infty\}$, i.e. the packet numbers less than n that are not lost in the network. If $\mathscr{E}_n$ is empty, the set is empty, (C.23) trivially holds. Therefore, we now assume that $\mathscr{E}_n$ is not empty. Let m be the maximum of $\mathscr{E}_n$. By the induction hypothesis, $D_m = \max_{k<m|E_k \neq +\infty}\{D_k\}$. Then (C.23) gives:

$$\begin{aligned} D_n &\geq \max_{k<n|E_k \neq +\infty} \{D_k\} = \max\left(\max_{k<m|E_k \neq +\infty}\{D_k\}, \max_{m \leq k<n|E_k \neq +\infty}\{D_k\}\right) \\ &= \max\left(D_m, \max_{m \leq k<n|E_k \neq +\infty}\{D_k\}\right). \end{aligned} \tag{C.26}$$

Since $m = \max\{\mathscr{E}_n\}$, we have $\max_{m \leq k<n|E_k \neq +\infty}\{D_k\} = D_m$. Therefore we need to show $D_n \geq D_m$ to prove the theorem. Due to RTO bound for packet m:

$$\begin{aligned} &\forall j > m: \; E_m \leq E_j + \lambda \leq E_j + T, \\ \text{thus} \quad & E_m \leq T + \min_{j>m}\{E_j\}. \end{aligned} \tag{C.27}$$

Since $E_m \leq T + E_m$, then:

$$E_m \leq T + \min_{j \geq m}\{E_j\}. \tag{C.28}$$

By part (2) of Theorem 6.1 for packet m, we have:

$$G_m = \min\left(D_{m-1}, T + \min_{j \geq m}\{E_j\}\right) \leq T + \min_{j \geq m}\{E_j\}, \tag{C.29}$$

$$D_m = \max(G_m, E_m) \leq \max\{T + \min_{j \geq m}\{E_j\}, E_m\} \overset{\text{Eq. (C.28)}}{\Longrightarrow} D_m \leq T + \min_{j \geq m}\{E_j\}. \tag{C.30}$$

We consider the two possible cases for m: 1) $m = n-1$, 2) $m < n-1$.

- $m = n - 1$. Since $n - 1 < n$, $\min_{j \geq n-1}\{E_j\} \leq \min_{j \geq n}\{E_j\}$. Therefore, by (C.29), $D_{n-1} \leq T + \min_{j \geq n}\{E_j\}$. Now, using (C.24):

$$D_n = \max(G_n, E_n) \geq G_n = \min\left(D_{n-1}, T + \min_{j \geq n}\{E_j\}\right) = D_{n-1}. \tag{C.31}$$

- $m < n - 1$. Then $E_{n-1} = +\infty$ and in turn, $D_{n-1} = \max\{G_{n-1}, E_{n-1}\} = +\infty$. By (C.24), we have:

$$D_n = \max(G_n, E_n) \geq G_n = \min\left(D_{n-1}, T + \min_{j \geq n}\{E_j\}\right) = T + \min_{j \geq n}\{E_j\}. \tag{C.32}$$

  Since $m < n - 1$, $\min_{j \geq m}\{E_j\} \leq \min_{j \geq n}\{E_j\}$. Therefore using (C.29) and (C.32), we have:

$$D_n \geq T + \min_{j \geq n}\{E_j\} \geq T + \min_{j \geq m}\{E_j\} \geq D_m. \tag{C.33}$$

This establishes (C.23).

Second, we show that if for any $\lambda > 0$, if $T < \lambda$, there exists a scenario with RTO λ where the re-sequencing buffer discards a packet Consider a trace with two packets 1 and 2, received at the re-sequencing buffer at a times $E_2 = t_0$ and $E_1 = t_0 + \lambda$ for some $t_0 \geq 0$. The RTO of this trace is λ. By Theorem 6.1, $D_1 = I_1$ and $I_1 = +\infty$ because $E_1 > \min_{j \geq n}\{E_j\} + T = E_2 + T$. Thus packet 1 is discarded by the re-sequencing buffer.

C.2.5 Proof of Theorem 6.3

Item (1). Consider Fig. 6.1. Assume the size of the re-sequencing buffer is unlimited; the actual buffer content at time t is

$$L(t) = \sum_{E_k < t, D_k \geq t} l_k, \tag{C.34}$$

First, we show that $L(t) \leq \pi$ at all times t that immediately follow a packet arrival; this will imply that a buffer of size π is sufficient to avoid overflow.

Packet 1 is never stored in the buffer. Consider some fixed but arbitrary packet $n > 1$, with size l_n, and define the set of indices $\mathscr{X}$ by

$$\mathscr{X} \overset{\text{def}}{=} \left\{i \in \mathbb{Z}^+ \mid i < n, D_i \geq E_n\right\} \tag{C.35}$$

If $\mathscr{X}$ is empty then $D_{n-1} < E_n$; observe that $D_n = \max(E_1, ..., E_n) = \max(D_{n-1}, E_n) = E_n$, i.e. packet n is not stored in the buffer, and therefore buffer overflow does not occur when packet n arrives. Hence, we assume that $\mathscr{X}$ is not empty and let $m = \min(\mathscr{X})$. The actual content of the buffer just after the arrival of packet n is

$$\begin{aligned} L(E_n) + l_n &= \sum_{k, E_k < E_n \le D_k} l_k + l_n && \text{(C.36)} \\ &= \sum_{k<m, E_k < E_n \le D_k} l_k + \sum_{k \ge m, E_k < E_n \le D_k} l_k + l_n && \text{(C.37)} \\ &= \sum_{k \ge m, E_k < E_n \le D_k} l_k + l_n \le \sum_{k \ge m, E_k < E_n} l_k + l_n, && \text{(C.38)} \end{aligned}$$

where the last equality is because the first sum in (C.37) is 0 by definition of m. By Lemma C.1, $E_m \ge E_n$; since $m \neq n$ and we exclude simultaneous packet arrivals at the re-sequencing buffer, $E_m > E_n$. Thus

$$\Big[(k \ge m \text{ and } E_k < E_n) \text{ or } (k = n)\Big] \implies (k > m \text{ and } E_k < E_m), \tag{C.39}$$

therefore

$$\sum_{k \ge m, E_k < E_n} l_k + l_n = \sum_{k, (k \ge m, E_k < E_n) \text{or} (k=n)} \le \sum_{k > m, E_k < E_m} l_k = \pi_m. \tag{C.40}$$

Combined with (C.36)-(C.38), this shows that

$$L(E_n) + l_n \le \pi_m \le \pi \tag{C.41}$$

Lemma C.1. $E_m \ge E_n$.

Proof. By construction, $m \in \mathscr{X}$ thus $D_m \ge E_n$. Since $D_m = \max_{k \le m} E_k$, it follows that

$$\exists k \in \{1 \dots m\} \text{ such that } E_k \ge E_n \tag{C.42}$$

If $m = 1$ the conclusion follows. Else, $m - 1$ is not in $\mathscr{X}$ thus $D_{m-1} < E_n$. Since $D_{m-1} = \max_{k \le m-1} E_k$, it follows that $\forall k \in \{1 \dots m-1\}$, $E_k < E_n$. Combined with (C.42), this shows that $E_m \ge E_n$. □

Second, we show that for any possible $\lambda > 0$ and valid RBO value π there exists one execution trace of a flow with packet sizes between $L^{\min}$ and $L^{\max}$, with RTO λ and RBO π, that achieves a buffer content equal to π. First observe that, by definition, π can be written as $\pi = \sum_{j=1}^{k} \ell_j$ for some positive integer k and $\ell_j \in [L^{\min}, L^{\min}]$. The packet sequence is as follows. It has $k+1$ packets in total. Packet 1 has some arbitrary size $l_1 \in [L^{\min}, L^{\min}]$ and is observed at time $E_1 = \lambda$. Packets 2 to $k+1$ have sizes $l_2 = \ell_1, \dots l_{k+1} = \ell_k$ and are observed at times $E_j = \frac{(j-1)\lambda}{k+1}$. Packets 2 to $k+1$ arrive before packet 1, are stored in the buffer until packet 1 arrives, and the buffer content when packet 1 arrives is $\sum_{j=1}^{k} l_j$. The RTOs are $\lambda_1 = \lambda, \lambda_2 = \dots = \lambda_{k+1} = 0$ and the RTO of the trace is λ. The RBOs are $\pi_1 = \sum_{j=1}^{k} \ell_j = \pi$, $\pi_2 = \dots = \pi_{k+1} = 0$ thus the RBO of the sequence is π.

Item (2). First we show that the buffer size is upper bounded by $\alpha(T+V)$. Since delay jitter from the source to the input of the re-sequencing buffer is V, by Lemma 6.1, the flow has arrival curve $\alpha'(t)=\alpha(t+V)$ at the input of re-sequencing buffer. Also, by Theorem 6.4 (the proof of which is independent of this result) the delay at the re-sequencing buffer is upper bounded by the time-out T; by Lemma 6.2, the amount of backlog inside the buffer is thus upper bounded by $\alpha'(T)=\alpha(V+T)$.

Fix some $\varepsilon>0$, smaller than V and T. By the second technical assumption at the end of Section 6.2.4, there exists an integer n and a sequence of packet lengths $\ell_k \in [L^{\min}, L^{\max}]$ such that $\alpha(V+T-\varepsilon)=\sum_{k=1}^{n}\ell_k$. Since the arrival curve is achievable, there also exists a sequence of emission times $t_1=0,\dots t_n=V+T-\varepsilon$ such that the packet sequence $t_1,\dots t_n,\ell_1,\dots,\ell_n$ satisfies the arrival curve constraint α. We now derive another packet sequence of $n+1$ packets as follows.

1. Packet 1 is emitted at time $A_1=0$ and has size $l_1=L^{\max}$.
2. For $k=2\dots n+1$, packet k is emitted at time $A_k=t_0+t_{k-1}$ and has size $l_k=\ell_{k-1}$, where t_0 is a positive number, large enough so that $\alpha(t_0)\geq\sum_{k=1}^{n+1}l_k$. Such a number exists because we assume $\lim_{t\to\infty}\alpha(t)=+\infty$. We have thus $A_{n+1}-A_2=V+T-\varepsilon$.

The arrival times of the $n+1$ packets to the input of re-sequencing buffer are as follows.

1. Packet 1 is lost, i.e. $E_1=+\infty$.
2. Packet $k=2$ arrives at time $E_2=V+A_2-\frac{\varepsilon}{4}$.
3. If $n\geq 2$, for $k=3\dots n+1$, packet k arrives at time $E_k=\max(E_{k-1},A_k)+\frac{\varepsilon}{3(n-1)}$.

We now verify that our scenario satisfies all constraints. There is no simultaneous arrival at the re-sequencing buffer as required by our modelling assumptions. Obviously $E_k\geq A_k$ (the scenario is causal) and $E_k>E_{k-1}$ for $k\geq 3$ therefore there is no reordering, and any RTO or RBO constraint is satisfied.

We now verify that the jitter is $\leq V$. We first show by induction on $k\geq 2, k\leq n+1$ that

$$E_k-A_k\leq V-\frac{\varepsilon(k-2)}{3(n-1)} \tag{C.43}$$

For $k=2$ it follows from the definition of E_2. Consider now $k\geq 3$ and assume it holds for $k-1$.

Then, by the induction hypothesis:

$$\begin{aligned} E_{k-1} &\le A_{k-1} + V - \frac{\varepsilon(k-3)}{3(n-1)} \\ &\le A_k + V - \frac{\varepsilon(k-3)}{3(n-1)} \\ \text{thus} \qquad E_{k-1} + \frac{\varepsilon}{3(n-1)} &\le A_k + V - \frac{\varepsilon(k-2)}{3(n-1)}. \end{aligned} \tag{C.44}$$

Also, as $V > \varepsilon$:

$$A_k + \frac{\varepsilon}{3(n-1)} \le A_k + V - \frac{\varepsilon(k-2)}{3(n-1)}. \tag{C.45}$$

Then, (C.44) and (C.45) give:

$$E_k = \max(E_{k-1}, A_k) + \frac{\varepsilon}{3(n-1)} \le A_k + V - \frac{\varepsilon(k-2)}{3(n-1)}, \tag{C.46}$$

as required. It follows from (C.43) that the jitter of the trace is less than or equal to V.

Next, packet 2 arrives at time $V + A_2 - \frac{\varepsilon}{4}$ and is out of order (due to the loss of packet 1), which triggers a timeout at time $T + V + A_2 - \frac{\varepsilon}{4}$. We now verify that all packets $k \ge 3$ arrive before $V + T + A_2 - \frac{\varepsilon}{4}$. To this end, we show by induction on $k \ge 2$ that

$$E_k \le T + V + A_2 - \frac{\varepsilon(2n-k)}{3(n-1)}. \tag{C.47}$$

For $k = 2$, it follows from the definition of E_2 and $T > \varepsilon$. Assume it holds for $k-1$. Then, by the induction hypothesis

$$\begin{aligned} E_{k-1} &\le T + V + A_2 - \frac{\varepsilon(2n-k+1)}{3(n-1)} \\ \text{thus } E_{k-1} + \frac{\varepsilon}{3(n-1)} &\le T + V + A_2 - \frac{\varepsilon(2n-k)}{3(n-1)}. \end{aligned} \tag{C.48}$$

Also $A_k \le A_{n+1} = T + V + A_2 - \varepsilon$, therefore,

$$A_k + \frac{\varepsilon}{3(n-1)} \le T + V + A_2 - \frac{\varepsilon(3n-2)}{3(n-1)} \le T + V + A_2 - \frac{\varepsilon(2n-k)}{3(n-1)}, \tag{C.49}$$

hence,

$$E_k \le T + V + A_2 - \frac{\varepsilon(2n-k)}{3(n-1)}, \tag{C.50}$$

as required.

Now $\frac{\varepsilon(2n-k)}{3(n-1)} > \frac{\varepsilon}{4}$ for $k = 3 \ldots n+1$ thus every packet other than 2 arrives before packet 2 times out. Thus the buffer content just after the arrival of packet $n+1$ is all packets 2 to $n+1$, i.e. its size is $\alpha(V+T-\varepsilon)=\sum_{k=1}^{n}\ell_k$.

It remains to verify that the trace satisfies the arrival curve constraint. Let $R(t)$ be the cumulative arrival function of the trace, i.e. $R(t)=\sum_{n=1}^{+\infty} l_n 1_{A_n<t}$. First, the sequence of packets 2 to $n+1$ is obtained by time-shifting by t_0 a sequence that satisfies the arrival curve constraint, therefore it also does, namely, $R(t)-R(s)\le\alpha(t-s)$ whenever $s\le t$, $s\ge t_0$ and $t\ge t_0$. It remains to see the other cases:

- $0<s\le t<t_0$: then $R(t)-R(s)=0\le\alpha(t-s)$
- $0=s\le t<t_0$: then $R(t)-R(s)=R(t)-R(0)=l_1\le\alpha(0+)\le\alpha(t-s)$
- $0<s<t_0\le t$: then $R(t)-R(s)=R(t)-R(t_0)\le\alpha(t-t_0)\le\alpha(t-s)$
- $0=s<t_0\le t$: then $R(t)-R(s)=R(t)-R(0)=R(t)-R(t_0)+l_1\le\sum_{k=1}^{n+1}l_k\le\alpha(t_0)$ by construction of t_0. Thus $R(t)-R(0)\le\alpha(t_0)\le\alpha(t)$.

This shows that the arrival curve constraint is satisfied.

At this stage, we have shown that, for every ε small enough, there is a scenario where the backlog reaches $\alpha(V+T-\varepsilon)$. Thus the minimal bound is at least $\sup_{\varepsilon>0}\alpha(V+T-\varepsilon)=\alpha(V+T)$ because arrival curves are left continuous.

C.2.6 Proof of Theorem 6.4

Proof. We use the notations in Fig. 6.1. Suppose that the system has $\delta^{\max}$ and $\delta^{\min}$ as worst-case and best-case delays, thus, for any packet i, $\delta^{\min}\le E_i-A_i\le\delta^{\max}$. Now, consider a received packet n. Since $D_n\ge E_n$, $D_n-A_n\ge E_n-A_n\ge\delta^{\min}$; this shows that the best-case delay is not decreased.

Part (1). The system is lossless, therefore by part (3) of Theorem 6.1, $D_n=\max_{i\le n}\{E_i\}$. Therefore:

$$D_n-A_n=\max_{i\le n}\{E_i\}-A_n=\max_{i|i\le n}\{E_i-A_n\}. \tag{C.51}$$

Since $\forall i\in\mathbb{Z}^+, A_i\le A_{i+1}$:

$$D_n-A_n\le\max_{i\le n}\{E_i-A_i\}\le\delta^{\max}, \tag{C.52}$$

which proves that the worst-case delay is not increased. Since the best case delay is not decreased, the delay jitter is not increased.

Part (2). The system is not lossless. By part (2) of Theorem 6.1, we have:

$$\begin{aligned} G_n &= \min\left(D_{n-1}, T + \min_{j\geq n} E_j\right) \leq T + \min_{j\geq n} E_j \leq T + E_n, \\ D_n &= \max(G_n, E_n) \leq \max(T + E_n, E_n) = T + E_n. \end{aligned} \tag{C.53}$$

Thus

$$D_n - A_n \leq T + E_n - A_n \leq T + \delta^{\max}, \tag{C.54}$$

that proves part (2). □

C.2.7 Proof of Theorem 6.5

Proof. First, we obtain an upper bound for RTO of the flow separately for each part of the theorem. Second we show that each bound is achievable.

Consider a packet n. Let us denote A_n as the arrival time of a packet n (with size l_n) into the system and E_n as its exit time. Now, consider another packet m such that $m \leq n-1$ and $E_n < E_m$. Since V is a jitter bound for this system.

$$(E_m - A_m) - (E_n - A_n) \leq V. \tag{C.55}$$

Then:

$$E_m - E_n \leq V - (A_n - A_m). \tag{C.56}$$

If the flow has arrival curve α, then by (3.21), $A_n - A_m \geq \alpha^{\downarrow}\left(\sum_{k=m}^{n} l_k\right)$:

$$E_m - E_n \leq V - \alpha^{\downarrow}\left(\sum_{k=m}^{n} l_k\right). \tag{C.57}$$

Since $\alpha^{\downarrow}$ is wide-sense increasing and $m \leq n-1$:

$$E_m - E_n \leq V - \alpha^{\downarrow}(l_{n-1} + l_n) \leq V - \alpha^{\downarrow}(2L^{\min}) \leq \left[V - \alpha^{\downarrow}(2L^{\min})\right]^+, \tag{C.58}$$

that proves item (1) in the statement of theorem.

If the flow has packet-level arrival curve α_{pkt}, then by (4.5), $A_n - A_m \geq \alpha^{\downarrow}_{\text{pkt}}(n-m+1)$. Then from (C.56):

$$E_m - E_n \leq V - \alpha^{\downarrow}_{\text{pkt}}(n-m+1). \tag{C.59}$$

Since n is integer and $n > m$, then $n - m \geq 1$:

$$E_m - E_n \leq V - \alpha^{\downarrow}_{\text{pkt}}(2) \leq \left[V - \alpha^{\downarrow}_{\text{pkt}}(2)\right]^+, \tag{C.60}$$

which proves item (2) in the statement of theorem.

Second, we show that the bounds are achievable by constructing a scenario where the RTO for a packet of the flow reaches the bound in item (1) of the theorem.

Consider two packets 1 and 2 with sizes $l_1 = l_2 = L^{\min}$ and a non order-preserving system with a jitter bound V. Packet 1 is issued at $A_1 = 0$ and packet 2 arrives at $A_2 = A_1 + \alpha^{\downarrow}(l_1 + l_2)$, i.e, $(A_2 = t_2)$.

Packet 1 experiences a delay of $d + V$, $E_1 = A_1 + d + V$, and Packet 2 experiences a delay of d, $E_2 = A_2 + d$. Then we have:

$$\begin{aligned} E_1 - E_2 &= A_1 + d + V - (A_2 + d) = V - (t_2 - t_1) \\ &= V - \alpha^{\downarrow}(l_1 + l_2) = V - \alpha^{\downarrow}(2L^{\min}), \end{aligned} \tag{C.61}$$

which shows that the RTO for packet 1 is equal to the bound in part (1) of the theorem.

Now, we verify that jitter bound and arrival curve assumptions are not violated. The difference between the delay of two packets is:

$$(E_1 - A_1) - (E_2 - A_2) = (d + V) - (d) = V, \tag{C.62}$$

which is equal to the jitter bound. Also:

$$A_2 - A_1 = t_2 - t_1 = \alpha^{\downarrow}(l_1 + l_2), \tag{C.63}$$

which shows the arrival curve constraint holds.

The tightness scenario for item (2) of the theorem is similar to the one for part (1). We set $t_2 = t_1 + \alpha^{\downarrow}_{\text{pkt}}(2)$, and the rest follows the description of item (1). □

C.2.8 Proof of Theorem 6.6

Proof. First, we show the bounds in items (1) and (2).

Item (1). First observe that, from Theorem 6.5, if $\alpha(V) < 2L^{\min}$ then there is no reordering and the RBO is 0.

Next, assume that there is some reordering and consider a packet index m such that $\lambda_m > 0$. Let $\mathcal{E}_m = \{i \in \mathbb{Z}^+ | i > m, E_i < E_m\}$. Since $\lambda_m > 0$ and there is no simultaneous arrival of packets, $\mathcal{E}_m$ is not empty. Then let $n = \max\{\mathcal{E}_m\}$. Due to the jitter bound of this system and since

$E_n < E_m$, we have:

$$(E_m - A_m) - (E_n - A_n) \leq V,$$

$$\text{thus} \qquad A_n - A_m \leq V + (E_n - E_m) < V. \tag{C.64}$$

Since the flow has arrival curve α, then by (3.21), $A_n - A_m \geq \alpha^{\downarrow}\left(\sum_{k=m}^{n} l_k\right)$. Therefore,

$$\alpha^{\downarrow}\left(\sum_{k=m}^{n} l_k\right) \leq A_n - A_m < V. \tag{C.65}$$

By [124, Property P7, Section 10.1], we obtain:

$$\sum_{k=m}^{n} l_k \leq \alpha(V). \tag{C.66}$$

We exclude packet m from the left side of the above equation:

$$\sum_{k=m+1}^{n} l_k \leq \alpha(V) - l_m \leq \alpha(V) - L^{\min}. \tag{C.67}$$

The reordering byte offset π_m for packet m, defined in (6.3), includes only the packets with larger index and smaller exit time than packet m. Thus:

$$\pi_m = \sum_{k|k>m, E_k<E_m} l_k \leq \sum_{k=m+1}^{n} l_k. \tag{C.68}$$

Using (C.68) in (C.67), we have:

$$\pi_m \leq \alpha(V) - L^{\min}, \tag{C.69}$$

which proves item (1) of the theorem.

Item (2). The flow has packet-level arrival curve α_{pkt}. Observe that, from Theorem 6.5, if $\alpha_{\text{pkt}}(V) < 2$ then there is no reordering and the RBO is 0.

Next, since the flow has packet-level arrival curve α_{pkt}, then by (4.5), $A_n - A_m \geq \alpha^{\downarrow}_{\text{pkt}}(n - m + 1)$. From (C.64), we have:

$$\alpha^{\downarrow}_{\text{pkt}}(n - m + 1) \leq A_n - A_m < V, \tag{C.70}$$

By [124, Property P7, Section 10.1], we obtain:

$$n - m + 1 \leq \alpha_{\text{pkt}}(V). \tag{C.71}$$

Since for any packet k, $l_k \leq L^{\max}$:

$$\sum_{k=m+1}^{n} l_k \leq L^{\max}(n-m) \leq L^{\max}\left(\alpha_{\text{pkt}}(V)-1\right). \tag{C.72}$$

Since $\pi_m \leq \sum_{k=m+1}^{n} l_k$, item (2) of the theorem is proven.

Second, we show the tightness.

Fix some $\varepsilon > 0$, smaller than λ. By the second technical assumption at the end of Section 6.2.4, we show the tightness for the two cases, i) $L^{\max} \geq 2L^{\min}$, ii) $L^{\min} = L^{\max}$.

Case i) $L^{\max} \geq 2L^{\min}$. By assumption, we know that $\alpha(0^+) \geq L^{\max}$; therefore, $\alpha(0^+) \geq 2L^{\min}$.

By the second technical assumption at the end of Section 6.2.4, there exists an integer n and a sequence of packet lengths $l_k \in [L^{\min}, L^{\max}]$ such that $l_1 = L^{\min}$ and $\sum_{k=2}^{n} l_k = \alpha(V-\varepsilon) - L^{\min}$. Since $\varepsilon < \lambda$ and by Theorem 6.5 $\lambda \leq V$, $\alpha(V-\varepsilon) \geq 2L^{\min}$; therefore, $n \geq 2$. Now, since the arrival curve is achievable, there also exists a sequence of emission times $A_1 = 0, \ldots A_n = V-\varepsilon$ such that the packet sequence $A_1, \ldots A_n, l_1, \ldots, l_n$ satisfies the arrival curve constraint α.

Next, we construct the exit times of packets k from the system as follows:

$$E_1 = V+\varepsilon, E_k = V + \frac{(k-2)\varepsilon}{n}, \quad k = 2, \ldots, n. \tag{C.73}$$

Observe that $E_2 < E_3 < \cdots < E_n < E_1$. Also note that $A_1 \leq A_2 \leq \cdots \leq A_n = V-\varepsilon < E_2 = V$.

Now, according to (6.3), the RBOs for packet 1 and packet k, $k = 2, \ldots, n$, are:

$$\begin{aligned}
\pi_1 &= \sum_{j|j>1, E_j<E_1} l_j = \sum_{k=2}^{n} l_k = \alpha(V-\varepsilon) - L^{\min}, \\
\pi_k &= \sum_{j|j>k, E_j<E_k} l_j = 0.
\end{aligned} \tag{C.74}$$

Therefore, $\pi = \max_{1\leq i\leq n}\{\pi_i\} = \alpha(V-\varepsilon) - L^{\min}$.

Finally, we verify that the assumptions are not violated: 1) arrival curve, 2) jitter bound, 3) RTO of the flow.

(1) The arrival curve constraint is satisfied by construction.

(2) For any packet $k \geq 1$, we have:

$$E_k - A_k \leq E_1 - A_1 = V + \varepsilon, \tag{C.75}$$

$$E_k - A_k \geq E_2 - A_n = V - (V-\varepsilon) = \varepsilon. \tag{C.76}$$

Therefore, the jitter is:

$$\max_k\{E_k - A_k\} - \min_k\{E_k - A_k\} \leq V + \varepsilon - \varepsilon = V, \tag{C.77}$$

which conforms the jitter constraint.

(3) For any packet $k \geq 2$, the packets are in order

$$\lambda_k = E_k - \min_{j|j\geq k, E_j \leq E_k} = E_k - E_k = 0. \tag{C.78}$$

For packet 1, we have:

$$\lambda_1 = E_1 - \min_{j|j\geq 1, E_1 \leq E_k} = E_1 - E_2 = (V + \varepsilon) - V = \varepsilon, \tag{C.79}$$

therefore, $\lambda = \max_{1\leq i\leq n}\{\lambda_i\} = \varepsilon$, that satisfies the RTO constraint.

Thus, we have shown that, for every ε small enough, there is a scenario where the RBO reaches $\alpha(V-\varepsilon)-L^{\min}$. Thus the minimal bound is at least $\sup_{\varepsilon>0}\alpha(V-\varepsilon)-L^{\min} = \alpha(V)-L^{\min}$ because arrival curves are left continuous.

Case ii) $L^{\max} = L^{\min}$. Then all the packets have the same size $l = L^{\max}$. By assumption, we know that $\alpha(0^+) \geq l$.

By the second technical assumption at the end of Section 6.2.4, there exists an integer n and a sequence of packets with length l such that $\sum_{k=1}^{n} l_k = \alpha(V-\varepsilon) - l$. Now, if $\alpha(V-\varepsilon) < 2l$, then $n = 1$ and therefore no reordering occurs and $\pi = 0$. Hence, we consider the case $\alpha(V-\varepsilon) \geq 2l$; then $n \geq 2$. Now, since the arrival curve is achievable, there also exists a sequence of emission times $t_1 = 0, \dots t_n = V - \varepsilon$ such that the packet sequence $t_1, \dots t_n, l_1, \dots, l_n$ satisfies the arrival curve constraint α. The rest of the proof follows exactly as case (i).

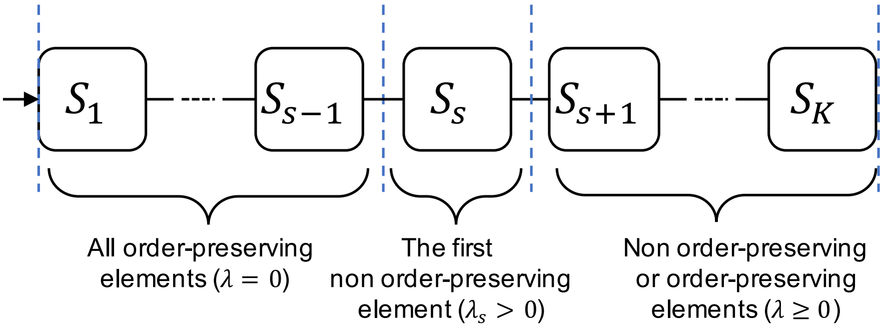
The tightness scenario for item (2) of the theorem is similar to the one for case (ii). We set $n = \alpha_{\text{pkt}}(V)$ and for any $k = 1, \dots, n$, $l_k = L^{\max}$. □

C.2.9 Proof of Theorem 6.7

Proof. First, we obtain an upper bound for RTO of the flow separately for each part of the theorem. Second we show that each bound is achievable.

Consider Fig. C.1. We denote the arrival and exit times of a packet i at S_h by E_i^{h-1} and E_i^h. Consider two packet indices m and n such that $m < n$ and $E_n^K < E_m^K$. Since S_s is the first system with nonzero RTO, for any system S_h, $h < s$, $E_m^h < E_n^h$. Therefore, $E_m^{s-1} \leq E_n^{s-1}$, i.e., at the output of system S_{s-1} (input of system S_s), packet m and n are in-order. Therefore, by definition of the RTO bound at S_s,

$$E_m^s - E_n^s \leq \lambda_s. \tag{C.80}$$



S_1
S_{s-1}
S_s
S_{s+1}
S_K
All order-preserving elements ($\lambda = 0$)
The first non order-preserving element ($\lambda_s > 0$)
Non order-preserving or order-preserving elements ($\lambda \geq 0$)

Figure C.1: Notation for the sequence of network elements used in Theorem 6.7.

Now, according to the jitter bound for the concatenation of systems S_{s+1} to S_K, we have:

$$\left(E_m^K - E_m^s\right) - \left(E_n^K - E_n^s\right) \leq \sum_{h=s+1}^{K} V_h. \tag{C.81}$$

Then, we have:

$$E_m^K - E_n^K \leq \left(E_m^s - E_n^s\right) + \sum_{h=s+1}^{K} V_h. \tag{C.82}$$

Combining with (C.80):

$$E_m^K - E_n^K \leq \lambda_s + \sum_{h=s+1}^{K} V_h \overset{\text{def}}{=} \Lambda(K). \tag{C.83}$$

Second, we show tightness. We are given a sequence of systems with RTOs λ_h and jitters V_h, and we construct a scenario that conforms with these parameters and where a packet reaches the RTO bound in Theorem 6.7 at system K. We use the same notation as before. In particular, S_s is the first system in the sequence for which $\lambda_s > 0$. Now, consider a trace with two packets 1 and 2 entering S_1. Also, consider a set of positive values $\{d_s, d_{s+1}, \ldots, d_K\}$.

Packet 1 and 2 arrive at S_1 at times $E_1 = 0$ and $E_2 = \varepsilon$, with $\lambda_s > \varepsilon > 0$. Each packet then has the same transfer time through system S_j at time t_j, $j \in \{1, 2, \ldots, s-1\}$, thus preserving order and $E_2^{s-1} = E_1^{s-1} + \varepsilon$.

The transfer times through S_s are $d_s + \lambda_s$ for packet 1 and d_s for packet 2, i.e., $E_1^s = E_1^{s-1} + d_s + \lambda_s$ and $E_2^s = E_2^{s-1} + d_s$. Packet 1 and 2 experience the delays of $d_h + V_h$ and d_h at system S_h, $h \in \{s+1, s+2, \ldots, K\}$:

$$\begin{aligned} E_1^h &= E_1^{h-1} + d_h + V_h, \\ E_2^h &= E_2^{h-1} + d_h. \end{aligned} \tag{C.84}$$

Then at the output of system S_K, we have:

$$
\begin{aligned}
E_1^K &= E_1^s + \sum_{h=s+1}^{K} (d_h + V_h) = E_1^{s-1} + \lambda_s + \sum_{h=s}^{K} d_h + \sum_{h=s+1}^{K} V_h, \\
E_2^K &= E_2^s + \sum_{h=s+1}^{K} d_h = E_1^{s-1} + \varepsilon + \sum_{h=s}^{K} d_h. \qquad \text{(C.85)}
\end{aligned}
$$

We now verify that the assumptions in the statement of theorem are not violated.

(1) We check that RTOs for all the systems are not violated. For any system $S_j, j < s$, packets 1 and 2 preserve order by construction, i.e. $\lambda_j = 0$.

For system S_s, according to the departure times of packets 1 and 2 from S_s, we have:

$$
E_1^s - E_2^s = \lambda^s - \varepsilon, \qquad \text{(C.86)}
$$

which satisfies the constraint λ_s on RTO conforms to RTO assumption for system S_s being the first system that RTO is equal to λ_s. At the output of system S_s we have $E_2^s < E_2^s$, i.e., packet 2 is prior to packet 1. For system S_h, $h \in \{s+1, s+1, \dots, K\}$, we have

$$
\begin{aligned}
E_2^h - E_1^h &= \left(E_1^{s-1} + \varepsilon + \sum_{j=s}^{h} d_j \right) - \left(E_1^{s-1} + \lambda_s + \sum_{j=s}^{h} d_j + \sum_{j=s+1}^{h} V_j \right) \\
&= \varepsilon - \lambda_s - \sum_{j=s+1}^{h} V_j < 0, \qquad \text{(C.87)}
\end{aligned}
$$

that shows $E_2^h < E_1^h$, i.e., system $S_h, h \geq s+1$ preserves the order of its input. Therefore, RTO for each system S_h is 0, which satisfies any RTO constraint.

(2) We check that the jitter bounds are not violated.

For systems S_1 to S_{s-1}, since both packets experience the same delay, the jitter is 0.

For system S_s, we have:

$$
\left(E_1^s - E_1^{s-1}\right) - \left(E_2^s - E_2^{s-1}\right) = \lambda_s.
$$

Now, by Theorem 6.5, $\lambda_s \leq V_s$, thus the jitter bound assumption for S_s are satisfied.

For any system S_h where $h \in \{s+1, \dots, K\}$, we have:

$$
\left(E_1^h - E_1^{h-1}\right) - \left(E_2^h - E_2^{h-1}\right) = (d_h + V_h) - (d_h) = V_h,
$$

which shows that the jitter bound assumptions for systems $S_{s+1}, S_{s+2}, \dots, S_K$ are satisfied.

Last,

$$E_1^K - E_2^K = \lambda_s - \varepsilon + \sum_{h=s+1}^{K} V_h, \tag{C.88}$$

thus the RTO for packet 1 is equal to the bound in Theorem 6.7 minus ε. Since ε can be arbitrarily small, this shows the result.

□

C.3 Details of Computations for Case Study 1 in Section 6.5.2

Table C.1: Arrival curve propagation from point 1 to point 9 under lossless network condition. Arrival curve at point i is $\alpha_i(t) = \min(6.4e3\ t + b_i, 125e6\ t + M_i)$, where b_i and M_i are in bytes and shown in the table. Since under lossless network condition, the re-sequencing buffers do not increase delay and jitter bounds, the arrival curves are the same for the four placement strategies.

Burst	1	2	3	4	5	6	7	8	9
b_i	6400	6400		6400		6400		6400	
M_i	64	251		1751	64	251		1751	64

In these scenarios, we require arrival curve information to compute RTO, RBO, and delay upper bounds. Let us call α_i as arrival curve of flow f at point i in units of bytes. Tables C.1 and C.2 show the propagated arrival curve at points 1 to 9 in Fig. 6.4. The arrival curve at point i has the form of $\alpha_i(t) = \min(r\ t + b_i, c\ t + M_i)$, where $r = 6400$ bytes per second, $c = 125e5$ bytes per second (i.e. $1Gbps$), b_i and M_i are shown in Tables C.1 and C.2. $\alpha_0(t) = r\ t + b_0$, where $b_0 = 6400\ bytes$. The arrival curves at point 1, 5, and 9 capture the line shaping and packetizer effects:

$$\begin{aligned}
\alpha_1(t) &= \min(r\ t + b_0, c\ t + 512),\\
\alpha_5(t) &= \min(r\ t + b_4, c\ t + 512),\\
\alpha_9(t) &= \min(r\ t + b_8, c\ t + 512).
\end{aligned} \tag{C.89}$$

The arrival curves at points 4 and 8 capture the effect of traversing the output FIFO systems with service curve $\beta(t) = 125e6[t - Q]^+\ bytes$, where $Q = 12\mu s$:

$$\begin{aligned}
\alpha_4(t) &= \min(r\ t + b_3 + rQ, c\ t + M_3 + cQ),\\
\alpha_8(t) &= \min(r\ t + b_7 + rQ, c\ t + M_7 + cQ).
\end{aligned} \tag{C.90}$$

The arrival curves at point 2 and 6 capture the effect of switching fabric with jitter bound V_{sf}:

$$\begin{aligned}
\alpha_2(t) &= \min(r\ t + b_1 + rV_{\text{sf}}, c\ t + M_1 + cV_{\text{sf}}),\\
\alpha_6(t) &= \min(r\ t + b_5 + rV_{\text{sf}}, c\ t + M_5 + cV_{\text{sf}}).
\end{aligned} \tag{C.91}$$

Table C.2: Arrival curve propagation from point 1 to point 9 under lossy network condition. Arrival curve at point i is $\alpha_i(t) = \min(6.4e3\ t + b_i, 125e6\ t + M_i)$, where b_i and M_i are in bytes and shown in the table.

Re-sequencing	Burst	1	2	3	4	5	6	7	8	9
Only at h_2	b_i	6400	6400		6400		6400		6400	
	M_i	64	251		1751	64	251		1751	64
Only at S_2	b_i	6400	6400		6400		6400	6400	6400	
	M_i	64	251		1751	64	251	2249	3749	64
At S_1 and h_2	b_i	6400	6400	6400	6400		6400		6400	
	M_i	64	251	626	1875	64	2012		1751	64
At S_1 and S_2	b_i	6400	6400	6400	6400		6400	6400	6400	
	M_i	64	251	626	1875	64	251	375	1875	64

The arrival curves at point 3 and 7 capture the effect of re-sequencing buffer (if any) at switches S_1 and S_2 with time-out values respectively T_1 and T_2 under lossy network condition:

$$\begin{aligned}\alpha_3(t) &= \min(r\ t + b_2 + rT_1, c\ t + M_2 + cT_1),\\ \alpha_7(t) &= \min(r\ t + b_6 + rT_2, c\ t + M_6 + cT_2).\end{aligned} \tag{C.92}$$

Under lossless network condition, the re-sequencing buffers do not increase delay and jitter bounds, hence, $\alpha_3 = \alpha_2$ and $\alpha_7 = \alpha_6$.

To compute delay bounds of output FIFO systems we use the results in [60, 25]. Accordingly, the delay bound of FIFO system of h_1 is $\delta^{\max}_{\text{FIFO},h_1} = 63.2\mu s$. Also, the minimum delay for each FIFO system is the transmission of a packet with minimum length, $\delta^{\min}_{\text{FIFO},h_1} = \delta^{\min}_{\text{FIFO},S_1} = \delta^{\min}_{\text{FIFO},S_2} = \frac{L^{\min}}{c} = 512ns$. Now, since we know delay upper and lower bounds for the FIFO system of h_1, its jitter is $V_{\text{FIFO},h_1} = 62.69\mu s$. Now we analyze the four strategies separately.

Re-sequencing only at h_2

We first obtain delay and jitter bounds of flow f for the FIFO output ports of S_1 and S_2. Using the arrival curves in Tables C.1 and C.2:

$$\delta^{\max}_{\text{FIFO},S_1} = \delta^{\max}_{\text{FIFO},S_2} = 14.01\ \mu\text{s}. \tag{C.93}$$

Since, we already compute the minimum delay for the FIFO systems of S_1 and S_2, the jitter bounds are:

$$V_{\text{FIFO},S_1} = V_{\text{FIFO},S_2} = 13.5\ \mu\text{s}. \tag{C.94}$$

Having the knowledge on jitters of each element, we now compute the RTO bound of the flow at h_2. To obtain the RTO bound, we use Theorem 6.7. Since the switching fabric (SF) in S_1 is the first non order-preserving element, we need to compute its corresponding RTO. Using

Theorem 6.5:

$$\lambda_{\text{SF},S_1} = \left[V_{\text{SF},S_1} - \alpha_1^{\downarrow}(2L^{\min})\right]^+ = 1.5\mu s - 0.512\mu s = 0.988\ \mu\text{s}. \tag{C.95}$$

Therefore the RTO at h_2 is:

$$\begin{aligned}\Lambda(h_2) &= \lambda_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + V_{\text{FIFO},S_2} \\ &= 0.988\mu + 13.5\mu + 1.5\mu + 13.5\mu = 29.49\ \mu\text{s}.\end{aligned} \tag{C.96}$$

Then $T_{h_2} = \Lambda(h_2) = 29.49\ \mu$s.

Now, if the network is lossless, due to Theorem 6.4, re-sequencing is for free; therefore, the bounds are:

$$\begin{aligned}\delta_{\text{e2e}}^{0,\max} &= \delta_{\text{FIFO},h_1}^{\max} + \delta_{\text{SF},S_1}^{\max} + \delta_{\text{FIFO},S_1}^{\max} + \delta_{\text{SF},S_2}^{\max} + \delta_{\text{FIFO},S_2}^{\max} = 63.2 + 2 + 14.01 + 2 + 14.01 = 95.22\ \mu\text{s}, \\ V_{\text{e2e}}^{0} &= V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + V_{\text{FIFO},S_2} = 62.69 + 1.5 + 13.5 + 1.5 + 13.5 = 92.69\ \mu\text{s}.\end{aligned}$$

Using Corollary 6.2, the RBO bound at h_2 is:

$$\Pi(h_2) = \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2}) - L^{\min} = \alpha(79.19\mu s) - 64 = 6336 \text{ bytes}. \tag{C.97}$$

Then $B_{h_2} = \Pi(h_2) = 6336$ bytes. Note that due to Theorem 6.6, we eliminate the FIFO system at S_2 as the switching fabric of S_2 is the last FIFO system.

If the network is lossy, by Theorem 6.4, the jitter and delay worst-case are increased by $T_{h_2} = 29.49\ \mu$s; therefore:

$$\delta_{\text{e2e}}^{\max} = 124.72\mu s,\ V_{\text{e2e}} = 122.19\ \mu\text{s}. \tag{C.98}$$

The size of re-sequencing buffer is:

$$B_{h_2} = \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + V_{\text{FIFO},S_2} + T_{h_2}) = \alpha(122.19\ \mu\text{s}) = 6400 \text{ bytes}.$$

Re-sequencing only at S_2

Similarly to the previous scenario, $\delta_{\text{FIFO},S_1}^{\max} = 14.01\ \mu$s and $V_{\text{FIFO},S_1} = 13.5\ \mu$s; and therefore, we obtain RTO bound in switching fabric of S_1 as $\lambda_{\text{SF},S_1} = 0.988\ \mu$s. Using Theorem 6.7, the RTO bound after the switching fabric of S_2 is:

$$\Lambda(S_2) = \lambda_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} = 0.988\mu + 13.5\mu + 1.5\mu = 15.99\ \mu\text{s}. \tag{C.99}$$

Then $T_{S_2} = \Lambda(S_2) = 15.99\ \mu$s.

Now, if the network is lossless, $\delta_{\text{FIFO},S_2}^{\max} = 14.01\ \mu$s. Then, due to Theorem 6.4, re-sequencing is for free; therefore, similarly to the previous strategy $\delta_{\text{e2e}}^{0,\max} = 95.22\ \mu$s and $V_{\text{e2e}}^{0} = 92.69\ \mu$s. Also for the RBO bound at S_2, using Corollary 6.2, we have:

$$\Pi(S_2) = \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2}) - L^{\min} = \alpha(79.19\mu s) - 64 = 6336 \text{ bytes.} \quad (\text{C.100})$$

By Theorem 6.2, and the size is $B_{S_2} = \Pi(S_2) = 6336$ bytes.

If the network is lossy, by Theorem 6.4, the jitter and delay worst-case are increased by $T_{S_2} = 15.99\mu s$; therefore this affects the arrival curve at point 7 and in turn delay bound of output FIFO system at S_2. Then, $\delta^{\max}_{\text{FIFO},S_2} = 30\ \mu$s and $V_{\text{FIFO},S_2} = 29.49\ \mu$s. Finally,

$$\begin{aligned}\delta^{\max}_{\text{e2e}} &= \delta^{\max}_{\text{FIFO},h_1} + \delta^{\max}_{\text{SF},S_1} + \delta^{\max}_{\text{FIFO},S_1} + \delta^{\max}_{\text{SF},S_2} + \delta^{\max}_{\text{FIFO},S_2} + T_{S_2} = 127.22\ \mu\text{s},\\ V_{\text{e2e}} &= V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + V_{\text{FIFO},S_2} + T_{S_2} = 124.69\ \mu\text{s}.\end{aligned}$$

The size of re-sequencing buffer is:

$$B_{S_2} = \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + T_{S_2}) = \alpha(95.18\ \mu\text{s}) = 6400 \text{ bytes.} \quad (\text{C.101})$$

Re-sequencing at S_1 and h_2

Since we already computed arrival curve at point 1, we compute RTO bound switching fabric at S_1 using Theorems 6.5, $\lambda_{\text{SF},S_1} = 0.988\ \mu$s. Then $T_{S_1} = \lambda_{\text{SF},S_1} = 0.988\ \mu$s. Similarly to switching fabric of S_1, we have $\lambda_{\text{SF},S_2} = 0.988\ \mu$s.

Now, if the network is lossless, using the arrival curves in Table C.1, $\delta^{\max}_{\text{FIFO},S_1} = \delta^{\max}_{\text{FIFO},S_2} = 14.01\ \mu$s and $V_{\text{FIFO},S_1} = V_{\text{FIFO},S_2} = 13.5\ \mu$s. Hence, due to Theorem 6.4, re-sequencing is for free; therefore, similarly to the previous strategy $\delta^{0,\max}_{\text{e2e}} = 95.22\ \mu$s and $V^0_{\text{e2e}} = 92.69\ \mu$s. To compute RTO bound at h_2, we need to find RTO bound switching fabric of S_2 as the first non order-preserving element after re-sequencing buffer of S_1 (the output of which is in-order). Using Theorem 6.7, the RTO bound at h_2 is:

$$\Lambda(h_2) = \lambda_{\text{SF},S_2} + V_{\text{FIFO},S_2} = 0.988\mu + 13.5\mu = 14.49\ \mu\text{s}.$$

Then $T_{h_2} = \Lambda(h_2) = 14.49\ \mu$s. Also for the RBO bound at S_1 and h_1, using Corollary 6.2, we have:

$$\begin{aligned}\Pi(S_1) &= \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1}) - L^{\min} = \alpha(64.19\ \mu\text{s}) - 64 = 6336 \text{ bytes},\\ \Pi(h_2) &= \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2}) - L^{\min} = \alpha(79.19\ \mu\text{s}) - 64 = 6336 \text{ bytes}.\end{aligned}$$

By Theorem 6.2, $B_{S_1} = B_{h_2} = \Pi(S_2) = 6336$ bytes.

If the network is lossy, the re-sequencing buffer at S_1 increases the jitter by T_{S_1}; the impact on arrival curves is shown in Table C.2. Hence, $\delta^{\max}_{\text{FIFO},S_1} = 15\ \mu\text{s}, \delta^{\max}_{\text{FIFO},S_2} = 14.01\ \mu$s. Using the obtained lower and upper delay bounds, $V_{\text{FIFO},S_1} = 14.49\ \mu\text{s}, V_{\text{FIFO},S_2} = 13.5\ \mu$s. To compute

RTO bound at h_2, similarly to the lossless case,

$$\Lambda(h_2) = \lambda_{\text{SF},S_2} + V_{\text{FIFO},S_2} = 0.988\mu + 13.5\mu = 14.49\ \mu\text{s}.$$

Then $T_{h_2} = \Lambda(h_2) = 14.49\ \mu$s. Finally:

$$\begin{aligned}\delta^{\max}_{\text{e2e}} &= \delta^{\max}_{\text{FIFO},h_1} + \delta^{\max}_{\text{SF},S_1} + \delta^{\max}_{\text{FIFO},S_1} + \delta^{\max}_{\text{SF},S_2} + \delta^{\max}_{\text{FIFO},S_2} + T_{S_1} + T_{h_2} = 111.72\ \mu\text{s},\\ V_{\text{e2e}} &= V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + V_{\text{FIFO},S_2} + T_{S_1} + T_{h_2} = 109.19\ \mu\text{s}.\end{aligned}$$

The size of re-sequencing buffers are:

$$\begin{aligned}B_{S_1} &= \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + T_{S_1}) = \alpha(65.18\ \mu\text{s}) = 6400 \text{ bytes},\\ B_{h_2} &= \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + T_{S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + V_{\text{FIFO},S_2} + T_{h_2}) = \alpha(109.19\ \mu\text{s}) = 6400 \text{ bytes}.\end{aligned}$$

Re-sequencing at S_1 and S_2

Similarly to the previous scenario, we have $T_{S_1} = \lambda_{\text{SF},S_1} = 0.988\ \mu$s. Also $\lambda_{\text{SF},S_2} = 0.988\ \mu$s.

Now, if the network is lossless, using the arrival curves in Table C.1, $\delta^{\max}_{\text{FIFO},S_1} = \delta^{\max}_{\text{FIFO},S_2} = 14.01\ \mu$s and $V_{\text{FIFO},S_1} = V_{\text{FIFO},S_2} = 13.5\ \mu$s. Hence, due to Theorem 6.4, re-sequencing is for free; therefore, similarly to the previous strategy $\delta^{0,\max}_{\text{e2e}} = 95.22\ \mu$s and $V^0_{\text{e2e}} = 92.69\ \mu$s. To compute RTO bound at h_2, we need to find RTO bound switching fabric of S_2 as the first non order-preserving element after re-sequencing buffer of S_1 (the output of which is in-order). Using Theorem 6.7, the RTO bound at S_2 is $\Lambda(S_2) = \lambda_{S_2} = 0.988\ \mu$s. Also for the RBO bound at S_1 and S_2, using Corollary 6.2, we have:

$$\begin{aligned}\Pi(S_1) &= \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1}) - L^{\min} = \alpha(64.19\ \mu\text{s}) - 64 = 6336 \text{ bytes},\\ \Pi(S_2) &= \alpha(V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2}) - L^{\min} = \alpha(79.19\ \mu\text{s}) - 64 = 6336 \text{ bytes}.\end{aligned}$$

By Theorem 6.2, $B_{S_1} = B_{S_2} = \Pi(S_2) = 6336$ bytes.

If the network is lossy, the re-sequencing buffer at S_1 increases the jitter by T_{S_1}; the impact on arrival curves is shown in Table C.2. Hence, $\delta^{\max}_{\text{FIFO},S_1} = 15\ \mu\text{s}, \delta^{\max}_{\text{FIFO},S_2} = 14.01\ \mu$s. Using the obtained lower and upper delay bounds, $V_{\text{FIFO},S_1} = 14.49\ \mu\text{s}, V_{\text{FIFO},S_2} = 13.5\ \mu$s. To compute RTO bound at h_2, similarly to the lossless case, $\Lambda(S_2) = \lambda_{\text{SF},S_2} = 0.988\ \mu$s. Then $T_{S_2} = \Lambda(S_2) = 0.988\ \mu$s. Finally:

$$\begin{aligned}\delta^{\max}_{\text{e2e}} &= \delta^{\max}_{\text{FIFO},h_1} + \delta^{\max}_{\text{SF},S_1} + \delta^{\max}_{\text{FIFO},S_1} + \delta^{\max}_{\text{SF},S_2} + \delta^{\max}_{\text{FIFO},S_2} + T_{S_1} + T_{S_2} = 99.22\ \mu\text{s},\\ V_{\text{e2e}} &= V_{\text{FIFO},h_1} + V_{\text{SF},S_1} + V_{\text{FIFO},S_1} + V_{\text{SF},S_2} + V_{\text{FIFO},S_2} + T_{S_1} + T_{S_2} = 96.69\ \mu\text{s}.\end{aligned}$$

The size of re-sequencing buffers are:

$$B_{S_1} = \alpha(V_{\mathrm{FIFO},h_1} + V_{\mathrm{SF},S_1} + T_{S_1}) = \alpha(65.18\ \mu\mathrm{s}) = 6400 \text{ bytes},$$
$$B_{S_2} = \alpha(V_{\mathrm{FIFO},h_1} + V_{\mathrm{SF},S_1} + T_{S_1} + V_{\mathrm{FIFO},S_1} + V_{\mathrm{SF},S_2} + T_{h_2}) = \alpha(83.19\ \mu\mathrm{s}) = 6400 \text{ bytes}.$$

D Appendix (Chapter 7)

D.1 Lemmas for Section 7.3

Lemma D.1. *Consider a non-decreasing sequence* $I = \{I_1, \ldots, I_n\}$ *and a sequence* $\phi = \{\phi_1, \ldots, \phi_n\}$. *Assume the sequence* $O = \{O_1, \ldots, O_n\}$ *is defined by*

$$O_1 = I_1 + \phi_1, \;\; O_n = \max(I_n, O_{n-1}) + \phi_n.$$

Then a closed-form formula for O *is:*

$$O_n = \max_{m \leq n} \left\{ I_m + \sum_{i=m}^{n} \phi_i \right\}.$$

Proof. We prove by induction. Base case $n = 1$.

$$O_1 = I_1 + \phi_1, \tag{D.1}$$

as required by the lemma.

Induction step. We assume that the lemma holds for all $i < n$. Then for $i = n - 1$, by the closed-form formula we have:

$$O_{n-1} = \max_{m \leq n-1} \left\{ I_m + \sum_{i=m}^{n-1} \phi_i \right\}. \tag{D.2}$$

Then, using the recursive definition of O_n:

$$\begin{aligned} O_n &= \max(I_n, O_{n-1}) + \phi_n = \max\left(I_n + \phi_n, \max_{m \leq n-1} \left\{ I_m + \sum_{i=m}^{n-1} \phi_i \right\} + \phi_n \right) \\ &= \max\left(I_n + \phi_n, \max_{m \leq n-1} \left\{ I_m + \sum_{i=m}^{n} \phi_i \right\} \right) = \max_{m \leq n} \left\{ I_m + \sum_{i=m}^{n} \phi_i \right\}. \end{aligned} \tag{D.3}$$

□

Lemma D.2. *Let* a*,* $x^{\min} \leq x^{\max}$ *and* $y^{\min} \leq y^{\max}$ *be five fixed real numbers. For any* $z \in \mathbb{R}$*, if*

$$x^{\min} + \max(a, y^{\min}) \leq z \leq x^{\max} + \max(a, y^{\max}), \tag{D.4}$$

then there exists some $x, y \in \mathbb{R}$ *such that*

$$z = x + \max(a, y), \tag{D.5}$$

$$x \in [x^{\min}; x^{\max}], \tag{D.6}$$

$$y \in [y^{\min}; y^{\max}]. \tag{D.7}$$

Comversely, if there exists $x, y \in \mathbb{R}$ *such that* (D.5)–(D.7) *hold, then* z *satisfies* (D.4).

Proof. 1) Let $D = [x^{\min}; x^{\max}] \times [y^{\min}; y^{\max}]$ and f the mapping $D \to \mathbb{R}$ defined by $f(x, y) = x + \max(a, y)$. f is continuous and D is compact and connected, therefore $f(D)$ is compact and connected. The compact and connected subsets of $\mathbb{R}$ are the closed, bounded intervals, therefore $f(D) = [m; M]$ for some $m, M \in \mathbb{R}$. Necessarily, m is the minimum of f over D and M is the maximum of f over D.

Now for every $(x, y) \in D$:

$$f(x^{\min}, y^{\min}) \leq f(x, y) \leq f(x^{\max}, y^{\max}). \tag{D.8}$$

It follows that the minimum of f over D is $f(x^{\min}, y^{\min})$, i.e. $m = f(x^{\min}, y^{\min})$. Similarly, $M = f(x^{\max}, y^{\max})$. Now let $z \in \mathbb{R}$ that satisfies (D.4), i.e. $z \in [m; M] = f(D)$. Thus, there exists some $(x, y) \in D$ such that $z = f(x, y)$, i.e. there exists $x, y \in \mathbb{R}$ such that (D.5)–(D.7) hold.

2) Conversely, if there exists $x, y \in \mathbb{R}$ such that (D.5)–(D.7) hold, then $z = f(x, y)$ and thus $m \leq z \leq M$, i.e. (D.4) holds. □

D.2 Proof of Theorem 7.1

Let us define d_i as the delay of JCS i ($i = 1, \ldots, K$), π_j as the delay of BDS j ($j = 1, \ldots, K'$) and t as the delay of the damper. Assume that JCS i is operating with clock $\mathscr{H}_i$ and the damper is operating with clock $\mathscr{H}_{\text{damper}}$. Then, the delay of the block, in TAI, is:

$$d^{\mathscr{H}_{\text{TAI}}} = \sum_{i=1}^{K} d_i^{\mathscr{H}_{\text{TAI}}} + \sum_{j=1}^{K'} \pi_j^{\mathscr{H}_{\text{TAI}}} + t^{\mathscr{H}_{\text{TAI}}} = \sum_{i=1}^{K} d_i^{\mathscr{H}_i} + \sum_{j=1}^{K'} \pi_j^{\mathscr{H}_{\text{TAI}}} + t^{\mathscr{H}_{\text{damper}}} + \gamma, \tag{D.9}$$

where $\gamma = \sum_{i=1}^{K} \left(d_i^{\mathscr{H}_{\text{TAI}}} - d_i^{\mathscr{H}_i} \right) + \left(t^{\mathscr{H}_{\text{TAI}}} - t^{\mathscr{H}_{\text{damper}}} \right)$. By (7.3) and (7.4), the delay of a packet with damper header H inside the damper is

$$H - \Delta^{\text{L}} \leq t^{\mathscr{H}_{\text{damper}}} \leq H + \Delta^{\text{U}}. \tag{D.10}$$

Moreover, the damper header is incremented when the packet is passing by a JCS. Therefore, by (7.12) and accounting for the errors in the computation of damper header, we have:

$$H = \sum_{i=1}^{K} (\delta_i - d_i^{\mathscr{H}_i} + e) = \sum_{i=1}^{K} \delta_i - \sum_{i=1}^{K} d_i^{\mathscr{H}_i} + Ke, \tag{D.11}$$

where e is defined in Section 7.3.3. Using (D.11) and the upper bound in (D.10), (D.9) gives:

$$\begin{aligned} d^{\mathscr{H}_{\text{TAI}}} &\leq \sum_{i=1}^{K} d_i^{\mathscr{H}_i} + \sum_{j=1}^{K'} \pi_j^{\mathscr{H}_{\text{TAI}}} + \left(\sum_{i=1}^{K} \delta_i - \sum_{i=1}^{K} d_i^{\mathscr{H}_i} + Ke \right) \\ &\quad + \Delta^{\text{U}} + \gamma = \sum_{i=1}^{K} \delta_i + \sum_{j=1}^{K'} \pi_j^{\mathscr{H}_{\text{TAI}}} + \Delta^{\text{U}} + Ke + \gamma. \end{aligned} \tag{D.12}$$

By Lemma D.3, $\gamma \leq \overline{\psi}$; also $|e| \leq \epsilon$; furthermore, the delay of the packet inside the BDS j is less than its worst-case delay, i.e., $\pi_{j,\text{worst}}^{\mathscr{H}_{\text{TAI}}}$. Therefore:

$$d^{\mathscr{H}_{\text{TAI}}} \leq \sum_{i=1}^{K} \delta_i + \sum_{j=1}^{K'} \pi_{j,\text{worst}}^{\mathscr{H}_{\text{TAI}}} + \Delta^{\text{U}} + K\epsilon + \overline{\psi}.$$

Similarly, using the lower bound in (D.10) and the value of H in (D.11), we have:

$$\begin{aligned} d^{\mathscr{H}_{\text{TAI}}} &\geq \sum_{i=1}^{K} d_i^{\mathscr{H}_i} + \sum_{j=1}^{K'} \pi_j^{\mathscr{H}_{\text{TAI}}} + \left(\sum_{i=1}^{K} \delta_i - \sum_{i=1}^{K} d_i^{\mathscr{H}_i} + Ke \right) \\ &\quad - \Delta^{\text{L}} + \gamma = \sum_{i=1}^{K} \delta_i + \sum_{j=1}^{K'} \pi_j^{\mathscr{H}_{\text{TAI}}} - \Delta^{\text{L}} + Ke + \gamma. \end{aligned} \tag{D.13}$$

By Lemma D.3, $\gamma \geq -\underline{\psi}$; also, $|e| \leq \epsilon$; furthermore, the delay of the packet inside the BDS j is less than its best-case delay, i.e., $\pi_{j,\text{best}}^{\mathscr{H}_{\text{TAI}}}$. Therefore:

$$d^{\mathscr{H}_{\text{TAI}}} \geq \sum_{i=1}^{K} \delta_i + \sum_{j=1}^{K'} \pi_{j,\text{best}}^{\mathscr{H}_{\text{TAI}}} - \Delta^{\text{L}} - K\epsilon - \underline{\psi}_i.$$

By subtracting the lower and upper bounds in (D.13) and (D.14), we obtain a jitter bound for the block:

$$
\begin{aligned}
V = \sum_{j=1}^{K'} \left(\pi^{\mathscr{H}_{\text{TAI}}}_{j,\text{worst}} - \pi^{\mathscr{H}_{\text{TAI}}}_{j,\text{best}}\right) + \Delta^{\text{U}} + \Delta^{\text{L}} + 2K\epsilon + \overline{\psi} \\
+ \underline{\psi} = \sum_{j=1}^{K'} \nu_j^{\mathscr{H}_{\text{TAI}}} + \Delta^{\text{U}} + \Delta^{\text{L}} + 2K\epsilon + \overline{\psi} + \underline{\psi},
\end{aligned}
\tag{D.14}
$$

which proves the jitter bound. Note that $\pi^{\mathscr{H}_{\text{TAI}}}_{j,\text{worst}} - \pi^{\mathscr{H}_{\text{TAI}}}_{j,\text{best}} \le \nu_j^{\mathscr{H}_{\text{TAI}}}$.

Moreover, since $\pi^{\mathscr{H}_{\text{TAI}}}_{j,\text{worst}} \le \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}}$ and $\pi^{\mathscr{H}_{\text{TAI}}}_{j,\text{best}} \ge \underline{\pi}_j^{\mathscr{H}_{\text{TAI}}}$, we have:

$$
\begin{aligned}
d^{\mathscr{H}_{\text{TAI}}} &\le \sum_{i=1}^{K} \delta_i + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + \Delta^{\text{U}} + K\epsilon + \overline{\psi} = \overline{D}, \\
d^{\mathscr{H}_{\text{TAI}}} &\ge \sum_{i=1}^{K} \delta_i + \sum_{j=1}^{K'} \underline{\pi}_j^{\mathscr{H}_{\text{TAI}}} - \Delta^{\text{L}} - K\epsilon - \underline{\psi} = \underline{D}.
\end{aligned}
\tag{D.15}
$$

Proof of tightness. Consider a packet that enters JCS 1. We show scenarios where the packet reaches the delay bound in the statement of the theorem; since $\overline{\psi}$ can take different values due to the $\min(.)$ function, we give two different scenarios.

First scenario. Assume that the clocks $\mathscr{H}_i$, $i = 1, \ldots, K$, and $\mathscr{H}_{\text{damper}}$ are adversarial and faster than TAI such that for any delay measurement d we have:

$$
\begin{aligned}
d^{\mathscr{H}_{\text{TAI}}} &= \rho d^{\mathscr{H}_i} + \eta, \quad i = 1, \ldots, K \\
d^{\mathscr{H}_{\text{TAI}}} &= \rho d^{\mathscr{H}_{\text{damper}}} + \eta.
\end{aligned}
\tag{D.16}
$$

This scenario assumes that for any clock $\mathscr{H}_i$, $\rho d^{\mathscr{H}_i} + \eta \le d^{\mathscr{H}_i} + 2\omega$ and $\rho d^{\mathscr{H}_{\text{damper}}} + \eta \le d^{\mathscr{H}_{\text{damper}}} + 2\omega$.

Let us define H_i as the damper header that is computed in JCS i. Then, let the packet experience a delay $d_i^{\mathscr{H}_i} \le \delta_i$ in JCS i in its local time; then the damper header written for this packet is

$$
\begin{aligned}
H_1 &= \delta_1 - d_1^{\mathscr{H}_1} + \epsilon, \\
H_i &= H_{i-1} + \delta_i - d_i^{\mathscr{H}_i} + \epsilon, \quad i = 2, \ldots, K
\end{aligned}
$$

assuming adversarial condition in the damper header computation error. The packet experiences a delay equal to $\overline{\pi}_j^{\mathscr{H}_{\text{TAI}}}$ in BDS j. Finally, the damper computes the eligibility time and releases it at the latest; i.e., the packet experiences a delay

$$
t^{\mathscr{H}_{\text{damper}}} = H_K + \Delta^{\text{U}} = \sum_{u=1}^{K} \delta_u - \sum_{u=1}^{K} d_u^{\mathscr{H}_u} + K\epsilon + \Delta^{\text{U}}.
$$

Therefore, for the damper, the delay of the packet in TAI, using (D.16), is:

$$t^{\mathscr{H}_{\text{TAI}}} = \rho\left(\sum_{u=1}^{K}\delta_u - \sum_{u=1}^{K} d_u^{\mathscr{H}_u} + K\epsilon + \Delta^{\text{U}}\right) + \eta. \tag{D.17}$$

Also, for JCS i, the delay of the packet in TAI is:

$$d_i^{\mathscr{H}_{\text{TAI}}} = \rho d_i^{\mathscr{H}_i} + \eta. \tag{D.18}$$

Now, to compute the per-hop delay of the packet, we sum up all the delays in TAI:

$$\begin{aligned} d_{\text{hop}}^{\mathscr{H}_{\text{TAI}}} &= \sum_{u=1}^{K} d_u^{\mathscr{H}_{\text{TAI}}} + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + t^{\mathscr{H}_{\text{TAI}}} = \sum_{u=1}^{K}\left(\rho d_u^{\mathscr{H}_i} + \eta\right) \\ &+ \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + \rho\left(\sum_{u=1}^{K}\delta_u - \sum_{u=1}^{K} d_u^{\mathscr{H}_u} + K\epsilon + \Delta^{\text{U}}\right) + \eta \\ &= \rho\left(\sum_{u=1}^{K}\delta_u + K\epsilon + \Delta^{\text{U}}\right) + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + (K+1)\eta = \sum_{u=1}^{K}\delta_u + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + \Delta^{\text{U}} + K\epsilon + \overline{\psi}, \end{aligned} \tag{D.19}$$

which is equal to the delay bound in the statement of the theorem.

Second scenario. Assume that the clocks $\mathscr{H}_i$, $i = 1, ..., K$, and $\mathscr{H}_{\text{damper}}$ are adversarial and faster than TAI such that for any delay measurement d we have:

$$\begin{aligned} d^{\mathscr{H}_{\text{TAI}}} &= d^{\mathscr{H}_i} + 2\omega, \quad i = 1, ..., K \\ d^{\mathscr{H}_{\text{TAI}}} &= d^{\mathscr{H}_{\text{damper}}} + 2\omega. \end{aligned} \tag{D.20}$$

This scenario assumes that for any clock $\mathscr{H}_i$, $\rho d^{\mathscr{H}_i} + \eta \geq d^{\mathscr{H}_i} + 2\omega$ and $\rho d^{\mathscr{H}_{\text{damper}}} + \eta \geq d^{\mathscr{H}_{\text{damper}}} + 2\omega$. Then following the same steps as the previous scenario, we have:

$$\begin{aligned} t^{\mathscr{H}_{\text{TAI}}} &= \sum_{u=1}^{K}\delta_u - \sum_{u=1}^{K} d_u^{\mathscr{H}_u} + K\epsilon + \Delta^{\text{U}} + 2\omega, \\ d_i^{\mathscr{H}_{\text{TAI}}} &= d_i^{\mathscr{H}_i} + 2\omega, \quad \forall i \in \{1, \ldots, K\}. \end{aligned} \tag{D.21}$$

This gives:

$$\begin{aligned} d_{\text{hop}}^{\mathscr{H}_{\text{TAI}}} &= \sum_{u=1}^{K} d_u^{\mathscr{H}_{\text{TAI}}} + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + t^{\mathscr{H}_{\text{TAI}}} = \sum_{u=1}^{K}\left(d_u^{\mathscr{H}_i} + 2\omega\right) \\ &+ \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + \sum_{u=1}^{K}\delta_u - \sum_{u=1}^{K} d_u^{\mathscr{H}_u} + K\epsilon + \Delta^{\text{U}} + 2\omega \\ &= \sum_{u=1}^{K}\delta_u + K\epsilon + \Delta^{\text{U}} + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + 2(K+1)\omega = \sum_{u=1}^{K}\delta_u + \sum_{j=1}^{K'} \overline{\pi}_j^{\mathscr{H}_{\text{TAI}}} + \Delta^{\text{U}} + K\epsilon + \overline{\psi}, \end{aligned} \tag{D.22}$$

which is equal to the delay bound in the statement of the theorem. The tightness for delay lower bound happens when the clocks $\mathcal{H}_i$, $i = 1, \ldots, K$, and $\mathcal{H}_{\text{damper}}$ are adversarial and slower than TAI. Similarly to the tightness proof of delay upper-bound, two tightness scenarios are given for the two possible values of $\underline{\psi}$. For the first scenario, for any delay measurement d, we have:

$$
\begin{aligned}
d^{\mathcal{H}_{\text{TAI}}} &= \frac{1}{\rho}\left(d^{\mathcal{H}_i} - \eta\right), \quad i = 1, \ldots, K, \\
d^{\mathcal{H}_{\text{TAI}}} &= \frac{1}{\rho}\left(d^{\mathcal{H}_{\text{damper}}} - \eta\right),
\end{aligned}
\tag{D.23}
$$

and for the second scenario, we have:

$$
\begin{aligned}
d^{\mathcal{H}_{\text{TAI}}} &= d^{\mathcal{H}_i} - 2\omega, \quad i = 1, \ldots, K, \\
d^{\mathcal{H}_{\text{TAI}}} &= d^{\mathcal{H}_{\text{damper}}} - 2\omega.
\end{aligned}
\tag{D.24}
$$

Considering the damper releases the packets at the earliest, the rest of the proof follows the same steps as the tightness proof of delay upper-bound.

Lemma D.3. $-\underline{\psi} \le \gamma \le \overline{\psi}$*, where* $\underline{\psi}$ *and* $\overline{\psi}$ *are defined in* (7.14)*:*

Proof. By definition of γ, we have:

$$
\gamma = \sum_{j=1}^{K}\left(d_j^{\mathcal{H}_{\text{TAI}}} - d_j^{\mathcal{H}_j}\right) + \left(t^{\mathcal{H}_{\text{TAI}}} - t^{\mathcal{H}_{\text{damper}}}\right).
\tag{D.25}
$$

First we prove the upper bound. By (7.1) the following holds for any JCS j:

$$
\begin{aligned}
d_j^{\mathcal{H}_{\text{TAI}}} - d_j^{\mathcal{H}_j} &\le (\rho - 1) d_j^{\mathcal{H}_j} + \eta, \\
d_j^{\mathcal{H}_{\text{TAI}}} - d_j^{\mathcal{H}_j} &\le 2\omega.
\end{aligned}
\tag{D.26}
$$

Using the value of H in (D.11) and the upper bound in (D.10), we have $t^{\mathcal{H}_{\text{damper}}} \le \sum_{j=1}^{K}\left(\delta_j - d_j^{\mathcal{H}_j} + \epsilon\right) + \Delta^{\text{U}}$. Hence, similarly to the previous equation:

$$
\begin{aligned}
t^{\mathcal{H}_{\text{TAI}}} - t^{\mathcal{H}_{\text{damper}}} &\le (\rho - 1)\left(\Delta^{\text{U}} + \sum_{j=1}^{K}\left(\delta_j - d_j^{\mathcal{H}_j} + \epsilon\right)\right) + \eta, \\
t^{\mathcal{H}_{\text{TAI}}} - t^{\mathcal{H}_{\text{damper}}} &\le 2\omega.
\end{aligned}
\tag{D.27}
$$

We first consider the case that the synchronization inequality is dominating for all systems (i.e., the second line of (D.26) and (D.27)). Hence, we have:

$$
\gamma \le \sum_{j=1}^{K} 2\omega + 2\omega = 2(K + 1)\omega.
\tag{D.28}
$$

Second, we consider the case that the free-running mode is dominating for all systems (i.e., in the first line of (D.26) and (D.27)). Then, we have:

$$\begin{aligned}\gamma &\leq \sum_{j=1}^{K}\left((\rho-1)d_j^{\mathcal{H}_j}+\eta\right)+(\rho-1)\Delta^{\mathrm{U}}\\ &+(\rho-1)\left(\sum_{j=1}^{K}\left(\delta_j-d_j^{\mathcal{H}_j}+\epsilon\right)\right)+\eta\\ &=(\rho-1)\left(\Delta^{\mathrm{U}}+\sum_{j=1}^{K}(\delta_j+\epsilon)\right)+(K+1)\eta. \end{aligned}\tag{D.29}$$

Finally, by (D.28) and (D.29), $\gamma \leq \overline{\psi}$.

Next, we prove the lower bound. Similarly, by (7.1) the following holds for any JCS j:

$$\begin{aligned}&d_j^{\mathcal{H}_{\mathrm{TAI}}}-d_j^{\mathcal{H}_j}\geq -\left(1-\frac{1}{\rho}\right)d_j^{\mathcal{H}_j}-\frac{\eta}{\rho},\\ &d_j^{\mathcal{H}_{\mathrm{TAI}}}-d_j^{\mathcal{H}_j}\geq -2\omega.\end{aligned}\tag{D.30}$$

Using the value of H in (D.11) and the upper bound in (D.10), we have

$$t^{\mathcal{H}_{\mathrm{damper}}}\geq \sum_{j=1}^{K}\left(\delta_j-d_j^{\mathcal{H}_j}-\epsilon\right)-\Delta^{\mathrm{L}}.$$

Hence:

$$\begin{aligned}&t^{\mathcal{H}_{\mathrm{TAI}}}-t^{\mathcal{H}_{\mathrm{damper}}}\geq -(1-\frac{1}{\rho})(-\Delta^{\mathrm{L}}+\sum_{j=1}^{K}(\delta_j-d_j^{\mathcal{H}_j}-\epsilon))-\frac{\eta}{\rho},\\ &t^{\mathcal{H}_{\mathrm{TAI}}}-t^{\mathcal{H}_{\mathrm{damper}}}\geq -2\omega.\end{aligned}\tag{D.31}$$

We first consider the case that the synchronization inequality is dominating for all systems (i.e., the second line of (D.26) and (D.27)). Hence, we have:

$$\gamma \geq -\sum_{j=1}^{K}2\omega-2\omega=-2(K+1)\omega.\tag{D.32}$$

Second, we consider the case that the free-running mode is dominating for all systems (i.e., in the first line of (D.26) and (D.27)). Then, we have:

$$\begin{aligned}\gamma &\geq -\left(1-\frac{1}{\rho}\right)\sum_{j=1}^{K}d_j^{\mathcal{H}_j}-\frac{K\eta}{\rho}-\left(1-\frac{1}{\rho}\right)\left(-\Delta^{\mathrm{L}}+\sum_{j=1}^{K}\left(\delta_j-d_j^{\mathcal{H}_j}-\epsilon\right)\right)-\frac{\eta}{\rho}\\ &=-\left(1-\frac{1}{\rho}\right)\left(-\Delta^{\mathrm{L}}+\sum_{j=1}^{K}\left(\delta_j-\epsilon\right)\right)-\frac{(K+1)\eta}{\rho}.\end{aligned}\tag{D.33}$$

Finally, by (D.32) and (D.33), $\gamma \geq -\underline{\psi}$. □

D.3 Proof of Theorem 7.2

Consider a packet that enters block 1 at time A. Let $\tilde{E}_i$ and E_i denote respectively the theoretical and actual eligibility time of the packet from the damper of block i. Then, the delay from A to the output of block N, i.e., E_N, is:

$$d_{\text{e2e}}^{\mathcal{H}_{\text{TAI}}} = E_N^{\mathcal{H}_{\text{TAI}}} - A^{\mathcal{H}_{\text{TAI}}} = \left(\tilde{E}_1^{\mathcal{H}_{\text{TAI}}} - A^{\mathcal{H}_{\text{TAI}}}\right) + \sum_{i=2}^{N-1} \left(\tilde{E}_i^{\mathcal{H}_{\text{TAI}}} - \tilde{E}_{i-1}^{\mathcal{H}_{\text{TAI}}}\right) + \left(E_N^{\mathcal{H}_{\text{TAI}}} - \tilde{E}_{N-1}^{\mathcal{H}_{\text{TAI}}}\right). \tag{D.34}$$

By Theorem 7.1 and setting the tolerances to zero, we can find delay and jitter bounds for $\left(\tilde{E}_1^{\mathcal{H}_{\text{TAI}}} - A^{\mathcal{H}_{\text{TAI}}}\right)$ in TAI. For any term $\left(\tilde{E}_i^{\mathcal{H}_{\text{TAI}}} - \tilde{E}_{i-1}^{\mathcal{H}_{\text{TAI}}}\right)$ in the summation, using Lemma D.4, we can obtain delay and jitter bound by setting $\Delta_i^{\text{L}} = \Delta_i^{\text{U}} = 0$ (since we are interested to find the delay to the theoretical eligibility time at the damper i). Finally, we can apply again Lemma D.4 to get the bounds for $\left(E_N^{\mathcal{H}_{\text{TAI}}} - \tilde{E}_{N-1}^{\mathcal{H}_{\text{TAI}}}\right)$. By summing up the delay and jitter bounds of each term in (D.34), we get the bounds in the statement of the theorem.

Lemma D.4. *Consider a block i in Fig. 7.5 that has K_i JCSs. Assume that TE time-stamping is used to compute damper headers. Let δ_{blk_i} denote the sum of the delay bounds of the JCSs in the block, $\underline{\pi}_{\text{blk}_i}^{\mathcal{H}_{\text{TAI}}}, \overline{\pi}_{\text{blk}_i}^{\mathcal{H}_{\text{TAI}}}$ and $v_{\text{blk}_i}^{\mathcal{H}_{\text{TAI}}}$ respectively denote the sum of delay lower and upper bounds and jitter bound of the BDSs. Then, the delay of a packet from theoretical eligibility time of damper $i-1$, $i = 2, \ldots, N$, to the actual eligibility time of damper i, in TAI, is upper-bounded by $\overline{D}_i$, low-bounded by $\underline{D}_i$ and has jitter bound V_i:*

$$\begin{aligned}
\overline{D}_i &= \delta_{\text{blk}_i} + \overline{\pi}_{\text{blk}_i}^{\mathcal{H}_{\text{TAI}}} + \Delta_{i-1}^{\text{U}} + \Delta_i^{\text{U}} + K_i\epsilon + \overline{\psi}'_i,\\
\underline{D}_i &= \delta_{\text{blk}_i} + \underline{\pi}_{\text{blk}_i}^{\mathcal{H}_{\text{TAI}}} + \Delta_{i-1}^{\text{U}} - \Delta_i^{\text{L}} - K_i\epsilon - \underline{\psi}'_i,\\
V_i &= v_{\text{blk}_i}^{\mathcal{H}_{\text{TAI}}} + \Delta_i^{\text{U}} + \Delta_i^{\text{L}} + 2K_i\epsilon + \overline{\psi}_i + \underline{\psi}'_i,
\end{aligned}$$

where,

$$\begin{aligned}
\overline{\psi}'_i &= \min\Big((\rho-1)\left(\delta_{\text{blk}_i} + \Delta_{i-1}^{\text{U}} + \Delta_i^{\text{U}} + K_i\epsilon\right) + (K_i+1)\eta, 2(K_i+1)\omega\Big),\\
\underline{\psi}'_i &= \min\left(\left(1-\frac{1}{\rho}\right)\left(\delta_{\text{blk}_i} + \Delta_{i-1}^{\text{U}} - \Delta_i^{\text{L}} - K_i\epsilon\right) + \frac{(K_i+1)\eta}{\rho}, 2(K_i+1)\omega\right).
\end{aligned} \tag{D.35}$$

Proof. The proof follows the same as Theorem 7.1 and using

$$H = (\delta_1 + \Delta_{i-1}^{\text{U}} - d_1^{\mathcal{H}_1} + e) + \sum_{j=2}^{K} (\delta_j - d_j^{\mathcal{H}_j} + e) = \sum_{j=1}^{K} \delta_j - \sum_{i=1}^{K} d_i^{\mathcal{H}_i} + \Delta_{i-1}^{\text{U}} + Ke, \tag{D.36}$$

instead of (D.11) for damper header computation. Note that here $d_j^{\mathcal{H}_j}$ is the delay of the packet from start of time stamping at JCS j to its departure time. □

D.4 Proof of Theorem 7.3

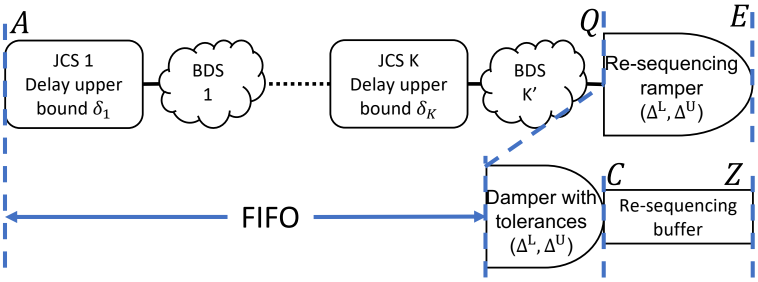

A
JCS 1
Delay upper bound δ_1
BDS 1
JCS K
Delay upper bound δ_K
BDS K'
Q
E
Re-sequencing ramper $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$
FIFO
Damper with tolerances $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$
C
Z
Re-sequencing buffer

Figure D.1: The notations used in the proof of Theorem 7.3.

Consider Fig. D.1 where A is the sequence of packet arrival times at entrance of the block, Q is the sequence of arrivals to the re-sequencing damper, and E is the sequence of actual eligibility times at the re-sequencing damper. By definition, the re-sequencing damper behaves as a damper with tolerances $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$ followed by a re-sequencing buffer. Denote with C the actual eligibility times from the damper with tolerance and with Z the output times of the re-sequencing buffer; the equivalence means that $E = Z$.

By Theorem 7.1, the delay from A to C has lower bound $\underline{D}$, upper bound $\overline{D}$ and the jitter bound is V. Now, let $\mathscr{P}$ and $\mathscr{P}'$ be the sets of packets seen respectively at A and Q, i.e., $\mathscr{P}' \subseteq \mathscr{P}$; by defining $\mathscr{P}'$, we only include the packets of $\mathscr{P}$ that arrive to Q; therefore, by [64, Theorem 4], the re-sequencing buffer does not increase the worst-case delay and jitter from A to C. Hence, the delay and jitter bounds from A to Z are the same as from A to C, which proves the theorem.

D.5 Proof of Theorem 7.4

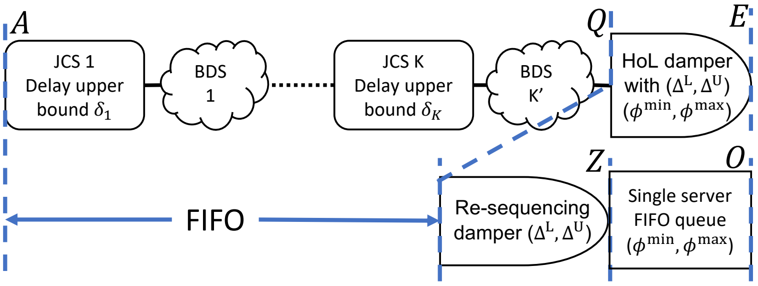

A
JCS 1
Delay upper bound δ_1
BDS 1
JCS K
Delay upper bound δ_K
BDS K'
Q
E
HoL damper with $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$ $(\phi^{\min}, \phi^{\max})$
FIFO
Z
O
Re-sequencing damper $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$
Single server FIFO queue $(\phi^{\min}, \phi^{\max})$

Figure D.2: The notations used in the proof of Theorem 7.4.

Consider Fig. D.2 where A is the sequence of packet arrival times at JCS 1 (entrance of the block), Q is the sequence of arrivals to the head-of-line damper, $\tilde{E}$ and E are the sequences of theoretical and actual eligibility times at the head-of-line damper. By Lemma D.5, the head-of-line damper is equivalent to a re-sequencing damper followed by a single-server FIFO queue with service time in range $[\phi^{\min}, \phi^{\max}]$. Denote with Z the output times of the re-sequencing damper and O as the output of the single-server FIFO queue; the equivalence means that $E = O$.

By Theorem 7.3, the delay from A to Z has lower bound $\underline{D}$, upper bound $\overline{D}$ and the jitter bound is V. Also, similarly to Corollary 7.1, arrival curve at Z is $\alpha_{\text{reseq}}(t) = \alpha(t+V)$.

By Lemma D.6, the delay upper-bound of the single-server FIFO queue is

$$\max_{k\in\mathbb{N}}\left\{k\phi^{\max} - \alpha^{\downarrow}_{\text{reseq}}(k)\right\}$$

for nonzero processing time, i.e. $\phi^{\max} > 0$; otherwise the delay upper-bound is zero. By [[183], Proposition 7], we obtain $\alpha^{\downarrow}_{\text{reseq}}(k) \geq \alpha^{\downarrow}(k) - V$. Therefore, the delay is upper-bounded by $\theta = \max_{k\in\mathbb{N}}\left\{k\phi^{\max} - \alpha^{\downarrow}(k) + V\right\}$ for nonzero processing time, i.e., $\phi^{\max} > 0$.

We can see that the delay lower bound is $\phi^{\min}$ as minimum processing time for a packet. This gives the jitter $V_{\text{SSQ}} = \theta - \phi^{\min}$. By summing the bounds from A to Z and the single-server FIFO queue, we obtain the bounds in the statement of the theorem.

Lemma D.5. *Consider a head-of-line damper shown in Fig. D.2. Then, this system can be abstracted as a re-sequencing damper with tolerances* $(\Delta^{\text{L}}, \Delta^{\text{U}})$ *followed by a single-server FIFO queue with service time in range* $[\phi^{\min}, \phi^{\max}]$.

Proof. We use the notation in Fig. D.2. Let E be the sequence of actual eligibility times at the head-of-line damper and $\tilde{E}$ the sequence of theoretical eligibility times. By the discussion that follows (7.9) and by Lemma D.2, there exist sequences $\bar{E}$ and ϕ such that, for every n:

$$\begin{aligned} \phi_n &\in [\phi^{\min}, \phi^{\max}],\\ \tilde{E}_n - \Delta^{\text{L}} \leq &\bar{E}_n \leq \tilde{E}_n + \Delta^{\text{U}},\\ E_n &= \max(\bar{E}_n, E_{n-1}) + \phi_n. \end{aligned} \tag{D.37}$$

Let us construct a re-sequencing damper with tolerances $(\Delta^{\text{L}}, \Delta^{\text{U}})$ and sequence of actual eligibility times Z such that

$$\begin{aligned} \tilde{E}_n - \Delta^{\text{L}} \leq \bar{E}_n \leq \tilde{E}_n + \Delta^{\text{U}},\\ Z_1 = \bar{E}_1,\ Z_n = \max\left\{\bar{E}_n, Z_{n-1}\right\}. \end{aligned} \tag{D.38}$$

Now consider a single-server FIFO queue with sequence of service times equal to ϕ and input sequence Z. Then, the output sequence from the FIFO queue, O, is

$$O_1 = Z_1 + \phi_1, O_n = \max(O_{n-1}, Z_n) + \phi_n; n \geq 2. \tag{D.39}$$

We now show by induction that $O_n = E_n$ for every $n \geq 1$. This holds for $n = 1$. Assume that it holds for $n-1$. Observe first that $Z_{n-1} \leq O_{n-1}$ (because $\phi_n \geq 0$) and therefore, by the induction hypothesis,

$$Z_{n-1} \le E_{n-1}. \tag{D.40}$$

By (D.39) and again the induction hypothesis:

$$O_n = \max(E_{n-1}, Z_n) + \phi_n. \tag{D.41}$$

By (D.38):

$$O_n = \max(E_{n-1}, \bar{E}_n, Z_{n-1}) + \phi_n = \max(E_{n-1}, \bar{E}_n) + \phi_n, \tag{D.42}$$

because of (D.40). It follows from (D.37) that $O_n = E_n$. □

Lemma D.6. *Consider a flow with per-packet arrival curve α that enters a single-server FIFO queue. Suppose that a head of line packet n has a processing time $\phi_n \in [\phi^{\min}, \phi^{\max}]$. Then a delay bound of the flow θ is:*

$$\theta = \max_{i \in \mathbb{N}} \left\{ i\phi^{\max} - \alpha^{\downarrow}(i) \right\}. \tag{D.43}$$

Proof. Let us call I_n and O_n as arrival and departure times of packet n. By Lemma D.7, we have:

$$O_n = \max_{m \le n} \left\{ I_m + \sum_{i=m}^{n} \phi_i \right\}. \tag{D.44}$$

We subtract I_n from both sides:

$$O_n - I_n = \max_{m \le n} \left\{ I_m + \sum_{i=m}^{n} \phi_i \right\} - I_n = \max_{m \le n} \left\{ \sum_{i=m}^{n} \phi_i - (I_n - I_m) \right\}. \tag{D.45}$$

By (3.21), we have $I_n - I_m \ge \alpha^{\downarrow}(n - m + 1)$; therefore,

$$O_n - I_n \le \max_{m \le n} \left\{ \sum_{i=m}^{n} \phi_i - \alpha^{\downarrow}(n - m + 1) \right\}. \tag{D.46}$$

Since $\phi_i \le \phi^{\max}$:

$$O_n - I_n \le \max_{m \le n} \left\{ (m - n + 1)\phi^{\max} - \alpha^{\downarrow}(n - m + 1) \right\} \le \max_{i \in \mathbb{N}} \left\{ i\phi^{\max} - \alpha^{\downarrow}(i) \right\} = \theta. \tag{D.47}$$

□

Lemma D.7. *Consider a single-server FIFO queue. A packet n arrives at time I_n and the service time is ϕ_n when it is at the head of the queue. Then, the departure time of packet n is O_n and* *computed as:*

$$O_n = \max_{m \le n} \left\{ I_m + \sum_{i=m}^{n} \phi_i \right\}. \tag{D.48}$$

Proof. Since we have for a single-server FIFO queue:

$$O_1 = I_1 + \phi_1, \;\; O_n = \max(I_n, O_{n-1}) + \phi_n, \tag{D.49}$$

by Lemma D.1 the statement is proven. ☐

D.6 Proof of Theorem 7.5

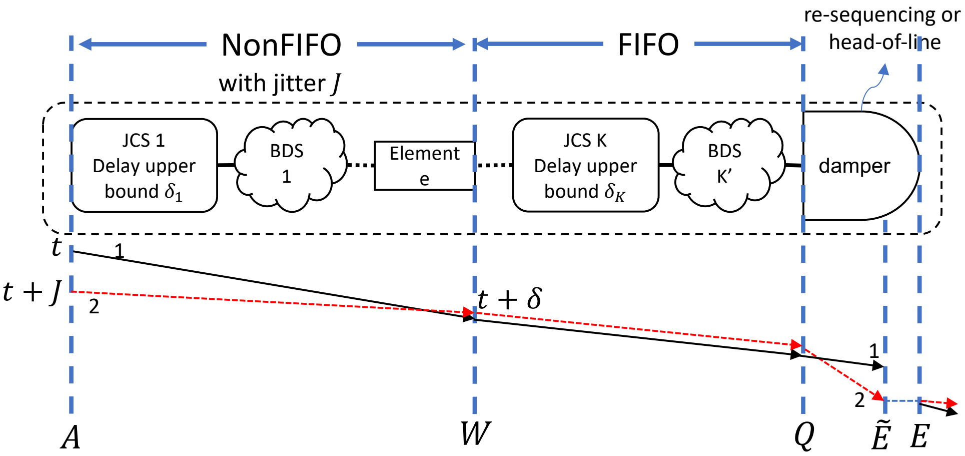


Figure D.3: The notations used in the proof of Theorem 7.5.

Consider Fig. D.3 where A is used to denote the arrival times to JCS 1, W the departure times from element e, Q the arrival times to the damper, $\tilde{E}$ and E respectively the theoretical and actual eligibility times at the damper. Now for packets $m, n : m < n$, since from output of element e to the input of the damper is FIFO, we have $W_m \le W_n$. Then, as the jitter from JCS 1 to element e is bounded by J, we have:

$$\begin{aligned} &(W_n - A_n) - (W_m - A_m) \le J \\ \text{thus } &A_m - A_n \le J + W_m - W_n \le J \\ \text{thus } &A_n \ge A_m - J. \end{aligned} \tag{D.50}$$

Now by (7.5), we have:

$$\begin{aligned} &\tilde{E}_n - \Delta^{\mathrm{L}} \le \bar{E}_n \le \tilde{E}_n + \Delta^{\mathrm{U}}, \\ &E_1 = \bar{E}_1, \;\; E_n = \max\left\{\bar{E}_n, E_{n-1}\right\}. \end{aligned} \tag{D.51}$$

By definition, the re-sequencing damper behaves as a damper with tolerances $(\Delta^{\mathrm{L}}, \Delta^{\mathrm{U}})$ followed by a re-sequencing buffer, where $\bar{E}$ is the output of the damper with tolerance and E is the output of the re-sequencing buffer. Now, let n be fixed and define packet index u as

$$u = \max\left\{k \leq n | \bar{E}_k = \max_{j \leq n}\{\bar{E}_j\}\right\}. \tag{D.52}$$

Then, we have $E_n - A_n = \bar{E}_u - A_n$. Combining with (D.50) for packets u and n, we obtain:

$$E_n - A_n \leq \bar{E}_u - A_u + J. \tag{D.53}$$

Since $\bar{E}$ is the output of the damper with tolerance, by Theorem 7.1, $\bar{E}_u - A_u \leq \overline{D}$; hence $E_n - A_n \leq \overline{D} + J$, which proves the delay upper bound. By (D.51) and Theorem 7.1, we have:

$$E_n - A_n \geq \bar{E}_n - A_n \geq \underline{D}, \tag{D.54}$$

which proves the delay lower bound for this case. Since the delay lower-bound is not changed, and the upper-bound is increased by J, hence the jitter bound is increased by J.

Proof of tightness. The tightness scenario for delay lower-bound is exactly the same as tightness scenario in Theorem 7.1 where a single packet in isolation reaches the delay lower-bound $\underline{D}$.

For the delay upper-bound and jitter bound tightness, consider two packets 1 and 2 as shown in Fig D.3 that arrive at times t and $t + J$ in TAI. Assume every local clock (of JCS or damper) $\mathcal{H}_i$ is adversarial and faster than TAI such that for any delay measurement d, we have:

$$d^{\mathcal{H}_{\mathrm{TAI}}} = \min\left(\rho d^{\mathcal{H}_i} + \eta, d^{\mathcal{H}_i} + 2\omega\right).$$

Let us call the worst-case delay, in TAI, from input of JCS 1 to the output of e, as δ. Suppose that packets 1 and 2 experience delays of δ and $\delta - J$, in TAI, to leave element e, i.e., $W_1 = W_2 = t + \delta$, and packet 2 arrives just before packet 1. Also, both experience the same delay from output of element e to the input of the damper while packet 2 is still prior to packet 1 due to the FIFO assumption. Since packet 1 arrives after packet 2, it is released after packet 2 becomes eligible (even if its theoretical eligibility time has passed).

Now, assume packet 2 experiences a delay, in TAI, equal to $\overline{D}$ from A to E, i.e., $E_2 = A_2 + \overline{D} = t + J + \overline{D}$ (the packet that reaches the upper-bound in tightness scenario of Theorem 7.1). Therefore, packet 1 is released after packet 2, i.e., $E_1 = E_2 = t + J + \overline{D}$. Hence, the delay of packet 1 from A to E is:

$$E_1 - A_1 = (t + J + \overline{D}) - (t) = J + \overline{D}, \tag{D.55}$$

that is equal to the delay upper-bound of the theorem statement.

Now, we verify the assumptions;

1) The jitter from JCS 1 to the output of e is not larger that J: The delay of packet 1 and packet 2 are respectively δ and $\delta - J$ in TAI, and therefore the jitter is J.

2) The FIFO constraint of the damper is not violated: Packet 2 arrives before packet 1, $Q_2 \le Q_1$, and also leaves before is $E_2 \le E_1$.

Hence, we showed an execution trace that with packet 2 experiencing a delay of $\overline{D}$, packet 1 experiences a delay of $\overline{D} + J$. Since there execution traces where in one, a packet reaches the lower-bound of Theorem 7.1 and in another one, a packet reaches the delay upper-bound of Theorem 7.1 plus J, we have that the jitter Theorem 7.1 is increased by J.

D.7 Proof of Theorem 7.6

Consider Fig. D.3 where A is used to denote the sequence of arrival times to JCS 1, W the departure times from element e, Q the arrival times to the damper, $\tilde{E}$ and E respectively the theoretical and actual eligibility times at the damper. By Lemma D.5, we abstract it as a re-sequencing damper followed by a single-server FIFO queue. Let Z denote the sequence of departure times from the re-sequencing damper; then, by Theorem 7.5, for a packet n, we have:

$$\underline{D} \le Z_n - A_n \le \overline{D} + J, \tag{D.56}$$

and the jitter from A to Z is $V + J$. As a result, by Lemma 6.1, an arrival curve at the output of the re-sequencing damper (input of the single-server FIFO queue) is $\alpha_{\text{reseq}} = \alpha(t + V + J)$. Then by Lemma D.6 for nonzero processing time:

$$E_n - Z_n \le \max_{k \in \mathbb{N}} \left\{ k\phi^{\max} - \alpha^{\downarrow}_{\text{reseq}}(k) \right\} \le \max_{k \in \mathbb{N}} \left\{ k\phi^{\max} - \alpha^{\downarrow}(k) + V + J \right\} = \theta + J, \tag{D.57}$$

where θ is defined in (7.28). Finally, by (D.56), we have,

$$E_n - A_n = (E_n - Z_n) + (Z_n - A_n) \le \overline{D} + J + (\theta + J) 1_{\{\phi^{\max} > 0\}}, \tag{D.58}$$

which proves the delay upper bound. Note that by Theorem 7.4, an upper-bound on the delay is $\overline{D} + \theta 1_{\{\phi^{\max} > 0\}}$.

Using minimum processing time and (D.56), we have:

$$E_n - A_n = (E_n - Z_n) + (Z_n - A_n) \ge \phi^{\min} + \underline{D}, \tag{D.59}$$

which proves the delay lower bound. Since the delay lower-bound is not changed, and the upper-bound is increased by $J + J 1_{\{\phi^{\max} > 0\}}$, the jitter bound is increased by $J + J 1_{\{\phi^{\max} > 0\}}$.

Bibliography

[1] S. M. Tabatabaee and J.-Y. Le Boudec, “Deficit round-robin: A second network calculus analysis,” in *2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS)*.

[2] A. Bouillard, M. Boyer, and E. Le Corronc, *Deterministic Network Calculus: From Theory to Practical Implementation*. Wiley-ISTE. [Online]. Available: https://www.wiley.com/en-au/Deterministic+Network+Calculus%3A+From+Theory+to+Practical+Implementation-p-9781119563419

[3] M. Manderscheid and F. Langer, “Network calculus for the validation of automotive ethernet in-vehicle network configurations,” in *2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2011, pp. 206–211.

[4] Use Cases - IEEE P802.1DG, V0.4. [Online]. Available: https://www.ieee802.org/1/files/public/docs2019/dg-pannell-automotive-use-cases-0919-v04.pdf

[5] Industrial Automation Traffic Types and their Mapping to QoS/TSN Mechanisms. [Online]. Available: https://www.ieee802.org/1/files/public/docs2018/60802-ademaj-traffic-type-characterization-1118-v01.pdf

[6] Aerospace Traffic Characterization. [Online]. Available: https://www.ieee802.org/1/files/public/docs2021/dp-Jabbar-et-all-Aerospace-Traffic-Characterization-0421-v02.pdf

[7] E. Grossman, “RFC8578: Deterministic Networking Use Cases,” May 2019. [Online]. Available: https://www.rfc-editor.org/info/rfc8578

[8] ITU-T, “ITU-T Y.3000-series - Representative use cases and key network requirements for Network 2030,” vol. Y.Sup67, 2020. [Online]. Available: https://www.itu.int/rec/T-REC-Y.Sup67-202007-I

[9] T. Wong, N. Finn, and X. Wang. TSN Profile for Service Provider Networks. [Online]. Available: https://www.ieee802.org/1/files/public/docs2018/new-tsn-wangtt-TSN-profile-for-service-provider-network-0718.pdf

[10] C. Mannweiler, B. Gajic, P. Rost, R. S. Ganesan, C. Markwart, R. Halfmann, J. Gebert, and A. Wich, “Reliable and deterministic mobile communications for industry 4.0: Key challenges and solutions for the integration of the 3gpp 5g system with IEEE,” in *Mobile Communication - Technologies and Applications; 24. ITG-Symposium*, 2019, pp. 1–6.

[11] C. P. Szydlowski, “CAN specification 2.0: Protocol and implementations,” 1992.

[12] ISO 17458-1, “Road Vehicles – FlexRay Communications System – Part 1: General Information and Use Case Definition,” 2013.

[13] J. A. R. De Azua and M. Boyer, “Complete modelling of AVB in network calculus framework,” in *the 22nd ACM Int’l Conf. on Real-Time Networks and Systems (RTNS)*, NY, USA, 2014, pp. 55–64.

[14] AS-2D2 Deterministic Ethernet and Unified Networking, *AS6802 Time-Triggered Ethernet*, nov 2016. [Online]. Available: https://doi.org/10.4271/AS6802

[15] G. Prytz, “A performance analysis of EtherCAT and PROFINET IRT,” in *2008 IEEE International Conference on Emerging Technologies and Factory Automation*, 2008, pp. 408–415.

[16] M. Boyer and C. Fraboul, “Tightening end to end delay upper bound for AFDX network calculus with rate latency FIFO servers using network calculus,” in *2008 IEEE International Workshop on Factory Communication Systems*, 2008, pp. 11–20.

[17] “IEEE Standard for Ethernet,” *IEEE Std 802.3-2018 (Revision of IEEE Std 802.3-2015)*, pp. 1–5600, 2018.

[18] “ISO/IEC/IEEE International Standard - Information technology – Telecommunications and information exchange between systems – Local and Metropolitan Area Networks – Specific requirements – Part 1BA: Audio video bridging (AVB) systems,” *ISO/IEC/IEEE 8802-1BA*, pp. 1–52, 2016.

[19] Time-sensitive networking (TSN) task group. [Online]. Available: https://1.ieee802.org/tsn/

[20] S. S. Craciunas, R. S. Oliver, and T. Ag, “An overview of scheduling mechanisms for time-sensitive networks,” *Proceedings of the French Summer School on Real-Time Systems-l’École d’Été Temps Réel (ETR)*, pp. 1551–3203, 2017.

[21] Deterministic Networking (detnet). [Online]. Available: https://datatracker.ietf.org/wg/detnet/about/

[22] J. Falk, D. Hellmanns, B. Carabelli, N. Nayak, F. Dürr, S. Kehrer, and K. Rothermel, “Nesting: Simulating ieee time-sensitive networking (tsn) in omnet++,” in *2019 International Conference on Networked Systems (NetSys)*, 2019, pp. 1–8.

[23] Y. Jiang, Y. Liu *et al.*, *Stochastic Network Calculus.* Springer, 2008, vol. 1.

[24] M. Fidler and A. Rizk, “A guide to the stochastic network calculus,” *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 92–105, 2015.

[25] J.-Y. Le Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet.* Springer Science & Business Media, 2001, vol. 2050.

[26] C.-S. Chang, *Performance Guarantees in Communication Networks*, ser. Telecommunication Networks and Computer Systems. Springer-Verlag. [Online]. Available: https://www.springer.com/gp/book/9781852332266

[27] A. Parekh and R. Gallager, “A generalized processor sharing approach to flow control in integrated services networks: the single-node case,” *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pp. 344–357, 1993.

[28] R. L. Cruz, “A calculus for network delay. I. Network elements in isolation,” *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 114–131, 1991.

[29] ——, “A calculus for network delay. II. Network analysis,” *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 132–141, 1991.

[30] ——, “Quality of service guarantees in virtual circuit switched networks,” *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, pp. 1048–1056, 1995.

[31] H. Sariowan, “A service-curve approach to performance guarantees in integrated-service networks,” Ph.D. dissertation, 1996, aAI9732701.

[32] P802.1DG – TSN Profile for Automotive In-Vehicle Ethernet Communications. [Online]. Available: https://1.ieee802.org/tsn/802-1dg/(Accessed:13/07/2022)

[33] IEC and IEEE, “IEC/IEEE 60802 - Time-Sensitive Networking Profile for Industrial Automation,” vol. IEC/IEEE 60802 (D1.3), Sep. 2021.

[34] IEEE, “Draft Standard for Local and metropolitan area networks — Time-Sensitive Networking Profile for Service Provider Networks,” *IEEE P802.1DF™/D0.1*, Dec. 2020.

[35] “IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks,” *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)*, pp. 1–1993, Jul. 2018, conference Name: IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014).

[36] R. Salazar, T. Godfrey, N. Finn, C. Powell, B. Rolfe, and M. Seewald, “Utility Applications of Time Sensitive Networking White Paper,” *Utility Applications of Time Sensitive Networking White Paper*, pp. 1–19, 2019.

[37] N. Finn, P. Thubert, B. Varga, and J. Farkas, “RFC8655: Deterministic Networking Architecture,” 2019. [Online]. Available: https://www.rfc-editor.org/info/rfc8655

[38] M. Andrews, “Instability of FIFO in session-oriented networks,” *Journal of Algorithms*, vol. 50, no. 2, pp. 232–245, 2004, Soda 2000 Special Issue. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0196677403000968

[39] ——, "Instability of FIFO in the permanent sessions model at arbitrarily small network loads," *ACM Trans. Algorithms*, vol. 5, no. 3, jul 2009. [Online]. Available: https://doi.org/10.1145/1541885.1541894

[40] C. Demichelis and P. Chimento, "RFC3393: IP Packet Delay Variation metric for IP performance metrics (ippm)," 2002. [Online]. Available: https://www.rfc-editor.org/info/rfc3393

[41] IEEE, "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks - Amendment 34:Asynchronous Traffic Shaping," *IEEE Std 802.1Qcr-2020 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018, IEEE Std 802.1Qcc-2018, IEEE Std 802.1Qcy-2019, and IEEE Std 802.1Qcx-2020)*, pp. 1–151, 2020.

[42] D. C. Verma, H. Zhang, and D. Ferrari, "Delay jitter control for real-time communication in a packet switching network," in *Proceedings of TRICOMM '91: IEEE Conference on Communications Software: Communications for Distributed Applications and Systems*, pp. 35–43.

[43] H. Zhang and D. Ferrari, "Rate-controlled static-priority queueing," in *IEEE INFOCOM '93 The Conference on Computer Communications, Proceedings*, Mar. 1993, pp. 227–236 vol.1.

[44] R. L. Cruz, "SCED+: efficient management of quality of service guarantees," in *Proceedings. IEEE INFOCOM '98, the Conference on Computer Communications. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Gateway to the 21st Century (Cat. No.98*, vol. 2, pp. 625–634 vol.2.

[45] L. Thomas and J.-Y. Le Boudec, "On time synchronization issues in time-sensitive networks with regulators and nonideal clocks," in *2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems*. New York, NY, USA: ACM, 2020, pp. 1–41. [Online]. Available: https://doi.org/10.1145/3393691.3394206

[46] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks–Amendment 29: Cyclic Queuing and Forwarding," *IEEE 802.1Qch-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd(TM)-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, IEEE Std 802.1Qbz-2016, and IEEE Std 802.1Qci-2017)*, pp. 1–30, 2017.

[47] "ISO/IEC/IEEE International Standard – Information technology – Telecommunications and information exchange between systems – Local and Metropolitan Area Networks – Specific requirements – Part 1Q: Bridges and bridged networks AMENDMENT 3: Enhancements for scheduled traffic," *ISO/IEC/IEEE 8802-1Q:2016/Amd.3:2017(E)*, pp. 1–62, 2018.

[48] J. Bennett, C. Partridge, and N. Shectman, “Packet reordering is not pathological network behavior,” *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 789–798, Dec. 1999.

[49] M. Laor and L. Gendel, “The effect of packet reordering in a backbone link on application throughput,” *IEEE Network*, vol. 16, no. 5, pp. 28–36, Sep. 2002.

[50] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, “Measurement and Classification of Out-of-Sequence Packets in a Tier-1 IP Backbone,” *IEEE/ACM Transactions on Networking*, vol. 15, no. 1, pp. 54–66, Feb. 2007.

[51] B. Varga, J. Farkas, S. Kehrer, and T. Heer, “Deterministic Networking (DetNet): Packet Ordering Function,” Nov. 2022, work in Progress. [Online]. Available: https://www.ietf.org/archive/id/draft-ietf-detnet-pof-01.html

[52] N. M. Piratla and A. P. Jayasumana, “Metrics for packet reordering—A comparative analysis,” *International Journal of Communication Systems*, vol. 21, no. 1, pp. 99–113, 2008. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.884

[53] Y. Gao and Y. Q. Zhao, “Large deviations for re-sequencing buffer size,” *IEEE Transactions on Information Theory*, vol. 58, no. 2, pp. 1003–1009, 2012.

[54] J. Li, Y. Zhou, L. Lamont, M. Huang, and Y. Q. Zhao, “Probabilistic analysis of resequencing queue length in multipath packet data networks,” in *2010 IEEE Global Telecommunications Conference (GLOBECOM 2010)*, 2010, pp. 1–5.

[55] International Electrotechnical Commission, “IEC 62439-3:2021 - Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR),” vol. IEC 62439-3:2021, 2021. [Online]. Available: https://webstore.iec.ch/publication/64423

[56] “IEEE Standard for Local and Metropolitan Area Networks–Frame Replication and Elimination for Reliability,” *IEEE Std 802.1CB-2017*, pp. 1–102, 2017.

[57] B. Varga, J. Farkas, and A. G. Malis, “Deterministic Networking (DetNet): DetNet PREOF via MPLS over UDP/IP,” Internet Engineering Task Force, Internet-Draft, Nov. 2022, work in Progress. [Online]. Available: https://www.ietf.org/archive/id/draft-ietf-detnet-mpls-over-ip-preof-02.html

[58] L. Thomas, A. Mifdaoui, and J.-Y. Le Boudec, “Worst-case delay bounds in time-sensitive networks with packet replication and elimination,” 2021. [Online]. Available: https://arxiv.org/abs/2110.05808

[59] J. Perser, A. Morton, L. Ciavattone, G. Ramachandran, and S. Shalunov, “RFC4737: Packet reordering metrics,” 2006. [Online]. Available: https://www.rfc-editor.org/info/rfc4737

[60] E. Mohammadpour, E. Stai, and J.-Y. Le Boudec, “Improved delay bound for a service curve element with known transmission rate,” *IEEE Networking Letters*, pp. 1–4, 2019.

[61] E. Mohammadpour, E. Stai, and J.-Y. L. Boudec, “Improved network calculus delay bounds in time-sensitive networks,” *arXiv preprint arXiv:2204.10906*, 2022.

[62] E. Mohammadpour, E. Stai, and J.-Y. Le Boudec, “Improved credit bounds for the credit-based shaper in time-sensitive networking,” *IEEE Networking Letters*, vol. 1, no. 3, pp. 136–139, 2019.

[63] E. Mohammadpour, E. Stai, M. Mohiuddin, and J.-Y. Le Boudec, “Latency and backlog bounds in time-sensitive networking with credit based shapers and asynchronous traffic shaping,” in *2018 30th International Teletraffic Congress (ITC 30)*, vol. 02, pp. 1–6.

[64] E. Mohammadpour and J.-Y. Le Boudec, “On Packet Reordering in Time-Sensitive Networks,” *IEEE/ACM Transactions on Networking*, pp. 1–13, 2021.

[65] E. Mohammadpour and J.-Y. L. Boudec, “Analysis of dampers in time-sensitive networks with non-ideal clocks,” *IEEE/ACM Transactions on Networking*, pp. 1–15, 2022.

[66] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury, “Ultra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research,” *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 88–145, 2019.

[67] A. Charny and J.-Y. Le Boudec, “Delay bounds in a network with aggregate scheduling,” in *Quality of Future Internet Services*. Springer, 2000, pp. 1–13.

[68] M. Carlson, W. Weiss, S. Blake, Z. Wang, D. Black, and E. Davies, “RFC2475: An Architecture for Differentiated Services,” 1998. [Online]. Available: https://www.rfc-editor.org/info/rfc2475

[69] D. D. Clark, S. Shenker, and L. Zhang, “Supporting real-time applications in an integrated services packet network: Architecture and mechanism,” in *Conference proceedings on Communications architectures & protocols*, 1992, pp. 14–26.

[70] M. Buchli, D. De Vleeschauwer, J. Janssen, and G. Petit, “Policing aggregates of voice traffic with the token bucket algorithm,” in *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333)*, vol. 4, 2002, pp. 2547–2551 vol.4.

[71] ITU-T, “I.371 : traffic control and congestion control in b-isdn.” [Online]. Available: https://www.itu.int/rec/T-REC-I.371-200403-I/

[72] H. Naser and A. Leon-Garcia, “Effects of internetworking on atm traffic descriptors,” in *1998 IEEE ATM Workshop Proceedings. ’Meeting the Challenges of Deploying the Global Broadband Network Infrastructure’ (Cat. No.98EX164)*, 1998, pp. 279–288.

[73] T. Tsuchiya and I. Saito, “The worst case cell arrival patterns that conform to the gcra in atm networks,” in *Proceedings of GLOBECOM ’95*, vol. 2, 1995, pp. 1432–1438 vol.2.

[74] S. Shenker and J. Wroclawski, "RFC2215: General characterization parameters for integrated service network elements," 1997. [Online]. Available: https://www.rfc-editor.org/info/rfc2215

[75] P802.1DP – TSN for Aerospace Onboard Ethernet Communications. [Online]. Available: https://1.ieee802.org/tsn/802-1dp/(Accessed:13/07/2022)

[76] M. Ashjaei, S. Mubeen, J. Lundbäck, M. Gålnander, K.-L. Lundbäck, and T. Nolte, "Modeling and timing analysis of vehicle functions distributed over switched ethernet," in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 8419–8424.

[77] J. Walrand, M. Turner, and R. Myers, "An architecture for in-vehicle networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 7, pp. 6335–6342, 2021.

[78] J. Imtiaz, J. Jasperneite, and L. Han, "A performance study of Ethernet audio video bridging (AVB) for industrial real-time communication," in *2009 IEEE Conference on Emerging Technologies Factory Automation*, pp. 1–8.

[79] J. Zhang, L. Chen, T. Wang, and X. Wang, "Analysis of tsn for industrial automation based on network calculus," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 240–247.

[80] B. Varga, J. Farkas, R. Cummings, Y. Jiang, and D. Fedyk, "RFC9016: Flow and Service Information Model for Deterministic Networking (DetNet)," 2021. [Online]. Available: https://www.rfc-editor.org/info/rfc9016

[81] "IEEE Standard for Local and Metropolitan Area Networks – Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic," *IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q-2014/Cor 1-2015)*, pp. 1–57, 2016.

[82] D. Thiele, R. Ernst, and J. Diemer, "Formal worst-case timing analysis of Ethernet TSN's time-aware and peristaltic shapers," in *2015 IEEE Vehicular Networking Conference (VNC)*, 2015, pp. 251–258.

[83] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. Elbakoury, "Performance Comparison of IEEE 802.1 TSN Time Aware Shaper (TAS) and Asynchronous Traffic Shaper (ATS)," *IEEE Access*, vol. 7, pp. 44 165–44 181, 2019.

[84] A. Arestova, K.-S. Jens Hielscher, and R. German, "Simulative evaluation of the TSN mechanisms time-aware shaper and frame preemption and their suitability for industrial use cases," in *2021 IFIP Networking Conference (IFIP Networking)*, 2021, pp. 1–6.

[85] L. Zhao, P. Pop, and S. S. Craciunas, "Worst-case latency analysis for IEEE 802.1Qbv time sensitive networks using network calculus," *IEEE Access*, vol. 6, pp. 41 803–41 815, 2018.

[86] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. J. Wellings, “Applying new scheduling theory to static priority pre-emptive scheduling,” *Software engineering journal*, vol. 8, no. 5, pp. 284–292, 1993.

[87] K. W. Tindell, A. Burns, and A. J. Wellings, “An extendible approach for analyzing fixed priority hard real-time tasks,” *Real-Time Syst.*, vol. 6, no. 2, p. 133–151, mar 1994. [Online]. Available: https://doi.org/10.1007/BF01088593

[88] B. Andersson, S. Baruah, and J. Jonsson, “Static-priority scheduling on multiprocessors,” in *Proceedings 22nd IEEE Real-Time Systems Symposium (RTSS 2001) (Cat. No.01PR1420)*, 2001, pp. 193–202.

[89] H. Bauer, J.-L. Scharbarg, and C. Fraboul, “Applying trajectory approach with static priority queuing for improving the use of available AFDX resources,” *Real-Time Syst.*, vol. 48, no. 1, p. 101–133, jan 2012. [Online]. Available: https://doi.org/10.1007/s11241-011-9142-9

[90] A. Demers, S. Keshav, and S. Shenker, “Analysis and simulation of a fair queueing algorithm,” *SIGCOMM Comput. Commun. Rev.*, vol. 19, no. 4, p. 1–12, aug 1989. [Online]. Available: https://doi.org/10.1145/75247.75248

[91] A. Parekh and R. Gallager, “A generalized processor sharing approach to flow control in integrated services networks: the single-node case,” *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pp. 344–357, 1993.

[92] ——, “A generalized processor sharing approach to flow control in integrated services networks: the multiple-node case,” *IEEE/ACM Transactions on Networking*, vol. 2, no. 2, pp. 137–150, 1994.

[93] J. Rexford, A. Greenberg, and F. Bonomi, “Hardware-efficient fair queueing architectures for high-speed networks,” in *Proceedings of IEEE INFOCOM ’96. Conference on Computer Communications*, vol. 2, 1996, pp. 638–646 vol.2.

[94] E. L. Hahne and R. Gallager, “Round robin scheduling for fair flow control in data communication networks,” in *Proc. IEEE Int. Conf. Commun.* IEEE, 1986, pp. 103–107.

[95] M. Katevenis, “Fast switching and fair control of congested flow in broadband networks,” *IEEE Journal on selected Areas in Communications*, vol. 5, no. 8, pp. 1315–1326, 1987.

[96] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, “Weighted round-robin cell multiplexing in a general-purpose atm switch chip,” *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, pp. 1265–1279, 1991.

[97] H. Chaskar and U. Madhow, “Fair scheduling with tunable latency: a round-robin approach,” *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 592–601, 2003.

[98] S. M. Tabatabaee, J.-Y. Le Boudec, and M. Boyer, “Interleaved weighted round-robin: A network calculus analysis,” *IEICE Transactions on Communications*, vol. 104, no. 12, pp. 1479–1493, 2021.

[99] M. Shreedhar and G. Varghese, “Efficient fair queueing using deficit round robin,” in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM ’95. New York, NY, USA: Association for Computing Machinery, 1995, p. 231–242. [Online]. Available: https://doi.org/10.1145/217382.217453

[100] L. Lenzini, E. Mingozzi, and G. Stea, “Tradeoffs between low complexity, low latency, and fairness with deficit round-robin schedulers,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 681–693, 2004.

[101] M. Boyer, G. Stea, and W. M. Sofack, “Deficit round robin with network calculus,” in *6th International ICST Conference on Performance Evaluation Methodologies and Tools*, 2012, pp. 138–147.

[102] L. Zhao, F. He, and J. Lu, “Comparison of AFDX and audio video bridging forwarding methods using network calculus approach,” in *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, 2017, pp. 1–7.

[103] L. Zhao, P. Pop, Z. Zheng, and Q. Li, “Timing analysis of AVB traffic in TSN networks using network calculus,” in *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2018, pp. 25–36.

[104] L. Zhao, P. Pop, Z. Zheng, H. Daigmorte, and M. Boyer, “Latency analysis of multiple classes of avb traffic in tsn with standard credit behavior using network calculus,” *IEEE Transactions on Industrial Electronics*, vol. 68, no. 10, pp. 10 291–10 302, 2021.

[105] J. Cao, M. Ashjaei, P. J. L. Cuijpers, R. J. Bril, and J. J. Lukkien, “An independent yet efficient analysis of bandwidth reservation for credit-based shaping,” pp. 1–10, 2018.

[106] L. Zhao, P. Pop, and S. Steinhorst, “Quantitative performance comparison of various traffic shapers in time-sensitive networking,” *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[107] F.-J. Gotz, “Traffic shaper for control data traffic (CDT),” in *IEEE 802 AVB Meeting*, 2012.

[108] F.-J. G. S. Kerschbaum and F. Chen, “Towards the calculation of performance guarantees for BLS in time-sensitive networks,” in *IEEE 802.1 TSN Meeting*, 2013.

[109] S. Thangamuthu, N. Concer, P. J. L. Cuijpers, and J. J. Lukkien, “Analysis of Ethernet-switch traffic shapers for in-vehicle networking applications,” in *2015 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2015, pp. 55–60.

[110] D. Thiele and R. Ernst, “Formal worst-case timing analysis of Ethernet TSN’s burst-limiting shaper,” in *2016 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2016, pp. 187–192.

[111] A. Finzi, A. Mifdaoui, F. Frances, and E. Lochin, “Network calculus-based timing analysis of afdx networks with strict priority and tsn/bls shapers,” in *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*, 2018, pp. 1–10.

[112] ——, “Incorporating tsn/bls in afdx for mixed-criticality applications: Model and timing analysis,” in *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*, 2018, pp. 1–10.

[113] ——, “Network calculus-based timing analysis of afdx networks incorporating multiple tsn/bls traffic classes,” *arXiv preprint arXiv:1905.00399*, 2019.

[114] A. Finzi and A. Mifdaoui, “Worst-case timing analysis of AFDX networks with multiple TSN/BLS shapers,” *IEEE Access*, vol. 8, pp. 106 765–106 784, 2020.

[115] H. Zhang and D. Ferrari, “Rate-controlled service disciplines,” *Journal of high speed networks*, vol. 3, no. 4, pp. 389–412, 1994.

[116] L. Georgiadis, R. Guerin, V. Peris, and K. Sivarajan, “Efficient network QoS provisioning based on per node traffic shaping,” *IEEE/ACM Transactions on Networking*, vol. 4, no. 4, pp. 482–501, 1996.

[117] J.-Y. Le Boudec, “Application of network calculus to guaranteed service networks,” *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 1087–1096, 1998.

[118] C.-S. Chang, “On deterministic traffic regulation and service guarantees: a systematic approach by filtering,” *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 1097–1110, 1998.

[119] E. Wandeler, A. Maxiaguine, and L. Thiele, “Performance analysis of greedy shapers in real-time systems,” in *Proceedings of the Design Automation Test in Europe Conference*, vol. 1, 2006, p. 6 pp.

[120] M. Rahmani, K. Tappayuthpijarn, B. Krebs, E. Steinbach, and R. Bogenberger, “Traffic shaping for resource-efficient in-vehicle communication,” *IEEE Transactions on Industrial Informatics*, vol. 5, no. 4, pp. 414–428, 2009.

[121] H. Ayed, A. Mifdaoui, and C. Fraboul, “Hierarchical traffic shaping and frame packing to reduce bandwidth utilization in the AFDX,” in *Proceedings of the 9th IEEE International Symposium on Industrial Embedded Systems (SIES 2014)*, 2014, pp. 77–86.

[122] J. Specht and S. Samii, “Urgency-based scheduler for time-sensitive switched Ethernet networks,” in *the 28th Euromicro Conference on Real-Time Systems (ECRTS)*, Jul. 2016, pp. 75–85.

[123] J.-Y. Le Boudec, "A theory of traffic regulators for deterministic networks with application to interleaved regulators," vol. 26, no. 6, pp. 2721–2733. [Online]. Available: https://doi.org/10.1109/TNET.2018.2875191

[124] J. Liebeherr, "Duality of the max-plus and min-plus network calculus," vol. 11, no. 3, pp. 139–282. [Online]. Available: https://www.nowpublishers.com/article/Details/NET-059

[125] K. Eriksson, D. Estep, C. Johnson, and J. Hoffman, *Applied mathematics: body and soul.* Springer, 2004, vol. 1.

[126] C.-S. Chang, *Performance Guarantees in Communication Networks.* New York: Springer-Verlag, 2000.

[127] M. Boyer, G. Stea, and W. M. Sofack, "Deficit Round Robin with network calculus," in *6th International ICST Conference on Performance Evaluation Methodologies and Tools*, 2012, pp. 138–147.

[128] W. Mangoua Sofack and M. Boyer, "Non preemptive static priority with network calculus: Enhancement," in *Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance.* Springer Berlin Heidelberg, 2012, pp. 258–272.

[129] A. Burchard and J. Liebeherr, "A general per-flow service curve for gps," in *2018 30th International Teletraffic Congress (ITC 30)*, vol. 02, 2018, pp. 31–36.

[130] J. B. Schmitt, F. A. Zdarsky, and I. Martinovic, "Improving performance bounds in feed-forward networks by paying multiplexing only once," in *14th GI/ITG Conference - Measurement, Modelling and Evalutation of Computer and Communication Systems*, pp. 1–15.

[131] J. B. Schmitt and F. A. Zdarsky, "The disco network calculator: A toolbox for worst case analysis," in *Proceedings of the 1st International Conference on Performance Evaluation Methodolgies and Tools*, ser. valuetools '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 8–es. [Online]. Available: https://doi.org/10.1145/1190095.1190105

[132] A. Mifdaoui and T. Leydier, "Beyond the accuracy-complexity tradeoffs of compositional analyses using network calculus for complex networks," in *10th International Workshop on Compositional Theory and Technology for Real-Time Embedded Systems (co-located with RTSS 2017)*, Paris, France, Dec. 2017, pp. pp. 1–8. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01690096

[133] L. Thomas, J.-Y. Le Boudec, and A. Mifdaoui, "On cyclic dependencies and regulators in time-sensitive networks," in *2019 IEEE Real-Time Systems Symposium (RTSS)*, 2019, pp. 299–311.

[134] S. Bondorf, P. Nikolaus, and J. B. Schmitt, "Quality and cost of deterministic network calculus: Design and evaluation of an accurate and fast analysis," *Proc.*

ACM Meas. Anal. Comput. Syst., vol. 1, no. 1, jun 2017. [Online]. Available: https://doi.org/10.1145/3084453

[135] F. Geyer and S. Bondorf, "Graph-based deep learning for fast and tight network calculus analyses," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 75–88, 2021.

[136] F. Geyer, A. Scheffler, and S. Bondorf, "Network calculus with flow prolongation – a feedforward FIFO analysis enabled by ML," 2022. [Online]. Available: https://arxiv.org/abs/2202.03004

[137] L. Bisti, L. Lenzini, E. Mingozzi, and G. Stea, "Numerical analysis of worst-case end-to-end delay bounds in FIFO tandem networks," *Real-Time Systems*, vol. 48, no. 5, pp. 527–569, 2012.

[138] A. Bouillard and G. Stea, "Exact worst-case delay in FIFO-multiplexing feed-forward networks," *IEEE/ACM Transactions on Networking*, vol. 23, no. 5, pp. 1387–1400, 2015.

[139] A. Bouillard, "Trade-off between accuracy and tractability of network calculus in FIFO networks," *Performance Evaluation*, vol. 153, p. 102250, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166531621000675

[140] N. Finn, J.-Y. Le Boudec, E. Mohammadpour, J. Zhang, and B. Varga, "RFC9320: Deterministic Networking (DetNet) Bounded Latency," 2022. [Online]. Available: https://www.rfc-editor.org/info/rfc9320

[141] H. Daigmorte, M. Boyer, and L. Zhao, "Modelling in network calculus a TSN architecture mixing Time-Triggered, Credit Based Shaper and Best-Effort queues," Jun. 2018, working paper or preprint. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01814211

[142] L. Maile, K.-S. Hielscher, and R. German, "Network calculus results for TSN: An introduction," in *2020 Information Communication Technologies Conference (ICTC)*. IEEE, 2020, pp. 131–140.

[143] Y. Jiang, "Some properties of length rate quotient shapers," *arXiv:2107.05021 [cs.PF]*, 2021. [Online]. Available: https://arxiv.org/abs/2107.05021

[144] ——, "A basic result on the superposition of arrival processes in deterministic networks," in *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6.

[145] L. Zhao, P. Pop, Z. Zheng, and Q. Li, "Timing analysis of AVB traffic in TSN networks using network calculus," in *Real-Time and Embedded Technology and App. Symp.*, ser. RTAS '18. IEEE, 2018, pp. 25–36.

[146] "RealTime-at-Work online Min-Plus interpreter for Network Calculus," https://www.realtimeatwork.com/minplus-playground, accessed: year-month-day.

[147] J. C. Bennett, K. Benson, A. Charny, W. F. Courtney, and J.-Y. Le Boudec, “Delay jitter bounds and packet scale rate guarantee for expedited forwarding,” *IEEE/ACM Transactions on Networking (TON)*, vol. 10, no. 4, pp. 529–540, 2002.

[148] A. Bouillard and G. Stea, “Exact worst-case delay for FIFO-multiplexing tandems,” in *6th International ICST Conference on Performance Evaluation Methodologies and Tools*, 2012, pp. 158–167.

[149] S. Thangamuthu, N. Concer, P. J. L. Cuijpers, and J. J. Lukkien, “Analysis of ethernet-switch traffic shapers for in-vehicle networking applications,” in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, ser. DATE ’15. EDA Consortium, pp. 55–60. [Online]. Available: http://dl.acm.org/citation.cfm?id=2755753.2755766

[150] “IEEE Standard for Local and Metropolitan Area Networks–Audio Video Bridging (AVB) Systems,” *IEEE Std 802.1BA-2021 (Revision of IEEE Std 802.1BA-2011)*, pp. 1–45, 2021.

[151] Time-sensitive networking (TSN) task group. [Online]. Available: https://1.ieee802.org/tsn/.

[152] H. Daigmorte, M. Boyer, and L. Zhao, “Modelling in network calculus a TSN architecture mixing time-triggered, credit based shaper and best-effort queues.” [Online]. Available: https://hal.archives-ouvertes.fr/hal-01814211

[153] J. Cao, P. J. Cuijpers, R. J. Bril, and J. J. Lukkien, “Independent yet tight WCRT analysis for individual priority classes in ethernet AVB,” in *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, ser. RTNS ’16. ACM, pp. 55–64, event-place: Brest, France. [Online]. Available: http://doi.acm.org/10.1145/2997465.2997493

[154] F. Kamoun, L. Kleinrock, and R. Muntz, “Queueing analysis of the ordering issue in a distributed database concurrency control mechanism.” in *Proceedings 2nd International Conference on Distributed Computing Systems.* IEEE, 1981, pp. 13–23.

[155] F. Baccelli, E. Gelenbe, and B. Plateau, “An end-to-end approach to the resequencing problem,” *Journal of the ACM (JACM)*, vol. 31, no. 3, pp. 474–485, 1984.

[156] V. Paxson, “End-to-end Internet packet dynamics,” in *Proceedings of the ACM SIGCOMM ’97 conference on Applications, technologies, architectures, and protocols for computer communication*, ser. SIGCOMM ’97. Cannes, France: Association for Computing Machinery, Oct. 1997, pp. 139–152. [Online]. Available: https://doi.org/10.1145/263105.263155

[157] J. Bellardo and S. Savage, “Measuring packet reordering,” in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment*, ser. IMW ’02. New York, NY, USA: Association for Computing Machinery, 2002, p. 97–105. [Online]. Available: https://doi.org/10.1145/637201.637216

[158] Y. Wang, G. Lu, and X. Li, "A study of Internet packet reordering," in *Information Networking. Networking Technologies for Broadband and Mobile Networks.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 350–359.

[159] N. M. Piratla, A. P. Jayasumana, and A. A. Bare, "Reorder Density (RD): A formal, comprehensive metric for packet reordering," in *Networking 2005. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems.* Springer Berlin Heidelberg, 2005, pp. 78–89.

[160] W. Saab, R. Rudnik, J.-Y. Le Boudec, L. Reyes-Chamorro, and M. Paolone, "Robust real-time control of power grids in the presence of communication network non-idealities," in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS).* IEEE, 2018, pp. 1–6.

[161] J.-Y. Le Boudec, "Some properties of variable length packet shapers," *IEEE/ACM Transactions on Networking*, vol. 10, no. 3, pp. 329–337, 2002.

[162] J.-Y. Le Boudec and A. Charny, "Packet scale rate guarantee for non-fifo nodes," in *Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 1. IEEE, 2002, pp. 84–93.

[163] Data center 40GE Switch study, Cisco Nexus 9508 DR 140126L. [Online]. Available: http://miercom.com/pdf/reports/20140126.pdf

[164] R. Obermaisser, *Time-Triggered Communication*, 1st ed. Boca Raton: CRC Press, 2012.

[165] A. Charny and J.-Y. Le Boudec, "Delay bounds in a network with aggregate scheduling," in *Quality of Future Internet Services*, ser. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp. 1–13. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-39939-9_1

[166] A. Grigorjew, F. Metzger, T. Hoßfeld, J. Specht, F.-J. Götz, F. Chen, and J. Schmitt, "Asynchronous traffic shaping with jitter control," Institut für Informatik, Tech. Rep., 2020.

[167] R. Shoushou, L. Bingyang, M. Rui, W. Chuang, J.-Y. Le Boudec, E. Mohammadpour, and A. El Fawal, "A method for sending data packets and network equipment," China Patent 114 095 454.

[168] R. Shoushou, L. Bingyang, J.-Y. Le Boudec, E. Mohammadpour, and A. El Fawal, "A method and device for data transmission," China Patent 202 011 539 631.

[169] M. Mathis and J. Mahdavi, "Deprecating the TCP macroscopic model," *ACM SIGCOMM Computer Communication Review*, vol. 49, no. 5, pp. 63–68, 2019.

[170] "G.810 : Definitions and terminology for synchronization networks." [Online]. Available: https://www.itu.int/rec/T-REC-G.810-199608-I/en

[171] D. Mills, E. J. Martin, J. Burbank, and W. Kasch, “RFC5905: Network Time Protocol Version 4: Protocol and Algorithms Specification,” 2010. [Online]. Available: https://www.rfc-editor.org/info/rfc5905

[172] “IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems,” *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pp. 1–269, Jul. 2008, iEEE Std 1588-2008 (Revision of IEEE Std 1588-2002).

[173] P. Moreira, J. Serrano, T. Wlostowski, P. Loschmidt, and G. Gaderer, “White rabbit: Sub-nanosecond timing distribution over Ethernet,” in *Control and Communication 2009 International Symposium on Precision Clock Synchronization for Measurement*, Oct. 2009, pp. 1–5, iSSN: 1949-0313.

[174] E. Powers and J. Hahn, “GPS and Galileo UTC time distribution,” pp. 484–488, Jan. 2004, publisher: IET Digital Library. [Online]. Available: https://digital-library.theiet.org/content/conferences/10.1049/cp_20040914

[175] “IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications,” *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pp. 1–421, 2020.

[176] R. Brown, “Calendar queues: a fast o(1) priority queue implementation for the simulation event set problem,” *Communications of the ACM*, vol. 31, no. 10, pp. 1220–1227, Oct. 1988. [Online]. Available: https://doi.org/10.1145/63039.63045

[177] A. Saeed, N. Dukkipati, V. Valancius, V. The Lam, C. Contavalli, and A. Vahdat, “Carousel: Scalable traffic shaping at end hosts,” in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM ’17. New York, NY, USA: ACM, 2017, pp. 404–417, event-place: Los Angeles, CA, USA. [Online]. Available: http://doi.acm.org/10.1145/3098822.3098852

[178] Triple-speed Ethernet MegaCore function user guide. [Online]. Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/archives/ug-ethernet-16.0.pdf

[179] Low latency Ethernet 10G MAC user guide. [Online]. Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/archives/ug-32b-10g-ethernet-mac-15.1.pdf

[180] “IEEE standard for local and metropolitan area networks–audio video bridging (AVB) systems,” *IEEE Std 802.1BA-2011.* [Online]. Available: https://ieeexplore.ieee.org/document/6032690

[181] D. Pannell. Audio video bridging Gen 2 assumptions. [Online]. Available: https://www.ieee802.org/1/files/public/docs2013/avb-pannell-gen2-assumptions-1113-v17.pdf

[182] M. Boyer and P. Roux, “A common framework embedding network calculus and event stream theory,” *HAL Preprint*, May 2016. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01311502

[183] ——, “Embedding network calculus and event stream theory in a common model,” in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. Berlin, Germany: IEEE Press, Sep. 2016, pp. 1–8. [Online]. Available: https://doi.org/10.1109/ETFA.2016.7733565

List of Publications

Following are lists of all my publications written as a PhD student at EPFL.

Papers

1. **E.Mohammadpour**, E. Stai, M.Mohiuddin, and J.-Y. Le Boudec, “Latency and backlog bounds in time-sensitive networking with credit based shapers and asynchronous traffic shaping,” in 2018 30th International Teletraffic Congress (ITC 30), vol. 02, pp. 1–6.
2. **E.Mohammadpour**, E. Stai, and J.-Y. Le Boudec, “Improved delay bound for a service curve element with known transmission rate,” IEEE Networking Letters, pp. 1–4, 2019.
3. **E.Mohammadpour**, E. Stai, and J.-Y. Le Boudec, “Improved credit bounds for the credit-based shaper in time-sensitive networking,” IEEE Networking Letters, vol. 1, no. 3, pp. 136–139, 2019.
4. **E. Mohammadpour** and J.-Y. Le Boudec, “On Packet Reordering in Time-Sensitive Networks,” IEEE/ACM Transactions on Networking, pp. 1–13, 2021.
5. **E.Mohammadpour** and J.-Y. L. Boudec, “Analysis of dampers in time-sensitive networks with non-ideal clocks,” IEEE/ACM Transactions on Networking, pp. 1–15, 2022.
6. **E. Mohammadpour**, E. Stai, and J.-Y. L. Boudec, “Improved network calculus delay bounds in time-sensitive networks,” submitted to IEEE/ACM Transactions on Networking [available online in arXiv preprint arXiv:2204.10906], 2022.

IETF Documents

1. N. Finn, J.-Y. Le Boudec, **E. Mohammadpour**, J. Zhang, and B. Varga, “Deterministic Networking (DetNet) Bounded Latency,” RFC 9320, 2022.

Patents

1. R. Shoushou, L. Bingyang, M. Rui, W. Chuang, J.-Y. Le Boudec, **E. Mohammadpour**, and A. El Fawal, “A method for sending data packets and network equipment,” China Nat.

Intellectual Property Admin., Beijing, China, Tech. Rep. CN114095454A, Jul. 2020.

2. R. Shoushou, L. Bingyang, J.-Y. Le Boudec, **E. Mohammadpour**, and A. El Fawal, "A method and device for data transmission," China Nat. Intellectual Property Admin., Beijing, China, Tech. Rep. 202011539631.2, Dec. 2020.

Ehsan MOHAMMADPOUR

Route de la Maladiére 8
Chavannes-près-Renens 1022, Switzerland
📱 +41 77 939 47 53
☎ +41 21 693 12 66
✉ ehsan.mohammadpour@epfl.ch
linkedin.com/in/ehsanmpd
Skype:ehsanmpd@gmail.com

- Time-sensitive communications, including IEEE TSN and IETF DetNet
- Network calculus and worst-case delay analysis
- Scheduling, shaping, bounded delay and low jitter mechanisms

Education

2018–now **Swiss Federal Institute of Technology (EPFL)**, *Switzerland*, PhD in Computer Science.

2014–2016 **Tehran Polytechnic**, *Iran*, M.Sc. in Information Technology (computer networks).
- Excellence admission award for PhD studies from Tehran Polytechnic.
- Ranked 2^{nd}; received top student award with GPA 19.40/20.

2010–2014 **Tehran Polytechnic**, *Iran*, B.Sc. in Computer Engineering.
- Excellence admission award for Master studies from Tehran Polytechnic.
- Ranked 2^{nd}; received top student award with GPA 18.05/20.

Professional Experience

Current
2018–Now **Laboratory for Computer Communications and Applications (EPFL)**, *Research Assistant*.
- The research was done in close collaboration with Huawei Technologies Co. Ltd. within the framework of the Deterministic Networks project.
- Worst-case analysis of time-sensitive networks in terms of delay, delay-jitter and buffer bounds.
- Worst-case analysis of packet misordering and the impact of re-sequencing buffers.
- Delivered the analysis in terms of IETF Informational Internet Draft, accepted as an RFC.
- Designing an architecture of dampers, as a mechanism to provide bounded delay and low delay-jitter, in large-scale time-sensitive networks (China patents).
- Formal calculation of delay and delay-jitter bounds in a network of dampers under general network settings with non-ideal clocks.
- Developed a TSN computation tool in Python (based on the FT-TFA tool) to evaluate dampers and re-sequencing buffers in a scalable framework.

2017–2018 **Laboratory for Computer Communications and Applications (EPFL)**, *Intern*.
- Proposed a heuristic algorithm for routing of critical flows using linear programming in a time-sensitive network with DRR schedulers.

2014–2016 **Network Optimization Laboratory (Tehran Polytechnic)**, *Research Assistant*.
- Proposed a near-optimal heuristic energy-aware algorithm for routing of flows in green networks.
- Modeling the energy-aware routing as a constrained MILP optimization problem.
- Developed a simulator in C#, integrated with CPLEX solver, to test and evaluate the proposed energy-aware routing algorithm.

2015 **Tehran Polytechnic Research Center**, *C# Software Developer*.
- Developed a graphical C# application to monitor the communication of devices through the CAN bus.
- The project was in collaboration with several teams of computer vision, electrical and mechanical engineers.

Expertise

Technical

Programming	Python, C++, Past: {C#, JAVA, MIPS, VHDL, Verilog}
Tools	Wireshark, LaTex, SVN ,GIT
OS	MAC OS, Linux, Windows
ML	scikit-learn (Online course with INRIA)

Soft Skills

Project management, Team-working , Stress and time managements, Structural and critical thinking

Concepts

Networking	TCP/IP, Congestion Control, ARP, OSPF, BGP, VPN, TSN, DetNet
Security	Symmetric/Asymmetric encryption, Side-channel attacks, SQL injection
ML	Regression, Classification, Over/Underfitting, Regularization

Related Teaching Experience

2018-2020 **TCP/IP Networking**, *Teaching Assistant*, EPFL.
Responsible for labs with hands-on exercises on socket programming, TCP congestion control, IPv4/IPv6 interworking, tunneling, routing, and network security.

2014-2016 **Computer Architecture and Logic Circuits Laboratory**, *Labratory Instructor*, Tehran Polytechnic.
Teaching of VHDL as a hardware description language. The lab contains design and implementation of logic circuits, design and implemention of various elements of basic computer, including RAM, ALU, and Control Unit using Xilinx Spartan3 FPGA board.

Extra-Curricular Activities

2018-now	Computer Science PhD Representative, *EPFL*.
2018-now	Collaboration with IETF Deterministic Networking Working Group (DetNet WG), *IETF*.
2019	PhD Observer at EPFLinnovators Admission Committee (3rd call), *EPFL*.

Languages

English (fluent), French (elementary), German (basic), Farsi (Native)

Honors and Awards

- Excellence scholarship award by Iranian National Elites Foundation.
- 3rd place in FPT'15 international FPGA design competition in Queenstown, New Zealand, 2015.
- 1st place in national Digital System Design competitions in hardware security, Iran, 2015.
- 2nd place in national Digital System Design competitions in hardware-software co-design, Iran, 2015.

Selected Publications

2022 N. Finn, J.-Y. Le Boudec, **E. Mohammadpour**, J. Zhang, and B. Varga, "RFC9320: Deterministic Networking (DetNet) Bounded Latency", 2022 [Link].

2020 R. Shoushou, L. Bingyang, M. Rui, W. Chuang, J.-Y. Le Boudec, **E. Mohammadpour**, A. El Fawal, "A Method for Sending Data Packets and Network Equipment," *China Patent: 202011539631.2*.

2020 R. Shoushou, L. Bingyang, M. Rui, W. Chuang, J.-Y. Le Boudec, **E. Mohammadpour**, A. El Fawal, "A Method and Device for Data Transmission," *China Patent: 202010760188.5*.

2018 **E. Mohammadpour**, E. Stai, M. Mohiuddin, and J.-Y. Le. Boudec, "Latency and Backlog Bounds in Time-Sensitive Networking with Credit Based Shapers and Asynchronous Traffic Shaping," *30th* 212 *International Teletraffic Congress (ITC30)*, Vienna, Austria, doi: 10.1109/ITC30.2018.10053.

More in Google Scholar