

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *Institut National Polytechnique de Toulouse (INP Toulouse)*
Discipline ou spécialité : *Réseaux, Télécoms, Systèmes et Architectures*

Présentée et soutenue par *Xiaoting Li*
Le *jeudi 19 septembre 2013*

Titre : *Worst-case delay analysis of real-time switched Ethernet networks with flow local synchronization*

JURY
Christian FRABOUL, INP Toulouse - IRIT
Laurent GEORGE, Université Paris-Est - LIGM
Nicolas NAVET, Université du Luxembourg - LASSY
Frédéric RIDOUARD, Université de Poitiers- LIAS
Jean-Luc SCHARBARG, INP Toulouse - IRIT
Ye-Qiong SONG, Université de Lorraine - LORIA
Yvon TRINQUET, Université de Nantes - IRCCyN

Ecole doctorale : *Mathématiques Informatique Télécommunications (MITT)*
Unité de recherche : *IRIT - Institut de Recherche en Informatique de Toulouse*
Directeur(s) de Thèse : *Christian FRABOUL - Jean-Luc SCHARBARG*
Rapporteurs : *Laurent GEORGE - Ye-Qiong SONG*

Acknowledgement / Remerciements

La présente étude n'aurait pas été possible sans le bienveillant soutien de certaines personnes. Et je ne suis pas non plus capable de dire dans les mots qui conviennent, le rôle qu'elles ont pu jouer à mes côtés pour en arriver là.

En premier lieu, je tiens à remercier mes directeurs de thèse, Christian Fraboul et Jean-Luc Scharbarg, professeurs à l'INP-ENSEEIHT et chercheurs à l'Institut de Recherche en Informatique de Toulouse (IRIT, UMR 5505 du CNRS), pour m'avoir guidée, encouragée, conseillée, fait beaucoup voyager pendant quatre ans tout en me laissant une grande liberté. Au travers de nos discussions, ils m'ont apporté une compréhension plus approfondie des divers aspects du sujet.

Yvon Trinquet m'a fait l'honneur de présider le jury, qu'il soit remercié pour son intérêt pour mon travail. J'adresse mes sincères remerciements à Laurent George et Ye-Qiong Song qui ont accepté de rapporter cette thèse, ainsi qu'à Nicolas Navet et Frédéric Ridouard pour l'avoir examinée.

Durant ces quatre ans, j'ai eu la chance de bénéficier de l'aide de nombreux chercheurs. Parmi les contributeurs à ce travail de thèse, je remercie chaleureusement Henri Bauer, chercheur à l'Université de Poitiers, pour ses travaux antérieurs et son aide désintéressée. Je remercie également son collègue Frédéric Ridouard, pour les échanges fructueux que nous avons entretenus. J'en profite aussi pour saluer Katia Jaffres-Runser, chercheur à l'INP de Toulouse, pour son aide pendant la rédaction de ma thèse.

Ensuite je tiens à remercier tous les personnels de l'équipe IRT à l'IRIT, André-Luc Beylot, Emmanuel Chaput, Jérôme Ermont, Julien Fasson, Katia Jaffres-Runser, Gentian Jakllari, Béatrice Paillassa, Maaike Verloop, $\cdots$, sans qui cette thèse ne serait pas ce qu'elle est. Merci également à Sylvie Armengaud et Sylvie Eichen pour leur professionnalisme sans faille, leurs encouragements et leur assistance. Je passe ensuite une dédicace spéciale à tous les jeunes que j'ai eu le plaisir de côtoyer durant ces années à Toulouse, à savoir Abdelaziz, Adnan, Bouchra, Emilie, Fabian, Florian, Michaël, Nesrine, Raoul, Razvan, Tony, Victor, $\cdots$. Les moments agreable que nous avons passés ensemble au bureau, dans le couloir, à la cantine, et au foyer, je les garde précieusement au fond de mon coeur.

Ma reconnaissance va à ceux qui ont plus particulièrement assuré le soutien affectif de ce travail doctoral : ma famille en Chine ainsi que Christine Fraboul. Enfin, j'adresse mille mercis à mon époux Chao, mon amour, pour tout ce qu'il me donne et tout ce qu'il m'apporte de bon depuis ces années, et à mon fils Noah, le trésor le plus précieux de ma vie!

Je vous aime profondément!

Xiaoting

Abstract

Full-duplex switched Ethernet is a promising candidate for interconnecting real-time industrial applications. But due to IEEE 802.1d indeterminism, the worst-case delay analysis of critical flows supported by such a network is still an open problem. Several methods have been proposed for upper-bounding communication delays on a real-time switched Ethernet network, assuming that the incoming traffic can be upper bounded. The main problem remaining is to assess the tightness, i.e. the pessimism, of the method calculating this upper bound on the communication delay. These methods consider that all flows transmitted over the network are independent. This is true for flows emitted by different source nodes since, in general, there is no global clock synchronizing them. But the flows emitted by the same source node are local synchronized. Such an assumption helps to build a more precise flow model that eliminates some impossible communication scenarios which lead to pessimistic delay upper bounds.

The core of this thesis is to study how local periodic flows synchronized with offsets can be handled when computing delay upper-bounds on a real-time switched Ethernet. In a first step, the impact of these offsets on the delay upper-bound computation is illustrated. Then, the integration of offsets in the Network Calculus and the Trajectory approaches is introduced. Therefore, a modified Network Calculus approach and a modified Trajectory approach are developed whose performances are compared on an Avionics Full-DupleX switched Ethernet (AFDX) industrial configuration with one thousand of flows. It has been shown that, in the context of this AFDX configuration, the Trajectory approach leads to slightly tighter end-to-end delay upper bounds than the ones of the Network Calculus approach. But offsets of local flows have to be chosen. Different offset assignment algorithms are then investigated on the AFDX industrial configuration. A near-optimal assignment can be exhibited.

Next, a pessimism analysis of the computed upper-bounds is proposed. This analysis is based on the Trajectory approach (made optimistic) which computes an under-estimation of the worst-case delay. The difference between the upper-bound (computed by a given method) and the under-estimation of the worst-case delay gives an upper-bound of the pessimism of the method. This analysis gives interesting comparison results on the Network Calculus and the Trajectory approaches pessimism.

The last part of the thesis, deals with a real-time heterogeneous network architecture where CAN buses are interconnected through a switched Ethernet backbone using dedicated bridges. Two approaches, the component-based approach and the Trajectory approach, are developed to conduct a worst-case delay analysis for such a network. Clearly, the ability to compute end-to-end delays upper-bounds in the context of heterogeneous network architecture is promising for industrial domains.

List of Abbreviations and Symbols

Abbreviations:

AFDX	Avionics Full-DupleX Ethernet
ARINC	Aeronautical Radio, Incorporated
BAG	Bandwidth Allocation Gap
CAN	Controller Area Network
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CrossedS	an offset assignment considering the number of crossed switches based on PairAssign (see PairAssign)
CrossedSSA	an offset assignment considering the number of crossed switches based on SingleAssign (see SingleAssign)
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
ECU	Electronic Control Unit
EDF	Earliest Deadline First
EPL	Ethernet Powerlink
ES	End System
ETE	End-To-End
ETH	Ethernet network
EtherCAT	Ethernet for Control Automation Technology
FCFS	First Come First Served
FIFO	First In First Out
FIP	Factory Instrumentation Protocol
FP	Fixed Priority
FP/EDF	Fixed Priority/Earliest Deadline First
FP/FIFO	Fixed Priority/First In First Out
gcd	greatest common divisor
GCD	dissimilar offset assignment
GCDMinus	a near-optimal offset assignment heuristic considering the minus value of the greatest common divisor of periods of a pair of flows
lcm	least common multiple
IdealAssign	an ideal offset assignment
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol

MostLoad	an offset assignment considering the load at output port based on PairAssign (see PairAssign)
MostLoadSA	an offset assignment considering the load at output port based on SingleAssign (see SingleAssign)
Opt	an optimal approach based on the modified Network Calculus approach and the modified Trajectory approach
PairAssign	offset assignments of near-optimal offset assignment heuristics and GCD (see GCD)
PC	Purely CAN flows in a heterogeneous network
PE	Purely Ethernet flows in a heterogeneous network
PF	Periodic Flow
PROFIBUS	Process Field Bus
PROFINET	Process Field Network
QoS	Quality of Service
RAGCD	a near-optimal offset assignment heuristic considering the rate sum and the greatest common divisor of periods of a pair of flows
RateAdd	a near-optimal offset assignment heuristic considering the rate sum of a pair of flows
RC	Remote CAN flows in a heterogeneous network
RMGCD	a near-optimal offset assignment heuristic considering the maximum rate and the greatest common divisor of periods of a pair of flows
RT	Real Time
SAC	Safe Arrival Curve
SF	Sporadic Flow
SingleAssign	an offset assignment oringinally designed for automotive
TCnet	Time-Critical Control Network
TCP	Transmission Control Protocol
TCP/UDP/IP	Transmission Control Protocol/User Datagram Protocol/Internet Protocol
TDMA	Time Division Multiple Access
TTE	Time-Triggered Ethernet
UDP	User Datagram Protocol
UTP	Unshielded Twisted Pair
VL	Virtual Link
VLAN	Virtual Local Area Network

Symbols:

α_i	the arrival curve of flow τ_i
α^*	the arrival curve of output flow
α^h	the arrival curve of the aggregated flow at the output port h
$\alpha_{\mathrm{IP}_k^h}$	the arrival curve of the aggregated flow of input link IP_k^h
$\alpha^h_{\mathcal{G}_x}$	the *Safe Arrival Curve* of a flow set $\mathcal{G}_x$ at the output port h
$a^h_{f_i}$	the arrival time of a frame f_i at the output port h
$A_{i,j}$	the workload interval for flow τ_j delaying flow τ_i
$A_{i,j,k}$	the constrained workload interval of τ_k to delay τ_i
β	the service curve
bp^h	a busy period at the output port h without idle time during it
$B_{i,t}$	the sum from Term 3.3 to Term 3.6
BAG_i	the *Bandwidth Allocation Gap* value of VL v_i
$\mathcal{B}_i^{slow}$	the computation range for the Trajectory approach
$Brdg_i$	a brigde connecting a CAN bus and the Ethernet backbone in a heterogeneous network
γ_{r_i,b_i}	the arrival curve of a leaky bucket flow model, where r_i is the leak rate and b_i is the burst
$comp_x$	a component in a heterogeneous network which has its own bandwidth and manages its flows based on a local scheduling
C_{ETH}	the maximum transmission time of an Ethernet frame encapsulating CAN frames in a heterogeneous network
C_i	the frame transmission time of flow τ_i
C_i^h	the frame transmission time of τ_i on node h
$C_i^{comp_x}$	the frame transmission time of flow τ_i at the component $comp_x$ in a heterogeneous network
CAN_i	A CAN bus
$dest_i$	the destination node of flow τ_i
D_{hp_i}	the delay of flow τ_i generated by interference flows in hp_i
$D_{max_i}^{comp_x}$	the maximum delay of flow τ_i generated in component $comp_x$ of a heterogeneous network
$D_{min_i}^{comp_x}$	the minimum delay of flow τ_i generated in component $comp_x$ of a heterogeneous network
D_{sp_i}	the delay of flow τ_i generated by interference flows in sp_i
δ_i	the delay of flow τ_i generated by flows in δ_i due to FIFO with non-preemptive effect
Δ_i^h	a factor considering the frame serialization with reachable frame sequences in the pessimism analysis
$\Delta_{i,t}^h$	the frame serialization factor
f_i	one frame of flow τ_i
$f_{i,j}$	the jth frame of flow τ_i
$f(h)$	the first frame of the busy period bp^h

$first_i$	the source node of flow τ_i
$first_{j,i}$	the first output port where a competing flow τ_j crosses the flow τ_i
$first_{j,i}^{FP}$	the first shared FP output port of flow τ_i and flow τ_j along the path $\mathcal{P}_i$ in a heterogeneous network
$\mathcal{G}_x$	the xth subset containing dependent flows
h	a node of a network
$h(\alpha, \beta)$	the maximum horizontal deviation between α and β
hp_i	a set of flows having higher priorities than flow τ_i
IP_k^h	the set of flows crossing the kth input link of h
j_i	the release jitter of flow τ_i
J_i	the maximum release jitter of flow τ_i
$J_i^{comp_x}$	the maximum release jitter of flow τ_i at the component $comp_x$ in a heterogeneous network
l_k^h	the duration of sequence seq_k^h without its first frame
l_{max}	the maximum frame length of a Ethernet frame
l_{min}	the minimum frame length of a Ethernet frame
$last_{i,j}^{FIFO}$	the last share FIFO output port of flow τ_i and flow τ_j along the path $\mathcal{P}_i$ in a heterogeneous network
$last_i$	the last visited output port of flow τ_i
lp_i	a set of flows having lower priorities than flow τ_i
L_{max}	the maximum processing delay when a frame crosses a link
L_{min}	the minimum processing delay when a frame crosses a link
LD_i	the transmission delay on links of flow τ_i along the path $\mathcal{P}_i$
M_i^h	the earliest possible starting instant of bp^h for the flow τ_i
$MD_{i,j}^h$	the minimum duration from τ_i to τ_j at the output port h
n_B	the number of bridges visited by a studied flow in a heterogeneous network
$n_{B_{out}}$	the number of bridg ourpur ports connceting Ethernet visited by flow τ_i in a heterogeneous network
n_S	the number of switches visited by a studied flow in a heterogeneous network
N_i	an end node of a real-time switched Ethernet
N_B	the maximum number of CAN frames encapsulated within an Ethernet frame at a bridge in a heterogeneous network
O_i	the offset of flow τ_i
OP^h	the output port link of h
p_i	the priority of flow τ_i
$p_i^{comp_x}$	the priority of flow τ_i at the component $comp_x$ in a heterogeneous network
$p(h)$	the last frame of the busy period bp^h
$P_{CF_{i,t}^{last_i}}$	the pessimism upper bound of the delay of flow τ_i generated by competing flows in the pessimism analysis
$P_{CT_i^{last_i}}$	the pessimism upper bound of the delay of flow τ_i generated by busy period transition in the pessimism analysis
$P_{S_{i,t}^{last_i}}$	the upper bounded pessimism introduced by the computation of frame serialization for flow τ_i in the pessimism analysis
$\mathcal{P}_i$	the path of flow τ_i
$\lvert\mathcal{P}_i\rvert$	the length of path $\mathcal{P}_i$

rl_k^h	a reachable duration of a frame sequence seq_k^h without its first frame in the pessimism analysis
R	the transmission rate of the network
R_c	the total constant cost in switches and in bridges a studied flow in a heterogeneous network
R_{CAN}	the bandwidth of a CAN bus in a heterogeneous network
R_{ETH}	the bandwidth of the Ethernet backbone in a heterogeneous network
R_i	the end-to-end delay of flow τ_i
R_{trs}	the tranmission cost of a studied flow along its path in a heterogeneous network
$R(t)$	input function of flow traffic over time t
$R^*(t)$	output function of flow traffic over time t
$RS_{i,t,x}$	the maximum workload generated by the interference set $\mathcal{G}_x$
seq_k^h	the frame sequence of IP_k^h
shp_i	$j \in shp_i$ if $p_j > p_i$ and τ_j shares pure FP or both FP and FIFO scheduling policies with τ_i in a heterogeneous network
slp_i	$j \in slp_i$ if $p_j < p_i$ and $first_{j,i}$ is with FP scheduling in a heterogeneous network
ssp_i	$j \in ssp_i$ if $p_j = p_i$; or if $p_j < p_i$ and $first_{j,i}$ is with FIFO scheduling in a heterogeneous network
sl	the switching latency delay of a switch
$slow_i$	the slowest node visited by τ_i on its path $\mathcal{P}_i$
$slow_{j,i}$	the slowest node visited by τ_j on the path $\mathcal{P}_i$
sp_i	a set of flows having the same priority as flow τ_i
$S_{max_i}^h$	the maximum delay experienced by a frame of flow τ_i from its source node $first_i$ to output port h
$S_{min_i}^h$	the minimum delay experienced by a frame of flow τ_i from its source node $first_i$ to output port h
S_{ETH}	the maximum size of an Ethernet frame encapsulating CAN frames in a heterogeneous network
SD_i	the total switching lantency delay of flow τ_i along the path $\mathcal{P}_i$
τ_i	a real-time switched Ethernet flow
$\tau_i^{comp_x}$	the input flow of flow τ_i at the component $comp_x$ in a heterogeneous network
T_B	a constant cost used in a brigde for frame encapsulation/decapsulation in a heterogeneous network
T_i	the period of flow τ_i
$T_i^{comp_x}$	the period of flow τ_i at the component $comp_x$ in a heterogeneous network
UP_i	the pessimism upper bounds of the worst-case ETE delay computation of the flow τ_i using the Trajectory approach
UR_i	the underestimation of the worst-case ETE delay of flow τ_i using the classical Trajectory approach

$UR_{CF_i^{last_i}}$	the lower bounded delay of flow τ_i generated by competing flows in the pessimism analysis
$UR_{CT_i^{last_i}}$	the lower bounded delay of flow τ_i generated by busy period transition in the pessimism analysis
$UR_{S_i^{last_i}}$	a reachable value of frame serialization effect in the pessimism analysis
v_i	a VL in the AFDX network
$W_{i,t}^{last_i}$	the latest starting time of a frame f_i of τ_i at its last visited output port $last_i$
WD	the maximum waiting delay of a CAN frame pending in a bridge output port in a heterogeneous network
WD_i	the total waiting delay in the output buffers of flow τ_i along the path $\mathcal{P}_i$

Mathematics Symbols:

$(a)^+ = \max(a, 0)$

$[\![1, n]\!]$ an integer set $\{1, 2, ..., n\}$

Contents

Acknowledgement / Remerciements **1**

Abstract **3**

List of Abbreviations and Symbols **5**

General Introduction **1**

1 Worst-case delay analysis of switched Ethernet networks **7**
1.1 Introduction . . . 7
1.2 Context of real-time Ethernet . . . 8
1.2.1 Evolution . . . 8
1.2.2 Real-time Ethernet solutions . . . 9
1.3 Real-time switched Ethernet network . . . 10
1.3.1 Network model . . . 11
1.3.2 Flow model . . . 12
1.3.3 A real-time switched Ethernet example: AFDX network . . . 13
1.4 End-to-end delay analysis on a real-time switched Ethernet . . . 13
1.4.1 End-to-end delay . . . 14
1.4.2 Minimum end-to-end delay . . . 16
1.4.3 Maximum end-to-end delay . . . 16
1.5 Existing approaches for worst-case delay analysis . . . 19
1.5.1 Simulation . . . 20
1.5.2 Model-checking . . . 20
1.5.3 Network Calculus . . . 20
1.5.4 Trajectory approach . . . 21
1.6 Temporal constraints of dependent flows . . . 22
1.6.1 Minimum duration . . . 22
1.6.2 Computation of minimum duration . . . 24
1.7 Conclusion . . . 29

2 Modified Network Calculus approach integrating minimum duration constraints **31**
2.1 Introduction . . . 31
2.2 Classical Network Calculus approach . . . 32
2.2.1 Arrival curves . . . 32

2.2.2 Service curves . . . 33
2.2.3 Delay computation . . . 34
2.2.4 Frame serialization . . . 35
2.2.5 Limitation of the approach . . . 36
2.3 A modified approach considering minimum duration constraints . . . 36
2.3.1 General idea . . . 36
2.3.2 Computation overview . . . 37
2.3.3 Computation of the arrival curve of dependent flows . . . 38
2.4 Application on a small network example . . . 41
2.4.1 Classical Network Calculus approach . . . 42
2.4.2 Modified Network Calculus approach . . . 44
2.5 Conclusion . . . 48

3 Modified Trajectory approach integrating minimum duration constraints 51
3.1 Introduction . . . 51
3.2 Classical Trajectory approach . . . 52
3.3 A modified approach considering minimum duration constraints . . . 58
3.3.1 General idea . . . 58
3.3.2 Computation overview . . . 59
3.3.3 Computation of the maximum workload of dependent flows . . . 59
3.4 Illustration on a small network example . . . 65
3.4.1 Classical Trajectory approach . . . 65
3.4.2 Modified Trajectory approach . . . 66
3.5 Conclusion . . . 70

4 A real-time switched Ethernet example: the AFDX network 71
4.1 Introduction . . . 71
4.2 AFDX context . . . 72
4.3 Example of an offset assignment algorithm . . . 74
4.4 Worst-case delay analysis based on the modified Network Calculus approach . . . 75
4.5 Worst-case delay analysis based on the modified Trajectory approach . . . 78
4.6 A comparison of the two modified approaches . . . 81
4.7 Near-optimal offset assignment for the industrial AFDX network . . . 82
4.7.1 Existing offset assignments . . . 83
4.7.2 Optimal scenario of dependent flows over the AFDX network . . . 84
4.7.3 Offset assignment algorithms in the context of AFDX network . . . 86
4.7.4 Results . . . 92
4.8 Conclusion . . . 94

5 Pessimism analysis 97
5.1 Introduction . . . 97
5.2 Pessimism analysis based on the classical Trajectory approach . . . 99
5.2.1 Review of the classical Trajectory approach . . . 99
5.2.2 Pessimism analysis of computing flows . . . 101
5.2.3 Pessimism analysis of busy period transition . . . 105
5.2.4 Pessimism analysis of serialization effect . . . 108

5.2.5 Analytical method for underestimated delays . . . 115
5.2.6 Analytical method for maximum potential pessimism . . . 115
5.3 Integration of the minimum duration constraints . . . 116
5.3.1 Review of the modified Trajectory approach . . . 117
5.3.2 Illustration on the overestimation of dependent flows . . . 117
5.3.3 Pessimism analysis of the workload of dependent flows . . . 120
5.4 Application on the AFDX network . . . 121
5.4.1 Upper bounding the pessimism of the Network Calculus approach . . . 122
5.4.2 Upper bounding the pessimism of the Trajectory approach . . . 122
5.5 Conclusion . . . 124

6 Worst-case delay analysis on a heterogeneous network 127
6.1 Introduction . . . 127
6.2 Heterogeneous network architecture . . . 129
6.2.1 CAN bus . . . 129
6.2.2 Switched Ethernet backbone . . . 129
6.2.3 Heterogeneous flows . . . 130
6.2.4 Bridging strategy . . . 131
6.2.5 End-to-end delay analysis . . . 132
6.3 Component-based approach for worst-case delay analysis . . . 133
6.3.1 Component-based architecture . . . 133
6.3.2 Network Calculus approach for Non-preemptive FP Scheduling . . . 138
6.4 Trajectory approach for worst-case delay analysis . . . 139
6.4.1 Classical Trajectory approach for FP/FIFO scheduling applied to a homogeneous network . . . 139
6.4.2 A modified Trajectory approach adapted to a heterogeneous network . . . 143
6.5 Case study . . . 147
6.6 Improved Trajectory approach . . . 150
6.7 Conclusion . . . 153

Conclusions and Perspectives 155

A Network Calculus 161
A.1 Network Calculus . . . 161
A.2 Application on a switched Ethernet network . . . 163
A.3 Integration of frame serialization . . . 165

B Trajectory approach 169
B.1 Trajectory approach . . . 169
B.2 Application on a switched Ethernet network . . . 173
B.3 Integration of frame serialization . . . 176

Bibliographie 187

List of Figures

1.1 Example of the network architecture . . . 11
1.2 An illustration of a static flow path $\mathcal{P}_{i,j}$ . . . 12
1.3 Example of mapping flows on the network architecture . . . 12
1.4 Temporal characteristics of a sporadic flow and a periodic flow . . . 13
1.5 Illustration of end-to-end delay of flow τ_1 . . . 15
1.6 Illustration of minimum end-to-end delay of flow τ_1 . . . 17
1.7 Illustration of end-to-end delay of flow τ_1 on a possible scenario . . . 18
1.8 The end-to-end delay characteristics . . . 19
1.9 A reference network example . . . 23
1.10 Minimum durations between τ_1 and τ_2 at the output port of N_1 . . . 23
1.11 Minimum duration between τ_1 and τ_2 at the output port of N_1 with release jitters 24
1.12 Propagation of the *Minimum Duration* . . . 25
1.13 Temporal durations from between a frame arrival of τ_1 and a frame arrival of τ_2 25
1.14 General case . . . 27
1.15 One possible scenario . . . 28
2.1 Recall of the network example for the Network Calculus approach . . . 32
2.2 The arrival curves of flows τ_1 and τ_2 at N_1 . . . 33
2.3 The maximum delay generated at the output port of N_1 . . . 34
2.4 Propagation of the arrival curve . . . 35
2.5 The arrival curves of flows . . . 35
2.6 Serialization of $\alpha_1^{S_1}$ and $\alpha_2^{S_1}$ . . . 36
2.7 Serialization of $\alpha_3^{S_1}$ and $\alpha_4^{S_1}$ . . . 36
2.8 Illustration of the burst workload at N_1 . . . 37
2.9 The arrival curves of α_i^h and α_j^h . . . 39
2.10 Sum arrival curves of $\mathcal{G}_x$ . . . 39
2.11 The *Safe Arrival Curve* of subset $\mathcal{G}_x$ . . . 41
2.12 The arrival curves of flows . . . 42
2.13 Maximum delay generated at S_1 . . . 43
2.14 Worst-case delay of τ_1 with the classical Network Calculus approach . . . 43
2.15 Sum arrival curves of $\mathcal{G}_1$ . . . 44
2.16 The *Safe Arrival Curve* of subset $\mathcal{G}_1$ . . . 45
2.17 The overall arrival curve α^{S_1} . . . 46
2.18 Comparison with the classical Network Calculus approach . . . 46
2.19 Illustration on the output port of S_2 . . . 47
2.20 The illustration on R_1 computed by the modified Network Calculus approach . 48
3.1 Recall of the network example for the Trajectory approach . . . 52

3.2 Worst-case scenario corresponding to the classical Trajectory approach 53
3.3 Illustration of the interval at $first_{j,i}$ 54
3.4 Illustration of the interval at $first_j$ 54
3.5 Illustration of the workload intervals of τ_3 56
3.6 Illustration of the computation of R_1 57
3.7 Illustration of the workload intervals of τ_3 and τ_4 58
3.8 An illustrative example 60
3.9 Illustration of the workload intervals $A_{i,j}$ and $A_{i,k}$ 60
3.10 Illustration of the constrained workload intervals $A_{i,j,k}$ 61
3.11 Illustration of the constrained workload intervals $A_{i,k,j}$ 61
3.12 Illustration of a possible scenario of workload interval 62
3.13 Illustration of the frame shift 62
3.14 Classical approach 65
3.15 Modified approach 65
3.16 Illustration on maximum workload curve $RS_{1,t,2}$ 67
3.17 Illustration on maximum workload curve $RS_{1,t,1}$ 68
3.18 Illustration on maximum workload curve $RS_{1,t,3}$ 69
3.19 Worst-case scenario corresponding to the modified Trajectory approach 69
4.1 An illustrative industrial AFDX network architecture 73
4.2 Improvement of the modified Network Calculus approach 76
4.3 Improvement of the modified Network Calculus approach with partial dependencies 77
4.4 Improvement of dependent flows of the modified Network Calculus 78
4.5 Improvement of independent flows of the modified Network Calculus 78
4.6 Improvement of the modified Trajectory approach 79
4.7 Improvement of the modified Trajectory approach with partial dependent flows 80
4.8 Improvement of dependent flows based on the Trajectory approach 81
4.9 Improvement of independent flows based on the Trajectory approach 81
4.10 Comparative results of two approaches 82
4.11 A sample AFDX network of Example 1 86
4.12 Scenarios of the VL v_1 86
4.13 A small example of AFDX network of Example 2 87
4.14 Illustration of the *SingleAssign* with low workload 88
4.15 Illustration of *SingleAssign* with increased workload 89
4.16 Illustration of *MostLoadSA* with high workload 90
4.17 A small example of AFDX 93
4.18 Comparison of GCD and SingleAssign 93
4.19 Comparative results of *IdealAssign*, *SingleAssign* and *MostLoadSA* 94
5.1 Example 3 101
5.2 Worst-case scenario for τ_1 of the Example 3 102
5.3 Illustration on minimum inter-arrival time between two frames of the same flow 102
5.4 Example 4 103
5.5 Worst-case scenario for τ_1 of the Example 4 104
5.6 Sample network of Example 5, 6, 7 105
5.7 Worst-case scenario for τ_1 of the Example 5 106
5.8 Worst-case scenario for τ_1 of the Example 6 107
5.9 Illustration of $\Delta_{1,t}^{S_1}$ 109

5.10 Worst-case scenario for τ_1 of the Example 7 . . . 110
5.11 Illustration of **Lemma 2** . . . 112
5.12 Illustration of seq_0^h . . . 113
5.13 Illustration of a reachable seq_0^h: Case 1 . . . 113
5.14 Illustration of a reachable seq_0^h: Case 2 . . . 114
5.15 Illustration of a reachable delay of τ_1 . . . 117
5.16 Example 8 . . . 118
5.17 Illustration on minimum duration $MD_{5,6}^{N_5}$ . . . 118
5.18 Worst case scenario for τ_1 in Example 8 . . . 119
5.19 Example 9 . . . 119
5.20 Worst case scenario for τ_1 in Example 9 . . . 120
5.21 Upper bounded pessimism of the Network Calculus approach . . . 123
5.22 Upper bounded pessimism of the Trajectory approach . . . 124
6.1 Heterogeneous network . . . 130
6.2 Heterogeneous path for τ_2 . . . 132
6.3 Hierarchical scheduling framework . . . 134
6.4 Compositional scheduling framework on an example . . . 135
6.5 Component-based approach for heterogeneous networks . . . 135
6.6 Component and interface . . . 136
6.7 Illustration of the component-based approach for flow τ_2 . . . 137
6.8 Interval of flow with higher priority . . . 141
6.9 Illustration of heterogeneous scheduling policies integration . . . 145
6.10 Heterogeneous network example . . . 147
6.11 Worst-case ETE delay of pure Ethernet flows . . . 148
6.12 Worst-case ETE delay of pure CAN flows . . . 149
6.13 Worst-case ETE delay of remote CAN flows . . . 149
6.14 Worst-case ETE delay of remote CAN flows according to the path . . . 150
6.15 Example network . . . 151
6.16 Illustration on τ_1 . . . 152
6.17 Worst-case ETE delay of remote CAN flows with the improved Trajectory approach 153
6.18 Worst-case ETE delay of remote CAN flows of CAN_1 to CAN_3 and CAN_1 to CAN_4 with the improved Trajectory approach . . . 154
A.1 Illustration of Network Calculus . . . 162
A.2 Propagation of Network Calculus . . . 163
A.3 Illustration of Network Calculus applied to real-time switched Ethernet . . . 164
A.4 Illustration on frame serialization . . . 166
A.5 The aggregated flow of one input link IP_k^h considering the frame serialization . . . 167
B.1 Illustration of busy period . . . 170
B.2 Illustration of the workload interval $A_{i,j}$ . . . 172
B.3 An switched Ethernet network illustration . . . 174
B.4 The transmission process of frame f_1 of flow τ_1 . . . 174
B.5 Illustration on a node h with an output link OP^h and $k_h + 1$ input link IP_k^h . . . 177
B.6 Illustration on term $\Delta_{i,t}^h$ . . . 178

List of Tables

1.1 Flow parameters of the reference network example 23
1.2 Flow parameters of the reference network example with release jitters 24
1.3 All the possible values of $MD_{1,2}^{N_1}(k,l)$ 26
1.4 All the possible values of $MD_{2,1}^{N_1}(k,l)$ 27
1.5 minimum durations of flows τ_1, τ_2, τ_3 and τ_2 29
2.1 Recall of flow parameters of the network example for the Network Calculus approach 32
2.2 End-to-end delay upper bounds of the reference example 48
3.1 Recall of flow parameters of the network example for the Trajectory approach . 52
3.2 Minimum durations of flows τ_1, τ_2, τ_3 and τ_4 at their source nodes 65
3.3 End-to-end delay upper bounds of the reference example 70
4.1 BAGs and frame lengths 74
4.2 VL paths lengths 74
4.3 The example of offset assignment 75
4.4 Improvement on burst workload at output ports of ESs and switches 79
4.5 The Configuration of flows of Example 1 86
4.6 The Configuration of flows of Example 2 87
4.7 The assigned offsets based on different offset assignments 91
4.8 Statistic reduction on end-to-end delay upper bounds for each offset algorithm 92
5.1 Number and percentage of paths with exact worst-case delays 124
6.1 Flows in the example in Figure 6.1 131

General Introduction

Due to a limited bandwidth, the fieldbus technologies cannot cope with the increasing demand of data exchange of the industrial applications. Ethernet technology therefore becomes a promising candidate since it provides large bandwidth of data transmission and steadily decreasing cost. Since vintage Ethernet is not able to reach the real-time requirements, one solution considered in this thesis is based on a full-duplex switched Ethernet with traffic assumptions on incoming flows (traffic shaping).

One important real-time requirement is the worst-case end-to-end (ETE) communication delay. For a full-duplex switched Ethernet network, the end-to-end communication delay of a flow transmitted in the network is the time elapsed between the generation time of one frame of the flow at its source node and the reception time of this frame at its destination node. Then the end-to-end delay is the sum of the transmission delay on each crossed link and the delay generated at each crossed switch along the path. The worst-case end-to-end delay considers the most unfavorable scenario along the path. It depends on the generation times of other flows crossing the path, and therefore it is difficult to compute the worst-case end-to-end delays of large-scale networks. One solution is to guarantee the upper bounds of the worst-case delays by worst-case delay analysis, and efforts have been dedicated to solve this problem.

There exist several end-to-end delay analysis methods for real-time switched Ethernet networks. Simulation approach is based on the model of the network and it provides valuable informations about the delay distribution. However, worst-case delay is often a rare event that is missed by the simulation approach. Model-Checking performs an exhaustive exploration of the possible scenarios in order to calculate an exact worst-case end-to-end delay. However, it is limited to small networks due to the huge combinatorial explosion problem for large configuration. Network Calculus approach is a powerful method that uses arrival curves, which upper bound the arriving traffic, to describe arriving flows. It computes the maximum delay of a flow generated at each crossed switch and therefore leads to the worst-case end-to-end delay upper bound along the path. This approach is classical although it accumulates pessimism at each hop. The Trajectory approach allows a direct analysis on the worst-case scenario that can happen to a frame along its trajectory. It has been shown that this approach improves the computation of end-to-end delay upper bounds for an industrial avionics switched Ethernet network [BSF10]. Therefore, the Trajectory approach attracts increasing attention in the worst-case delay analysis.

All the worst-case delay analysis methods mentioned above consider that flows transmitted over the network are independent. This is true with flows emitted by different source nodes since there is no global clock to synchronize flows. However, each source node does have a

local clock which can synchronize its flows. It means that local flows (emitted by the same source node) can be synchronized and that periodic flows can have known offsets. Thus offset assumptions increase the knowledge about the frame arrivals of periodic flows emitted by the same source node. They help to build a more precise flow model and eliminate some impossible scenarios which lead to pessimistic computation. Hence, the study of offset constraints is an important issue which needs to be addressed in the worst-case delay analysis of switched Ethernet networks.

The subject of this thesis is to integrate the offset constraints in the worst-case delay analysis of switched-Ethernet networks. This problem is of great interest since improved worst-case delay analysis can be obtained on large scale industrial switched Ethernet network. In the first step, we identify the offset constraints among a set of periodic flows emitted by the same source node and illustrate the offset constraint and its impacts on worst-case delay computation. In the second step, two approaches, the Network Calculus approach and the Trajectory approach, are proposed to integrate these offset constraints. Therefore, these two approaches allow us to compute the worst-case delay upper bounds of a real-time switched Ethernet network with offset assumption. As a real example of real-time switched Ethernet network, the Avionics Full-DupleX switched Ethernet network (AFDX) is introduced. The two proposed approaches are applied on an industrial AFDX configuration to show the improvement of the computed end-to-end delay upper bounds. Since the proposed approaches provide us a tool to evaluate a network with offsets, we are interested in the offset assignment algorithm that leads to minimum computed end-to-end delay upper bounds for the industrial AFDX network. An optimal scenario is introduced which minimizes the worst-case delays for all the flows. The existing offset assignments are presented and evaluated by comparing the results to those obtained based on the optimal scenario.

The worst-case delay analysis leads to pessimistic delay upper bound computation. In order to investigate how pessimistic the proposed approaches are, an analytical pessimism analysis is developed based on the Trajectory approach and an upper bound of the introduced pessimism is obtained by measuring the maximum gap of a computed worst-case delay upper bound and an underestimated value of the worst-case delay.

As a future direction, a heterogeneous network architecture which combines existing field buses and a switched Ethernet is then studied. A worst-case delay analysis is tricky in such an architecture due to its heterogeneity nature (different bandwidths, scheduling policies, etc.). Two approaches are proposed to solve this problem. One is component-based approach which preserves the network properties but introduces pessimism at each component level; the other one is an adjusted Trajectory approach which unifies the heterogeneities along each path.

Main contributions of this thesis are summarized below.

- **Integration of offset constraints in the Network Calculus approach and in the Trajectory approach applied to the real-time switched Ethernet network.**

  Each source node in the switched Ethernet network has a local clock based on which the transmissions of periodic flows are not independent. Indeed, periodic flows with known offsets emitted by the same source node are dependent. A benefit of knowing the offsets is to improve the worst-case delay analysis for a switched Ethernet network

since it can eliminate some impossible scenarios. Existing approaches of worst-case delay analysis do not consider the offsets. In this work, we show how to integrate the offset constraints in two approaches: the Network Calculus approach [LSF10b, LSF10a, LSF10c] and the Trajectory approach. These two modified approaches allow us to improve the computation of worst-case delay upper bounds of large scale switched Ethernet networks.

- **Applications of the modified Network Calculus approach and the modified Trajectory approach on an industrial avionics switched Ethernet network.**

  The Avionics Full-DupleX switched Ethernet network (AFDX) can be seen as an example of real-time switched Ethernet. The two proposed approaches are applied to an industrial AFDX configuration. The results show that the two modified approaches improve the computation of delay upper bounds. We compare the delay upper bounds computed by these two modified approaches. Such a comparison allows us to know which approach is more suitable for such an industrial configuration. Moreover, an optimal approach is presented by choosing the tighter (smaller) computed worst-case delay between the results of the two modified approaches. Such an optimal approach provides an improved network performance evaluation by combining the two modified approaches.

- **Evaluation of offset assignment algorithms for the industrial AFDX network.**

  In the context of processor and CAN bus, there exist studies about offset assignments which lead to the maximum system performance [Goo03, GHN08]. Since the choice of offset assignment is related to a specific application, in this work, we focus on the industrial AFDX network [LSFF11]. An optimal scenario is introduced which considers infinitive durantion between frame arrivals for all the flows. The comparison between the offset assignment algorithms and the optimal scenario allows us to know which algorithm leads to better performance (smaller computed delay upper bounds) in this context as well as how far the existing algorithms are from the optimal case.

- **Analysis of pessimism introduced by the Trajectory approach and formalization of computation on pessimism upper bound.**

  The Trajectory approach guarantees the worst-case delay upper bounds of the switched Ethernet network by introducing pessimistic computation. The introduced pessimism directly reflects the reliability of the obtained worst-case delay upper bounds which are overestimated. A pessimism analysis has been proposed in [BSF10] which is empirical based on an unfavorable scenario. It relies on simulation and needs to build an unfavorable scenario for each network configuration. In this work, an analytical pessimism analysis based on the Trajectory approach is studied [LSF11]. We analyze the factors leading to pessimistic computation in the Trajectory approach which allows us to know where the pessimism comes from. Then we propose an analytical method to calculate an underestimated value of the worst-case delay based on the Trajectory approach. The difference between an underestimated delay and an overestimated delay allows us to upper bound the introduced pessimism.

- **Worst-case delay analysis of a real-time heterogeneous network.**

  With the increasing exchange of information in the real-time application, field buses cannot satisfy the industrial communication demand any more. The heterogeneous network

using a backbone network to connect existing networks is considered as a solution to build large scale industrial network. Then, the worst-case delay analysis to guarantee deterministic communication on such a network is crucial. However, due to the heterogeneities existing in such a network, the problem of worst-case delay analysis is complex. In this work, we propose two approaches to solve this problem [LSF12a, LSF12b]. The component-based approach divides the network into several components based on its properties and keeps network properties at each component level. This approach is executed at each component level and allows us to build a large scale network with flexibility and scalability. The Trajectory approach integrates heterogeneities by unifying parameters along each path. It allows us to consider for each frame the worst-case scenario along its heterogeneous path and avoid being holistic.

The organization of this thesis is described as follows.

Chapter 1 introduces the basis of the real-time switched Ethernet network context and the worst-case delay analysis. A literature review of existing approaches of worst-case delay analysis is also presented. The offset constraint is illustrated on a network example. It has been shown that for two periodic flows with known offsets emitted by the same source node, there is a minimum duration between their frame arrivals. Thus the scenario where their frames arrive at the same time can be impossible. Then the consequences on the worst-case delay analysis are emphasized.

Chapter 2 first introduces the classical Network Calculus approach applied to the switched Ethernet network. Then a modified Network Calculus is proposed with the integration of constraints of minimum durations. The computation is conducted at each output port along the path. At each output port, dependent flows are first classified into subsets. Any two flows in one subset have minimum duration constraints. Then the integration is implemented in the arrival curve of each subset. The modified approach is illustrated on a small network example.

Chapter 3 first presents the application of the classical Trajectory approach to the switched Ethernet network. We propose a modified Trajectory approach which integrates the constraints of minimum durations. This approach first classifies the dependent flows into subsets. Any two flows in one subset have minimum duration constraints. Then for each subset, its maximum delay is computed with the integration of minimum duration constraints. The modified approach is also illustrated on the same small network example used in the second chapter.

Chapter 4 introduces the Avionics Full-DupleX switched Ethernet network (AFDX) as a real-time switched Ethernet example. The evaluations on an industrial AFDX configuration for the modified Network Calculus approach and the modified Trajectory approach are conducted. The comparison between the two approaches in terms of worst-case delay upper bound is shown. This chapter also evaluates different offset assignment algorithms for the industrial AFDX configuration based on a proposed optimal scenario. The obtained results show how far the different offset assignment algorithms are from the optimal case.

Chapter 5 analyzes the factors of the Trajectory approach which introduce pessimism in the computation. Based on each factor, an analytical method to calculate an underestimated value of the worst-case delay is developed. The difference between an underestimated value of the worst-case delay and a worst-case delay upper bound gives an analytical measurement of

maximum introduced pessimism. The pessimism analysis is further developed accounting for the offset constraints and applied on the industrial AFDX network.

Chapter 6 gives a look at a future research direction. A heterogeneous network architecture where a switched Ethernet backbone interconnects existing CAN buses through bridges is presented. In the context of such architectures, two approaches are proposed for the worst-case delay analysis. The component-based approach is applied in this context by defining the interface of each network component. The Trajectory approach is adjusted and modified for the heterogeneous network by unifying heterogeneities along each path. The two proposed approaches are compared based on a middle scale network example.

Personal Publication List

Journal publications

[LSFF11] X. Li, J.-L. Scharbarg, F.Ridouard, and C. Fraboul, "Existing offset assignments are near optimal for an industrial AFDX network," *SIGBED Rev., Special Issue on the RTN 2011 Workshop, in conjunction with the ECRTS 2011*, vol. 8, no. 4, pp. 49–54, Dec. 2011. [Online]. Available: http://doi.acm.org/10.1145/2095256.2095263

Conference publications

[LSF10a] X. Li, J.-L. Scharbarg, and C. Fraboul, "Improving end-to-end delay upper bounds on an AFDX network by integrating offsets in worst-case analysis," in *Proc. IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*. IEEE, Sep. 2010, pp. 1–8.

[LSF10b] ——, "Improving worst-case end-to-end delay analysis with partial synchronization of periodic flows on a switched Ethernet network," in *EUROMICRO Conference on Real-Time Systems (ECRTS) (session WiP)*, Brussels, Belgium, July 2010, pp. 45–48.

[LSF10c] ——, "Partially synchronizing periodic flows with offsets improves worst-case end-to-end delay analysis of switched Ethernet," in *Int. symposium on Leveraging Applications of Formal Methods, Verification, and Validation (ISoLA)*, Heraklion, Greece, Nov. 2010, pp. 228–242.

[LSF11] ——, "Analysis of the pessimism of the Trajectory approach for upper bounding end-to-end delay of sporadic flows sharing a switched Ethernet network," in *Int. Conf. on Real-Time and Network Systems (RTNS)*, Nantes, France, Sep. 2011.

[LSF12a] ——, "Applying Trajectory approach for worst-case delay analysis of a heterogeneous real-time network," in *Proc. 18th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS) (session WiP)*, Beijing, China, Apr. 2012.

[LSF12b] ——, "Worst-case delay analysis on a real-time heterogeneous network," in *Proc. 7th IEEE Int. Symposium on Industrial Embedded Systems (SIES)*, Karlsruhe, Germany, June 2012.

Chapter 1

Worst-case delay analysis of switched Ethernet networks

Contents

1.1 Introduction . . . 7
1.2 Context of real-time Ethernet . . . 8
1.2.1 Evolution . . . 8
1.2.2 Real-time Ethernet solutions . . . 9
1.3 Real-time switched Ethernet network . . . 10
1.3.1 Network model . . . 11
1.3.2 Flow model . . . 12
1.3.3 A real-time switched Ethernet example: AFDX network . . . 13
1.4 End-to-end delay analysis on a real-time switched Ethernet . . . 13
1.4.1 End-to-end delay . . . 14
1.4.2 Minimum end-to-end delay . . . 16
1.4.3 Maximum end-to-end delay . . . 16
1.5 Existing approaches for worst-case delay analysis . . . 19
1.5.1 Simulation . . . 20
1.5.2 Model-checking . . . 20
1.5.3 Network Calculus . . . 20
1.5.4 Trajectory approach . . . 21
1.6 Temporal constraints of dependent flows . . . 22
1.6.1 Minimum duration . . . 22
1.6.2 Computation of minimum duration . . . 24
1.7 Conclusion . . . 29

1.1 Introduction

During the last three decades, fieldbus technologies have been successively developed and widely used in industrial control system. For instance, technologies such as PROFIBUS [TV99] and WorldFIP [Wor] have been popular in the context of automation while CAN [CAN] is a *de facto* standard for automotive embedded systems and the ARINC 429 [ACC04] has been developed for avionics systems. These fieldbuses interconnect sensors, actuators and controllers in wide range of applications and support the deterministic industrial communications by ensuring

bounded end-to-end (ETE) communication delays of messages. The guarantees on the end-to-end delays are achieved either by the nature of the fieldbus(e.g. ARINC 429 is a mono-emitter bus) or by a worst-case analysis such as the one which has been developed for CAN [TB94, THW94, TBW95b, NS98, HNNP02, DBBL07].

However, fieldbus technologies offer a limited bandwidth (up to 1 $Mbit/s$) which cannot cope with the increasing demand of data exchange of the industrial applications. In the context of avionics, the mono-emitter feature of the ARINC 429 leads to a huge number of buses which is unacceptable in terms of weight and wiring complexity. Ethernet is a very popular communication technology in the context of non real-time applications. It provides large bandwidth for data transmission (from 10 $Mbit/s$ to 10 $Gbit/s$). It is a cost-effective solution due to its steadily decreasing cost brought by mass production. However, vintage Ethernet is not suitable for real-time applications. Indeed, it is based on CSMA/CD which is not deterministic. Thus, it is impossible to bound the time needed for a successful transmission of a given frame.

Many solutions have been proposed in order to make Ethernet real-time. In this thesis, such a solution is considered . It is based on a full-duplex switched Ethernet with traffic assumptions on the incoming flows. Such assumptions have been made in the context of avionics networks [ACC08]. As we will see, the end-to-end delay of a frame transmitted on this network highly depends on the generation time of the other frames in the network. Fortunately, it has been shown that this end-to-end delay can be upper bounded [SKS02, Son01, FJJ09, CEL05, LH04a, LH04b, BSF09, BSF10, BSF12].

The aim of this chapter is to summarize the evolution of Ethernet toward more or less real-time solutions , to present the network solution which is considered in this thesis, to present the existing approaches for the end-to-end delay analysis of such networks and to justify the contribution of the thesis.

In this Chapter, Section 1.2 first presents the Ethernet evolution and its real-time solutions. Section 1.3 defines network and traffic model of a real-time switched Ethernet network. Section 1.4 defines the end-to-end delay of a real-time flow and illustrates its best and worst cases. Section 1.5 introduces existing approaches for the worst-case delay analysis. Section 1.6 illustrates minimum duration constraints existing between dependent flows and derives its computation. Section 1.7 concludes this chapter.

1.2 Context of real-time Ethernet

1.2.1 Evolution

The first "Ethernet" was proposed by Metcalfe and his Xerox PARC colleagues in 1972. It is based on the use of a shared media and an access algorithm called Carrier Sense Multiple Access with Collision Detection (CSMA/CD). Coax cables were used in baseband mode, thus allowing only unicast transmissions. Each coax cable constitutes one collision domain, where only one station may send at the same time, and one broadcast domain, where any station receives the current frame sent. Coax cables can be connected by repeaters in order to extend the Ethernet segment. Early Ethernet topology is shared bus, such as 10Base2 and 10Base5.

The disadvantages of the early bus topology are: 1) if there is a break or a fault happening, it impacts several nodes in the network, even the whole network; 2) Any adding/removing node disrupts network; 3) It is inconvenient to locate the break of the coax cables.

Therefore, a new Ethernet topology was proposed which is called a star topology. Nodes are interconnected by a hub device (basically multi-port repeater) using cheap unshielded twisted-pair (UTP) cable, like 10BaseT. This is also known as "CSMA/CD in a box". One advantage of such a topology is that it allows the reuse of the structured cabling already installed in the building. Another advantage is that single cable break/fault effects only one node. However, such a topology still leads to one single collision and broadcast domain, which is a logical bus.

For performance reasons, bridging was created to communicate at the data link layer while isolating the physical layer. Bridges are store and forward devices which introduce significant delay. It can filter traffic based on the addresses associated with each port and it forwards network traffic only to the necessary segments to avoid unnecessary flooding of frames to certain segments. It also checks frames which means that only well-formed Ethernet frames are forwarded from one Ethernet segment to another; therefore collisions and frame errors are isolated. Thus, bridges segment the network into several collision domains. However, broadcast traffic is still forwarded to all network segments (one single broadcast domain).

The switch has been introduced in order to satisfy increasing data exchanging demand. A switched Ethernet is based on star topology. The full-duplex communication on twisted pair cables allows a collision-free Ethernet (no collision domain). Such a technique allows simultaneous transmissions between different nodes. A switch supports different transmission rates on different ports, special forwarding techniques (cut through or store and forward) etc.. Thus switched Ethernet is a collision-free plug and play scalable Ethernet. Moreover, Virtual LANs allows to split the network into several broadcast domains.

Full duplex switched Ethernet networks have received increasing attention in the industrial domain since it can offer a higher bandwidth for data transmission and a steadily decreasing cost of components. In the following paragraphs, we will introduce the characteristics of switched Ethernet designed for industrial real-time applications.

1.2.2 Real-time Ethernet solutions

Nowadays, Ethernet is the fastest growing segment of industrial networking due to its increased bandwidth plus its decreased product costs which can satisfy the timing requirement of real-time applications. Demand for Ethernet as a real-time control network is therefore increased. Since standard Ethernet is not able to reach the requirements of the real-time Ethernet, different solutions to modify the Ethernet have been proposed. Some of the material in this chapter are taken from [Fel05, Dec05], to which the reader is referred for a more detailed overview of this subject. There are in principle three different approaches [Fel05] for a real-time Ethernet solution:

- *Top of TCP/IP* approach builds real-time modification over unchanged TCP/UDP/IP protocols. Such an approach simply uses a real-time protocol over TCP/UDP/IP protocols without any special modification. Existing propositions include Modbus/TCP [Aut],

EtherNet/IP [Sch01], P-NET on IP [Eth04b] etc.. Such a solution provides only soft real-time communications due to the adoption of TCP/UDP/IP protocols.

- *Top of Ethernet* approach bypasses the TCP/UDP/IP protocols and accesses directly to the Ethernet functionality. It realizes real-time protocol directly over the Ethernet without altering its hardware. This approach enhances real-time guarantees by using master-slave system and/or time slicing mechanism. Example solutions are Ethernet Powerlink (EPL) [EP], Time-Critical Control Network (TCnet) [Eth04d], Time-Triggered Ethernet (TTE) [KAGS05] etc..

- *Modified Ethernet* approach modifies the Ethernet mechanism and infrastructure for real-time performance. It provides real-time services based on modifications in the hardware of the network infrastructures. This approach demands high synchronization guaranteed either by master-slave scheduling or by other synchronization protocols, like IEEE 1588 [EL02], in order to guarantee hard real-time requirements. CSMA/CDR [LR93], SERCOS [C+95], EtherCAT [JB04, Eth04a], PROFINET Isochronous RT (IRT) [Fel04, Eth04c] etc. are proposed solutions.

Some real-time applications, for example the real-time avionics application, demands hard real-time constraints and fully distributed network solution without a global clock (no global synchronization). In that case, the *Top of TCP/IP* approach cannot satisfy the stringent timing constraint, while the *Top of Ethernet* approach which is built on a master-slave scheduling and the *Modified Ethernet* approach which requires network synchronization cannot provide the required network architecture.

The switched Ethernet network has been chosen as a solution to some real-time applications, like the Avionics Full-DupleX switched Ethernet (AFDX) network [ACC08]. Such a technology considers a full-duplex switched Ethernet without a global clock. It uses statically defined routing and an upper bounded switching latency in each switch in order to ensure the real-time requirement. The flows emitted at each source node are shaped in order to guarantee the real-time characteristic. The network model and flow model are presented in the following paragraphs.

1.3 Real-time switched Ethernet network

A real-time switched Ethernet network is a network able to provide a determined data transmission service. The full-duplex communication eliminates the collision domains, but it shifts the problem to the switch level. Main assumption of the network is that the end-to-end delay of each flow should be upper bounded. Several works [LH04a, GRD02, JNTW02, FJJ09] have studied the real-time communication over a switched Ethernet network having the following features:

- full-duplex communication, which eliminates collisions on links;

- static routing mechanism, which uses a static routing table in each switch to avoid dynamic mechanisms such as a spanning tree.

- traffic shaping, which controls at each source node the maximum rate at which the traffic is sent. More precisely, the traffic shaping technique guarantees the minimum inter-frame duration between two consecutive frames of a flow. This is a main characteristic of a real-time switched Ethernet network since without a shaped traffic entering the network, it would be impossible to upper bound the end-to-end delays.

In our work, we consider the same network model which is described in the following paragraphs.

1.3.1 Network model

The real-time switched Ethernet network architecture is composed of a set of nodes interconnected by a full duplex switched Ethernet. It is a homogeneous single network.

The inputs and outputs of the network are nodes. Each node manages a set of flows and emits flows through an output port with a buffer supporting a scheduling strategy (First In First Out, FIFO for short, as an example). It can be connected to only one port of a switch and each port of a switch can be connected at most to one node.

Each switch uses a classical IEEE 802.1d store and forward policy. It has one buffer at each output port which supports a scheduling strategy. It receives frames from input ports and forwards them to the corresponding output ports based on a static routing table. There is a switching latency to deal with the frame forwarding between an input port and an output port of a given switch and it is upper bounded by a known value sl.

Links between switches are full-duplex defined by IEEE 803.1e. The full duplex characteristic guarantees that there are no collisions on links. The bandwidth (transmission rate) of the network is denoted by R ($100Mbit/s$ for example).

The nodes and the switches are not synchronized due to the lack of a global clock. A sample network architecture is depicted in Figure 1.1. It includes four nodes N_1, N_2, N_3 and N_4 interconnected by two switches S_1 and S_2 via full-duplex links.

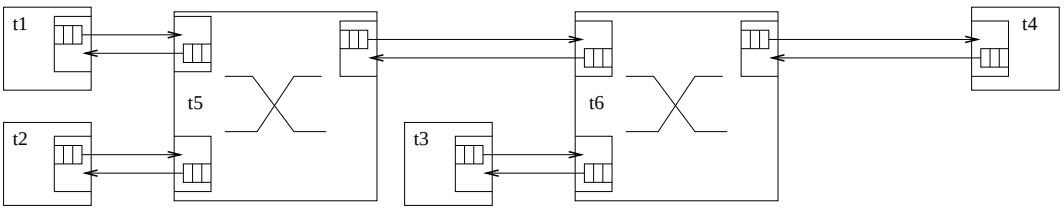


Figure 1.1: Example of the network architecture

1.3.2 Flow model

We assume that n flows τ_i, $i \in [\![1, n]\!]$ are transmitted over the network in order to exchange data. Each flow is unidirectional. It has one emitter and a non empty set of receivers (multicast). Thus n_i paths $\mathcal{P}_{i,j}$ ($j \in [\![1, n_i]\!]$) are associated to each flow τ_i. Each path is defined by a sequence of output ports $first_i, ..., last_{i,j}$ and a destination node $dest_{i,j}$. These notations are summarized in Figure 1.2.

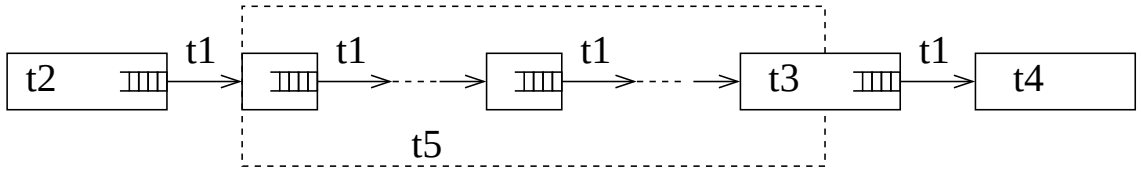


Figure 1.2: An illustration of a static flow path $\mathcal{P}_{i,j}$

Figure 1.3 shows flows transmitted on the network architecture in Figure 1.1. For instance, τ_1 in Figure 1.3 follows only one path $\mathcal{P}_{1,1} = \{N_1, S_1, S_2, N_4\}$. Its source node is $first_1 = N_1$. Its last visited output port is $last_{1,1} = S_2$ and its destination node is $dest_{1,1} = N_4$.

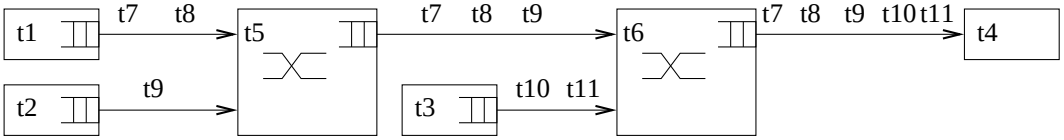


Figure 1.3: Example of mapping flows on the network architecture

Both sporadic flows (SF) and periodic flows (PF) are transmitted. The temporal features of a sporadic or periodic given flow τ_i are defined by the following parameters:

- the minimum inter-frame duration T_i,
- the maximum transmission time of one frame C_i,
- the maximum release jitter J_i which is the maximum delay between the generation time of a frame and its arrival at the output port of the source node, and
- the offset O_i which is the generation time of the first frame of τ_i.

Figure 1.4 summarizes these temporal features for both a sporadic and a periodic flow. $f_{i,j}$ denotes the jth frame of τ_i. The generation times are represented by $\downarrow$ and the frame arrivals in the first output port are represented by $\uparrow$. Each frame arrives within an interval of J_i after its corresponding generation time. The generation time of the first frame of τ_i at its source node is the offset O_i. The offset of a given flow can be known or not.

Classically, the inter-frame duration is always exactly T_i for a periodic flow τ_i. It is at least T_i for a sporadic flow. In a typical industrial application, up to 80% of the flows are periodic

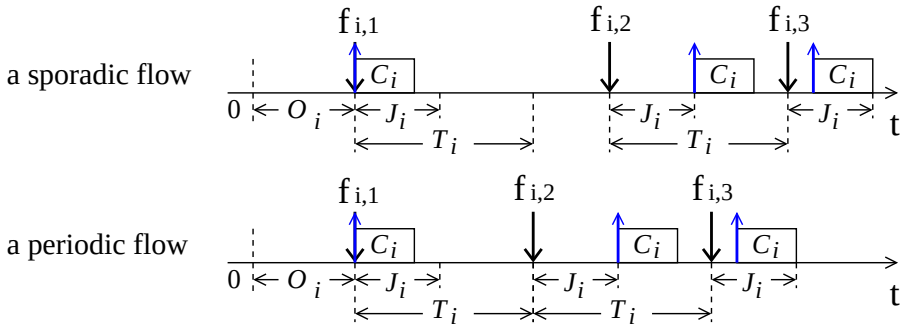


Figure 1.4: Temporal characteristics of a sporadic flow and a periodic flow

[JNTW02]. A periodic flow with a strict period and an offset leads to an exact knowledge of frame generation times at the source node.

1.3.3 A real-time switched Ethernet example: AFDX network

Avionics Full DupleX Switched Ethernet (AFDX) [ACC08] is a typical real-time switched Ethernet network. It has been defined in the context of avionics and developed for modern aircraft such as Airbus A380. It interconnects a set of *end systems* (ES) by a full-duplex switched Ethernet network. Each flow transmitted on this network is called a *Virtual Link* (VL). It is a multicast sporadic flow with static routing (Actually many VLs are periodic). The minimum inter-frame duration T_i is called the *Bandwidth Allocation Gap* (BAG_i). Possible values range in powers of 2 from 1 ms to 128 ms. The transmission time of a frame depends on the frame length and on the rate R of the link. Each VL v_i defines a minimum frame length and a maximum frame length (l_{min_i} and l_{max_i}) which respect the standard Ethernet frame. Thus, the maximum transmission time C_i of one frame of VL v_i is computed by:

$$C_i = \frac{l_{max_i} * 8}{R}$$

J_i and O_i follow the definitions in Section 1.3.2.

1.4 End-to-end delay analysis on a real-time switched Ethernet

The following paragraphs characterize the end-to-end delay of a frame transmitted on a real-time switched Ethernet.

1.4.1 End-to-end delay

Let us consider one frame of a flow τ_i. For the sake of simplicity, this frame is denoted f_i. This frame follows one of the paths of τ_i. For the sake of simplicity, this path is denoted $\mathcal{P}_i = [first_i, ..., last_i, dest_i]$ (We omit the index of the path).

The end-to-end (ETE) delay of a flow τ_i is the time elapsed between the generation time of f_i at its source node $first_i$ and the reception time of f_i at its destination node $dest_i$. The ETE delay R_i of f_i is given by:

$$R_i = j_i + LD_i + SD_i + WD_i \tag{1.1}$$

where

j_i is the release jitter of f_i;
LD_i is the transmission time on links along the considered path $\mathcal{P}_i$;
SD_i is the delay caused by switching latency sl of all the visited switches;
WD_i is the waiting delay caused by competitions with other frames in the output buffers.

The propagation delay is not considered in this work since it is considered negligible. For a twisted pair cable of 100 meter, the maximum length for a cable segment per TIA/EIA 568-5-A [PAN04], the propagation delay is about 560 nanoseconds. Therefore, the propagation delay is not taken into account in the end-to-end delay computation.

Let us consider the end-to-end delay of the first frame f_1 of flow τ_1 in Figure 1.3. It follows the path $\mathcal{P}_1 = \{N_1, S_1, S_2, N_4\}$. One possible scenario is depicted in Figure 1.5. f_1 is generated at its source node N_1 at its offset time O_1. It experiences a release jitter j_1 ($0 \leq j_1 \leq J_1$) and arrives at the output port of N_1 at time $a_{f_1}^{N_1}$. Since another frame f_2 of flow τ_2 arrives at the same output port earlier than f_1 and does not finish its transmission when f_1 arrives, f_1 waits in the buffer till the transmission of f_2 finishes. τ_2 is called an *interference* (or a *competing*) flow of τ_1 since it shares an output port with τ_1.

f_1 arrives at the switch S_1 where it first experiences an upper bounded switching latency sl. Then f_1 waits in the buffer of the output port of S_1 due to the transmission of f_2 which was delayed by a frame f_3 of τ_3. Similarly, at the switch S_2, f_1 experiences a switching latency sl and delays caused by the transmissions of other frames (f_3, f_5 and f_2). Finally, f_1 arrives at its destination node N_4. In Figure 1.5, the transmission delay LD_1 and the waiting delay WD_1 of f_1 are indicated by blocks with different patterns. The release jitter j_1 and switching latency at each visited switch are also marked in Figure 1.5. The end-to-end delay R_1 of f_1 is shown by a shadow block in Figure 1.5.

Figure 1.5 depicts one possible scenario leading to one possible value of the end-to-end delay. The following paragraphs characterize the minimum and the maximum values of this end-to-end delay.

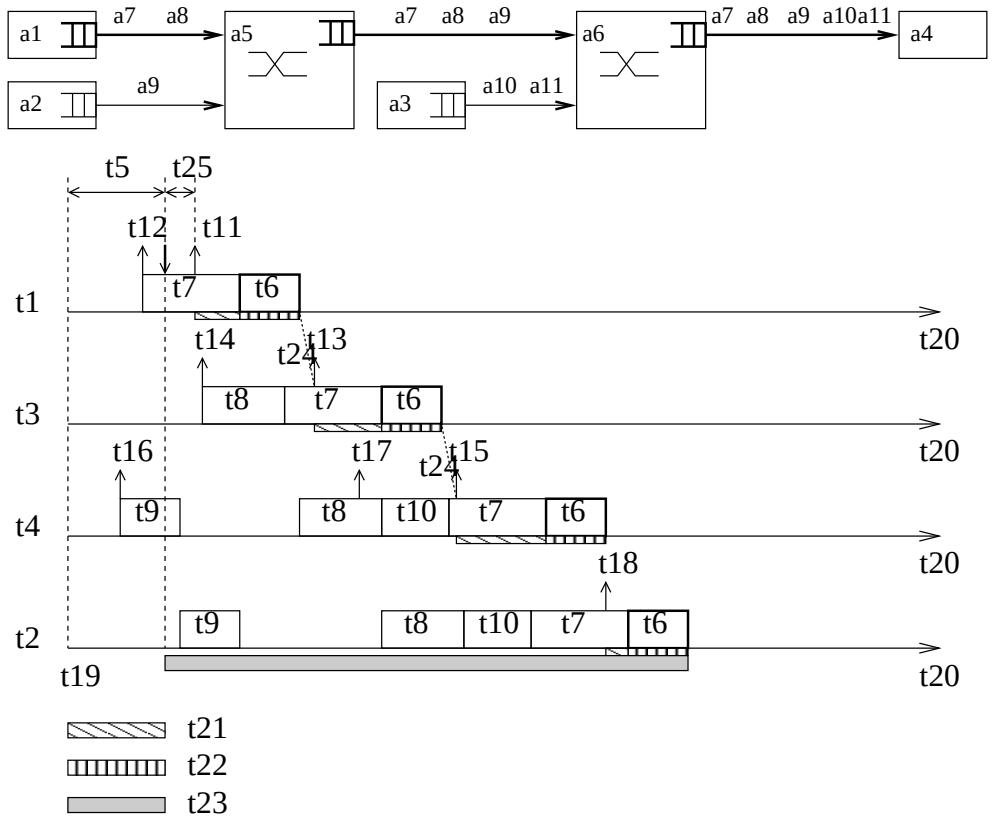


Figure 1.5: Illustration of end-to-end delay of flow τ_1

1.4.2 Minimum end-to-end delay

The minimum end-to-end delay of a flow τ_i happens when τ_i does not experience release jitter or competition with other frames for the output ports along its path. The lowest possible value of end-to-end delay is obtained when $j_i = 0$ and $WD_i = 0$ (no waiting delay in output buffers). Thus, the minimum ETE delay of a frame is the sum of LD_i and SD_i. The minimum delay of flow τ_1 is illustrated in Figure 1.6.

LD_i is obtained by considering that one frame of τ_i with minimum length l_{min_i} is transmitted. $|\mathcal{P}_i|$ is the number of nodes in the path $\mathcal{P}_i$. Then the frame of τ_i crosses $|\mathcal{P}_i| - 1$ links, and we have:

$$LD_i = (|\mathcal{P}_i| - 1) \times \frac{l_{min_i}}{R}$$

Since τ_i crosses $|\mathcal{P}_i| - 2$ switches, we have:

$$SD_i = (|\mathcal{P}_i| - 2) \times sl$$

Thus the minimum end-to-end delay of a flow can be easily computed.

1.4.3 Maximum end-to-end delay

The maximum end-to-end delay is obtained when each term in Formula 1.1 is upper bounded. Let us examine each of those terms:

- The release jitter j_i has a maximum value J_i which is a feature of the flow.
- The transmission delay on links is maximized when a frame with maximum transmission time C_i is transmitted. Thus we have:

$$LD_i = (|\mathcal{P}_i| - 1) \times C_i$$

- The switching latency is upper bounded. Thus we have:

$$SD_i = (|\mathcal{P}_i| - 2) \times sl$$

- The waiting delay WD_i in output buffers depends on the arrival times of competing frames at each output port crossing τ_i. This is illustrated in Figure 1.5 and Figure 1.7. Both figures consider the same frame f_1 (the first frame) of flow τ_1. Since the generation time of the other frames (f_2, f_3, f_4 and f_5) are not the same time, the waiting delay of f_1 changes, leading to different end-to-end delays.

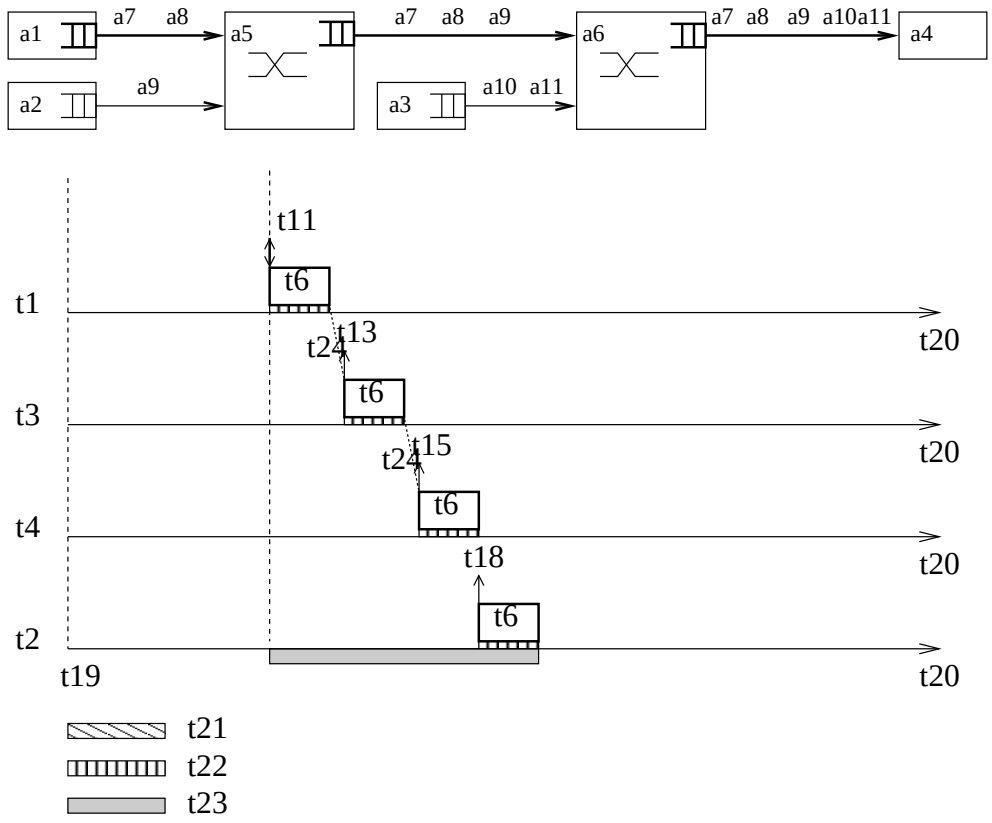


Figure 1.6: Illustration of minimum end-to-end delay of flow τ_1

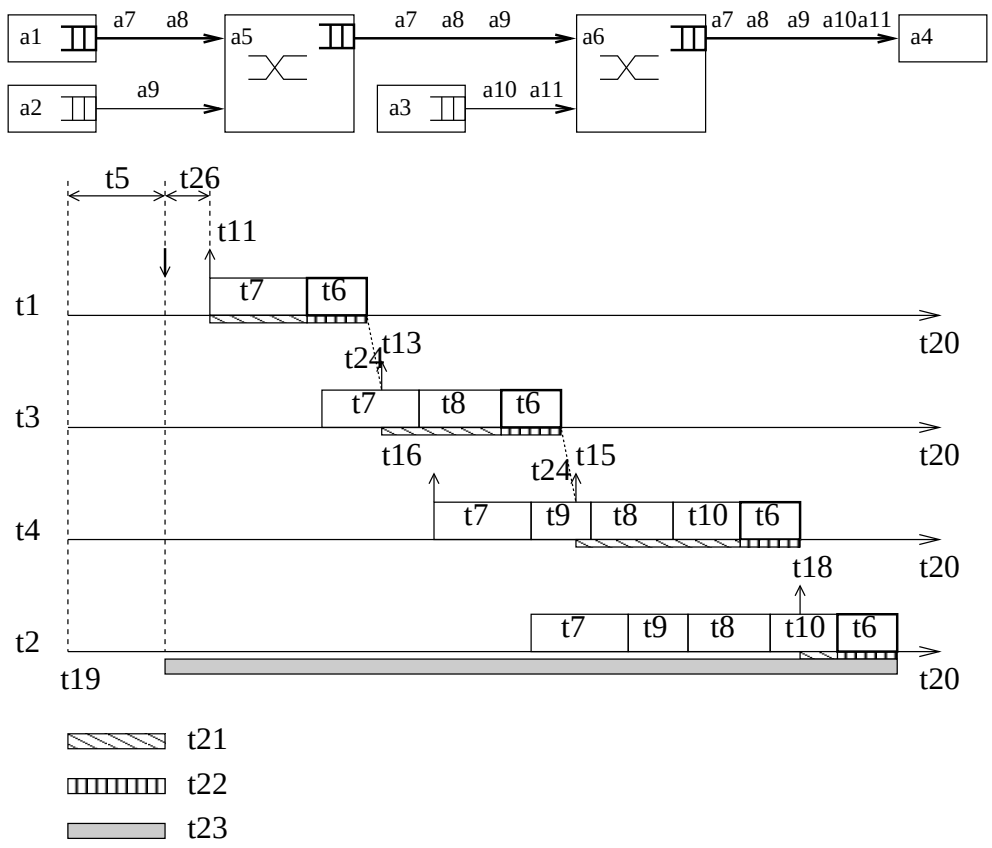


Figure 1.7: Illustration of end-to-end delay of flow τ_1 on a possible scenario

We can conclude that in order to get the maximum end-to-end delay, we have to determine the maximum possible value of WD_i. Up to now, this is still an open problem. Some approaches have been proposed in order to solve this problem. They are summarized in the following section.

1.5 Existing approaches for worst-case delay analysis

Three groups of approaches have been investigated for the worst-case delay analysis of a real-time switched Ethernet network.

- The goal of the first group is to calculate the end-to-end delay on a set of scenarios. It leads to a distribution of the delay. The highest delay of this distribution is an observed worst-case delay. The typical approach of this group is the simulation approach.
- The goal of the second group is to calculate an exact worst-case end-to-end delay. This is achieved by an exhaustive exploration of the possible scenarios. This group of approaches is based on Model-Checking (MC) and they are not scalable.
- The goal of the last group is to compute a sure upper bound of the end-to-end delay. These approaches (mainly the holistic approach [TC94, TBW95a], the Network Calculus approach ,NC for short, and the Trajectory approach) upper bound all the parts of the delay. Most of the time, they are scalable.

Figure 1.8 summarizes these different groups of approaches. When possible, the Model-Checking approach gives the exact worst-case end-to-end delay. Otherwise, the other approaches give an interval where these worst-case delays are. The following paragraphs give a short overview of the existing calculations for all the approaches.

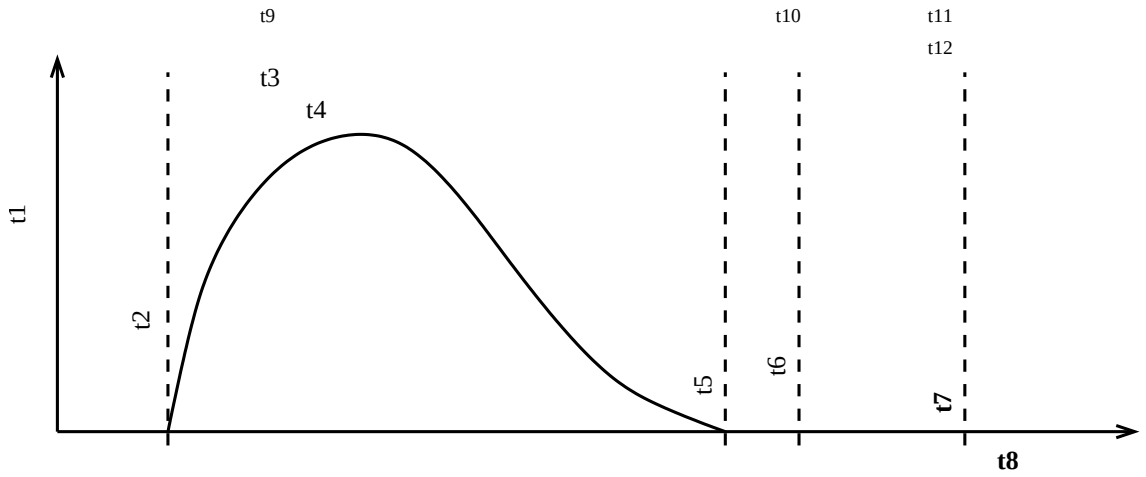


Figure 1.8: The end-to-end delay characteristics

1.5.1 Simulation

A simulation approach is based on a model of the system. The goal is to get informations concerning the typical behaviors of the system. These informations are accurate as long as the model captures the features of the systems. A simulation approach has been proposed for the temporal analysis of a switched Ethernet network [GRD02, CSEF06, SRF09, DH03, SF07, MFF07]. The goal was to obtain the end-to-end delay distribution of a given flow. The main issue is to cope with the huge set of possible scenarios when an industrial network configuration is considered. One solution has been proposed in [SF07, SRF09]. It consists in focusing the simulation on the part of the network configuration which has an influence on the distribution of the considered flow. With this approach, it is possible to analyze most of the flows of a typical avionics configuration. [GRD02] considers a switched Ethernet network with shaped traffic, while [DH03] analyzes a similar network with a time-division scheme. The simulation approach gives valuable informations concerning the delay distribution and a maximum observed delay of each flow. However, this maximum observed delay is most of the time smaller than the exact worst-case delay, since this worst-case is often a rare event which is missed by the approach. This approach is not considered in the rest of this thesis.

1.5.2 Model-checking

Model-checking [BBF+10] allows the automatic verification of software and reactive systems. It performs a reachability analysis on a model of the system. There exist different formalisms for the modeling of the system. Timed automata [AD94] is one of the most popular among these formalisms. These automata describe the system behaviors with times. A model is composed of a set of finite automata with a set of clocks (real and possible variables increasing uniformly with time). Preliminary approaches have been proposed in the context of a mixed TTCAN/switched Ethernet architecture [ESF06] and an avionics switched Ethernet [CSEF06]. The later one cannot cope with more than 8 flows due to the well-known combinatorial explosion problem. A promising idea to miligate this problem is to drastically limit the research space. It consists in considering only the scenarios which are candidates to the worst-case. In [ASF11, ASEF12], properties of such scenarios have been established. Thanks to these properties, it is possible to get the exact worst-case delay for network configurations with up to 60 flows, which is a significant improvement. However, up to now, this approach cannot cope with industrial configurations with 1000 flows. Thus it will not be considered in the rest of this thesis.

1.5.3 Network Calculus

Network Calculus was first proposed by Cruz [Cru91], and then applied to guaranteed service networks by Le Boudec *et al.* [Bou98, BT01]. This approach is considered as a holistic approach. The holistic approach considers the worst-case scenario on each node visited by a flow, accounting for the maximum possible jitter introduced by the previous visited nodes. A generation of the Network Calculus approach has been applied to the switched Ethernet network in order to guarantee its real-time communication [FJJ09, CEL05, LMS05, GRD02,

LH04a, LH04b, JNTW02]. In [GRD02], the real-time characteristics of a switched Ethernet with shaped traffic are evaluated by the Network Calculus which computes delay upper bounds. It provides the worst case delay upper bounds which the simulation method cannot calculate. [GRD02] presents a switched Ethernet model with a shared-memory architecture in each switch. The Network Calculus is developed to evaluate the maximum delay on such a switch Ethernet. [LH04a] uses the traffic shaping techniques to implement hard real-time distributed systems on commodity switched Ethernet. It shows that the delay bounds, derived based on the Network Calculus, depend on the traffic shaping. [LH04b] extends the work in [LH04a] by using firmware offloading to lower the CPU utilization. [CEL05] focuses on the scenarios at the output port of a FIFO multiplexer. It demonstrates that iteratively applying the "optimal" output bounds when flows pass through several FIFO nodes does not guarantee the overall tight bound. [LMS05] refines the service curves of the Network Calculus to improve end-to-end delay bounds for FIFO aggregates. [FJJ09] improves the delay bounds of a packet-switched network by refining the Network Calculus for the source node and the switch connected to the source node. In [FFG06], this approach has been improved by considering serialization effect, which refers to the fact that frames transmitted by the same input link are serialized and cannot arrive at the output port at the same time. The Network Calculus provides delay upper bounds with pessimistic computation. Then these upper bounds can be larger than the exact worst-case delays, as shown in Figure 1.8.

Network Calculus approach provides a theoretical framework to analyze performance guarantees (backlog bound, delay bound and output flow) in a network. It computes the delay upper bound at a node h by considering a maximum arriving flow traffic and a minimum service capacity provided by node h. The maximum arriving traffic is modeled by the arrival curve $\alpha(t)$ for $t \geq 0$, which limits the traffic sent by source nodes by an upper bounded curve. It means that on any time window of width τ, the number of bits for the flow is limited by $\alpha(\tau)$. The minimum service provided by node h is modeled by the service curve $\beta(t)$ for $t \geq 0$ which guarantees that on any time window of width τ, the traffic of at least $\beta(\tau)$ can be served. Therefore, the delay upper bound is determined by considering the maximum time interval between the arrival of an amount of traffic till its departure (being served) at node h, which is the maximum horizontal distance between the arrival curve $\alpha(t)$ and the service curve $\beta(t)$, and presented by $h(\alpha, \beta)$. For several flows multiplexed at a node h, the overall arrival curve is the sum of the arrival curve of each arriving flow, which corresponds to the case where a frame of each arriving flow arrives at the same time at the node h. The details of the approach can be found in Appendix A, and a brief review of the approach applied on a switched Ethernet can be found in Chapter 2.

1.5.4 Trajectory approach

Trajectory approach was first proposed in [MM06a] to cope with worst-case response time of flow with FIFO scheduling, and it was expended for the Fixed Priority (FP) scheduling in [MM06b] and for the Non-preemptive Fixed Priority/Earliest Deadline First (FP/EDF) scheduling in [MMG06]. It is further applied to an avionics switched Ethernet network with FIFO scheduling in [BSF09, BSF10] as well as with FP scheduling in [BSF12] with an integration of frame serialization. In [MMG05], it has been shown that the Trajectory approach is

less pessimistic than the holistic approach for a distributed system. In [BSF10], it has shown interesting properties of worst-case delay analysis on the avionics application as it can bring tighter delay upper bounds than the Network Calculus approach. Therefore it is interesting to consider the Trajectory approach for the delay upper bound computation. The Trajectory approach calculates pessimistic delay upper bounds which can be larger than the exact worst-case delays, as shown in Figure 1.8.

The Trajectory approach considers the worst-case scenario that can happen to a frame along its trajectory instead of considering the worst-case scenario on each node visited by a flow. Therefore it is not holistic. The approach computes the delay upper bounds by maximizing each part of delay generated along the considered path as illustrated in Figure 1.7. The release jitter of a considered flow is upper bounded by the maximum value. The switching latency is upper bounded by the value sl and the number of switches visited by the considered flow. The transmission delay can be maximized by considering the largest frame transmission at each visited node. The waiting delay generated by other competing flows is maximized by considering the maximum possible number of frames of each competing flow that can delay the considered flow. Therefore, the computed delay upper bound is the sum of the upper bound of each part of delay. The details of the approach can be found in Appendix B, and a brief review of the approach applied on a switched Ethernet can be found in Chapter 3.

1.6 Temporal constraints of dependent flows

1.6.1 Minimum duration

The three approaches for the worst-case delay analysis on a real-time switched Ethernet network make no assumptions on the generation instants of the frames (except that they are shaped). This is realistic when flows are generated by different nodes since these nodes do not have a common clock. This is not always the case for flows generated by the same node. Indeed periodic flows are constrained by their offsets and periods. Then periodic flows emitted by the same source node are scheduled based on the same local clock, and therefore they are called locally synchronized and they are dependent. The dependencies lead to a limitation on the number of possible scenarios at the output ports. We will show in the following chapters that integrating this limitation in the worst-case delay analysis can significantly reduce the obtained upper bounds on the delays.

Let us illustrate the impact of offsets and periods for periodic flows generated by the same node. We consider a reference network example in Figure 1.9. This network example supports FIFO scheduling. The transmission rate is $R = 100\ Mbit/s$ and the switching latency is $sl = 10\ \mu s$.

Five flows are transmitted over the network example. There are two periodic flows τ_1 and τ_2 generated at node N_1, and two periodic flows τ_3 and τ_4 generated at node N_2. Flow τ_5 is an independent flow emitted by node N_3. They have parameters listed in Table 1.1:

The hyper-period of a flow τ_i and a flow τ_j is the Least Common Multiple of periods T_i and T_j, i.e., $lcm(T_i, T_j)$. Since the hyper-period $lcm(T_i, T_j)$ is the multiple of both T_i and T_j,

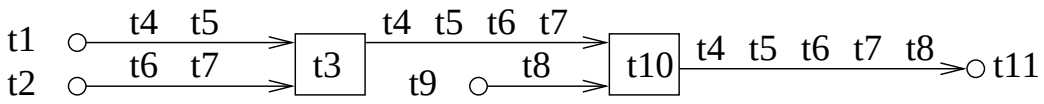


Figure 1.9: A reference network example

τ_i	$T_i(\mu s)$	$C_i(\mu s)$	$O_i(\mu s)$	$J_i(\mu s)$
τ_1	2000	40	0	0
τ_2	4000	40	3500	0
τ_3	4000	40	0	0
τ_4	8000	40	1000	0
τ_5	16000	40	0	0

Table 1.1: Flow parameters of the reference network example

the frame generation sequence of these two flows infinitely repeats since flows are periodic. Therefore, we consider the frame generation sequence in one hyper-period. We focus on flows τ_1 and τ_2 in Figure 1.9. Figure 1.10 depicts the sequence of frame generations of τ_1 and τ_2 on a hyper-period ($lcm(T_1, T_2) = 4000\ \mu s$).

Based on the periods and offsets, frames of τ_1 are ready at the time instant $k * 2000\ \mu s$, while frames of τ_2 are ready at the time instant $3500 + l * 4000\ \mu s$. Thus there is a minimum duration of 1500 μs from the arrival time of a frame of τ_1 to the arrival time of a frame of τ_2 at the output port of N_1. The minimum duration is 500 μs in the reverse order.

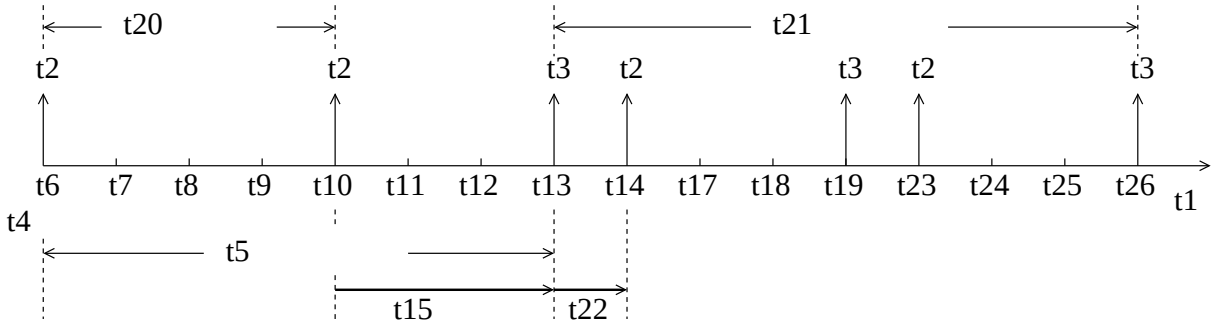


Figure 1.10: Minimum durations between τ_1 and τ_2 at the output port of N_1

Release jitters are null in Table 1.1 and in Figure 1.10. We consider release jitters of flows as shown in Table 1.2.

Integrating these release jitters leads to the sequence in Figure 1.11. $\downarrow$ represents the frame generations and $\uparrow$ represents the frame arrivals. Each frame can be delayed and is ready at any time instant of an interval (maximum release jitter) after its generation. Frames of τ_1 are ready in the intervals $[k * 2000, 500 + k * 2000]$, while frames of τ_2 are ready in the intervals $[3500 + l * 4000, 3600 + l * 4000]$. Thus there is a minimum duration of 1000 μs from the arrival time of a frame of τ_1 to the arrival time of a frame of τ_2 at the output port of N_1. This

τ_i	$T_i(\mu s)$	$C_i(\mu s)$	$O_i(\mu s)$	$J_i(\mu s)$
τ_1	2000	40	0	500
τ_2	4000	40	3500	100
τ_3	4000	40	0	0
τ_4	8000	40	1000	0
τ_5	16000	40	0	0

Table 1.2: Flow parameters of the reference network example with release jitters

minimum duration is 400 μs in the reverse order.

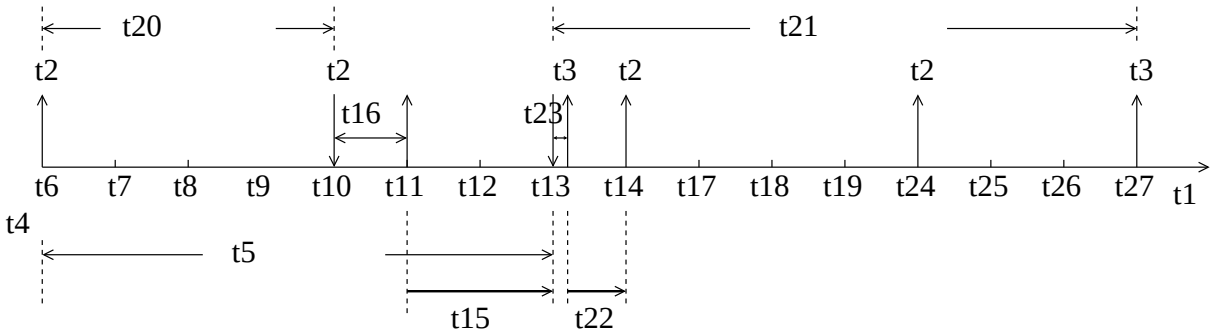


Figure 1.11: Minimum duration between τ_1 and τ_2 at the output port of N_1 with release jitters

For the purpose of simplicity, we consider Table 1.1 as the flow parameters in the rest of the thesis.

Such a duration can be determined for any couple of flows at one shared output port. This minimum duration is null as soon as one of the two flows is not periodic or when the two flows are not generated by the same node.

More formally, the minimum duration from any frame f_i of τ_i to any frame f_j of τ_j transmitted after f_i at the output port h is denoted $MD^h_{i,j}$. $MD^h_{i,j}$ is null as soon as τ_i or τ_j is not periodic or when the τ_i and τ_j are not generated by the same node.

The minimum duration propagates along the path followed by the flows. However, it can decrease from one node to the following one. This is illustrated in Figure 1.12 considering flow τ_1 in the example in Figure 1.9. It shows that the scheduling of the flows along their path and the evolution of the minimum duration from one frame arrival of τ_1 to one frame arrival of τ_2. The computations of $MD^{N_1}_{1,2}$, $MD^{S_1}_{1,2}$ and $MD^{S_2}_{1,2}$ are detailed in the following paragraphs.

1.6.2 Computation of minimum duration

In order to compute the minimum duration $MD^{first_i}_{i,j}$ from one frame arrival of flow τ_i to one frame arrival of flow τ_j in their source node $first_i$, we have to consider all the frame generations of τ_1 and τ_2 during one hyper-period.

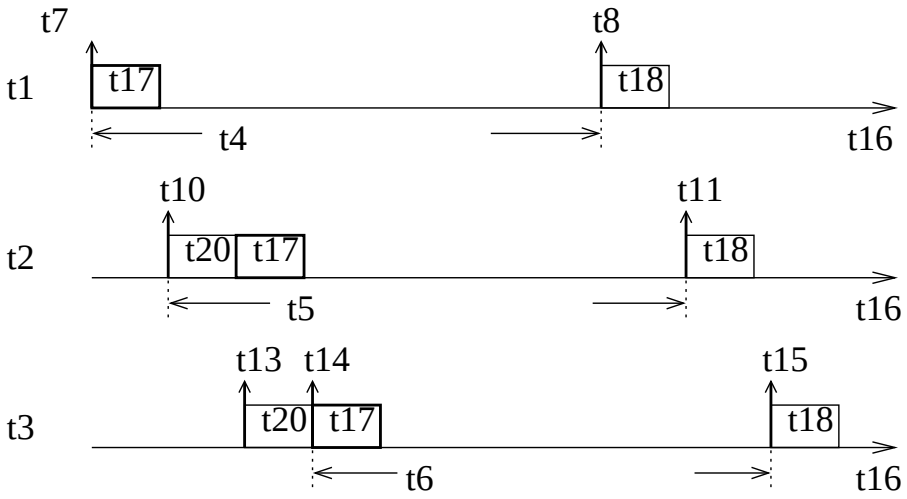


Figure 1.12: Propagation of the *Minimum Duration*

We illustrate the computation on flows τ_1 and τ_2 in the example in Figure 1.9. Figure 1.13 shows frame arrivals of τ_1 and τ_2. Since the hyper-period of τ_1 and τ_2 is 4000 μs, for the computation of $MD_{1,2}^{N_1}$, we consider frame arrivals in the time interval $[O_1, O_1 + 4000) = [0, 4000)$ during which there are the first two frame arrivals of τ_1 and the first frame arrival of τ_2. The duration from the first frame arrival of τ_1 to the first frame arrival of τ_2 is 3500 μs, and the duration from the second frame arrival of τ_1 to the first frame arrival of τ_2 is 1500 μs. Therefore the minimum duration from a frame arrival of τ_1 to a frame arrival of τ_2 is 1500 μs. In the reverse order, we consider frame arrivals in the time interval $[O_2, O_2 + 4000) = [3500, 7500)$. Similarly, the minimum duration from a frame arrival of τ_2 to a frame arrival of τ_1 is 500 μs.

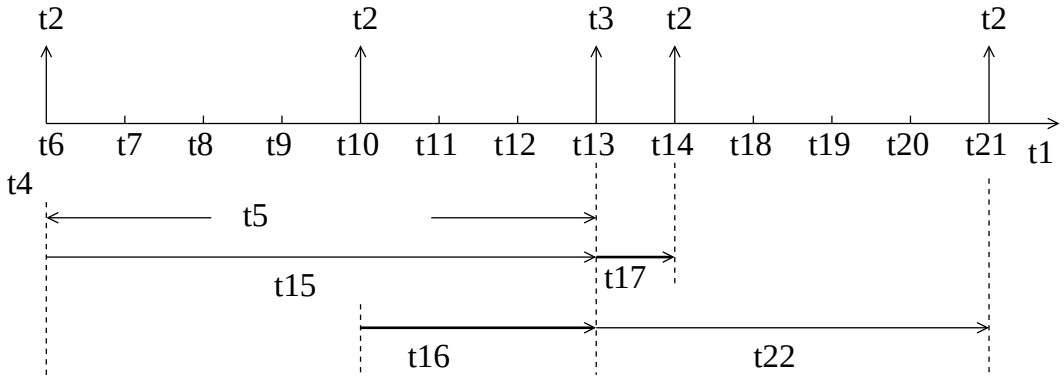


Figure 1.13: Temporal durations from between a frame arrival of τ_1 and a frame arrival of τ_2

The computation of the minimum duration at the source node can be formalized in the following way. Two periodic flows τ_i and τ_j are emitted by their source node $first_i$. They have known offsets O_i and O_j. Let us denote $MD_{i,j}^{first_i}(k, l)$ the time interval from the kth frame

generation of τ_i to the lth frame generation of τ_j. It is computed by

$$MD_{i,j}^{first_i}(k,l) = (T_j \cdot l + O_j) - (T_i \cdot k + O_i) \tag{1.2}$$

We consider frame generations in a hyper-period $lcm(T_i, T_j)$ during which the frame generations are repeated. The first frame generation of τ_i is at time O_i, then we consider frame generations in a time interval $[O_i, O_i + lcm(T_i, T_j))$. Since all the frame generations of τ_i are ready at time instant $O_i + T_i * k$, we have:

$$O_i \leq O_i + T_i * k < O_i + lcm(T_i, T_j)$$

Then it leads to the computation range of k as follows:

$$k \in [\![0, \lceil \frac{lcm(T_i, T_j)}{T_i} \rceil - 1]\!]$$

Similarly, all the frame generations of τ_j are ready at time instant $O_j + T_j * l$, and then we have:

$$O_i \leq O_j + T_j * l < O_i + lcm(T_i, T_j)$$

which leads to the computation range of l as follows:

$$l \in [\![\lceil \frac{O_i - O_j}{T_j} \rceil, \lceil \frac{O_i - O_j + lcm(T_i, T_j)}{T_j} \rceil - 1]\!]$$

Let us consider the computation of $MD_{1,2}^{N_1}(k,l)$. The computation range of k is $[\![0, 1]\!]$, and the computation range of l is $[\![0, 0]\!]$. Then we have all the possible values of $MD_{1,2}^{N_1}(k,l)$ as listed in Table 1.3.

$MD_{1,2}^{N_1}(k,l)$		k	
		0	1
l	0	3500 μs	1500 μs

Table 1.3: All the possible values of $MD_{1,2}^{N_1}(k,l)$

Similarly for the computation of $MD_{2,1}^{N_1}(k,l)$, all the possible values are listed in Table 1.4.

Since $MD_{i,j}^{first_i}$ is achieved only if the frame of τ_i experiences maximum release jitter J_i and the frame of τ_j does not experience any release jitter, the minimum duration $MD_{i,j}^{first_i}$ is the maximum value between 0 and the smallest possible non negative value of $MD_{i,j}^{first_i}(k,l)$ minus J_i:

$$MD_{i,j}^{first_i} = (\min_{MD_{i,j}^{first_i}(k,l) \geq 0} (MD_{i,j}^{first_i}(k,l)) - J_i)^+ \tag{1.3}$$

$MD_{2,1}^{N_1}(k,l)$		k
		0
l	2	500 μs
	3	2500 μs

Table 1.4: All the possible values of $MD_{2,1}^{N_1}(k,l)$

for any k and j in the computation ranges:

$$k \in [\![0, \lceil \frac{lcm(T_i,T_j)}{T_i} \rceil - 1]\!], \quad l \in [\![\lceil \frac{O_i - O_j}{T_j} \rceil, \lceil \frac{O_i - O_j + lcm(T_i,T_j)}{T_j} \rceil - 1]\!]$$

For $N_1 = first_1$ in Figure 1.9, $MD_{1,2}^{N_1}(k,l) = (4000 \times l + 3500) - (2000 \times k + 0)$. Then $MD_{1,2}^{N_1} = 1500\ \mu s$ when $k = 1$, $l = 0$. Similarly, the minimum duration from τ_2 to τ_1 at N_1 is $MD_{2,1}^{N_1} = 500\ \mu s$ when $k = 0$, $l = 2$. They are illustrated in Figure 1.13.

For two flows τ_i and τ_j emitted by the same source node $first_i$ and sharing the output ports from $first_i$ till h, the minimum duration from τ_i to τ_j propagates at each switch output port of their shared path. The general case is depicted in Figure 1.14. The frame f_i of τ_i and the frame f_j of τ_j are separated by $MD_{i,j}^{first_i}$ at their source node $first_i$, which can be determined by Equation 1.3. When f_i and f_j arrive at the output port h, one possible scenario is shown in Figure 1.15.

Frames f_i and f_j experience different delays from $first_i$ to h as they can be delayed by different competing flows. Since the minimum duration $MD_{i,j}^{h}$ from τ_i to τ_j at h is the minimum possible time interval, then it is obtained when the delay experienced by f_i from $first_i$ to h is maximized and the delay experienced by f_j from $first_i$ to h is minimized. It may not be equal to $MD_{i,j}^{first_i}$ as illustrated in Figure 1.15. Hence, it is necessary to compute the minimum duration at each shared switch output port.

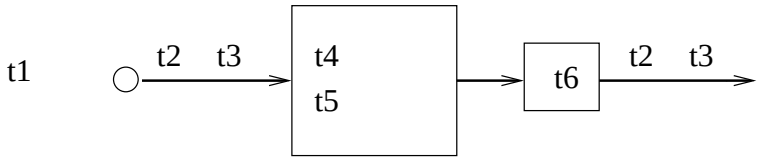


Figure 1.14: General case

The maximum delay of f_i from $first_i$ to h is denoted $S_{max_i}^{h}$, and occurs when f_i is delayed by all possible competing flows at each crossed switch between $first_i$ and h. This delay can be computed by worst-case delay computations which will be introduced in Chapter 2 and in Chapter 3. The minimum delay of frame f_j from $first_i$ to h is denoted $S_{min_j}^{h}$ and occurs when f_j is not delayed by any competing flows. Provided that the transmission rate of switched Ethernet is R, there are n_h links between $first_i$ and h, and the switching latency is sl, the

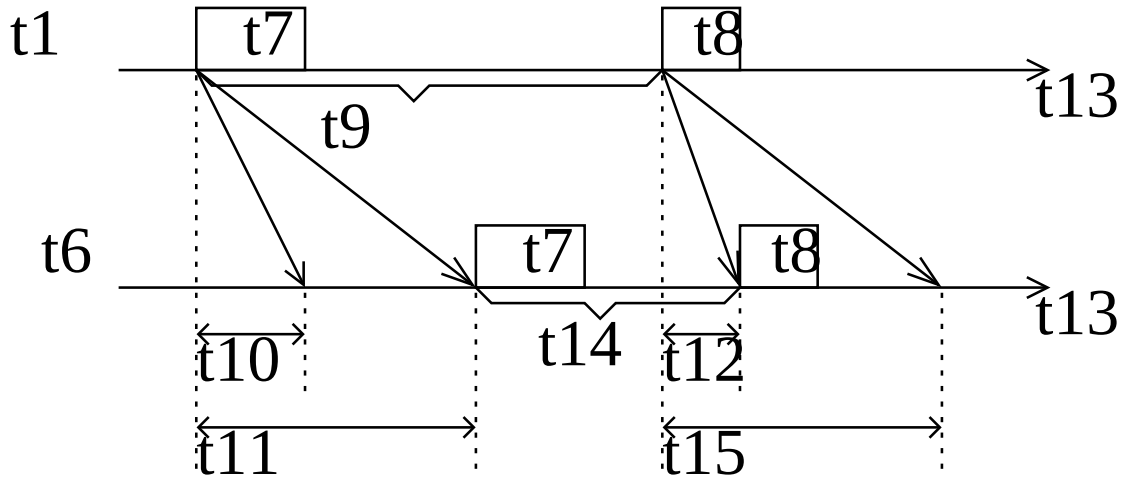


Figure 1.15: One possible scenario

minimum delay is given by:

$$S^h_{min_j} = (\frac{l_{min_j}}{R} \times n_h) + sl \times (n_h - 1) \tag{1.4}$$

Therefore, $MD^h_{i,j}$ is a non negative value and determined by:

$$MD^h_{i,j} = (MD^{first_i}_{i,j} + S^h_{min_j} - S^h_{max_i})^+ \tag{1.5}$$

The computation of the propagated minimum duration is illustrated by the reference example where $MD^{N_1}_{1,2} = 1500\ \mu s$. Since τ_1 and τ_2 are not delayed by any other flow at N_1, when they arrive at S_1, they experience the maximum delays $S^{S_1}_{max_1} = S^{S_1}_{max_2} = 40\ \mu s$. Assuming that these flows have a constant frame length, the minimum delays of τ_1 and τ_2 are $S^{S_1}_{min_1} = S^{S_1}_{min_2} = 40\ \mu s$. Then $MD^{S_1}_{1,2}$ (illustrated in Figure 1.12) is computed by Equation 1.5:

$$MD^{S_1}_{1,2} = (MD^{N_1}_{1,2} + S^{S_1}_{min_2} - S^{S_1}_{max_1})^+ = 1500 + 40 - 40 = 1500\ \mu s$$

This computation propagates to the output port of S_2. Since τ_1 is delayed by flow τ_3 at the output port S_1 (the reason will be explained in the following chapters) while τ_2 is not delayed by any other flows at S_1, when they arrive at S_2, τ_1 experiences a maximum delay $S^{S_2}_{max_1} = 120\ \mu s$ while τ_2 experiences a minimum delay $S^{S_2}_{min_2} = 80\ \mu s$. Then $MD^{S_2}_{1,2}$ (illustrated in Figure 1.12) is computed by Equation 1.5:

$$MD^{S_2}_{1,2} = (MD^{N_1}_{1,2} + S^{S_2}_{min_2} - S^{S_2}_{max_1})^+ = 1500 + 80 - 120 = 1460\ \mu s$$

Similarly, the minimum durations from τ_2 to τ_1 along the path $\{N_1, S_1, S_2, N_4\}$ as well as the minimum durations from τ_3 to τ_4 and from τ_4 to τ_3 along the path $\{N_2, S_1, S_2, N_4\}$ can be

determined by Equation 1.3 and Equation 1.5, and they are listed in Table 1.5.

h	$MD^h_{1,2}(\mu s)$	$MD^h_{2,1}(\mu s)$	$MD^h_{3,4}(\mu s)$	$MD^h_{4,3}(\mu s)$
N_1	1500	500	-	-
N_2	-	-	1000	3000
S_1	1500	500	1000	3000
S_2	1460	460	960	2960

Table 1.5: minimum durations of flows τ_1, τ_2, τ_3 and τ_2

Since N_1 and N_2 do not share one common clock, τ_1 and τ_2 are not synchronized with τ_3 and τ_4. Then the temporal constraints between flows of N_1 (τ_1, τ_2) and flows of N_2 (τ_3, τ_4) are not considered.

These minimum duration constraints should be introduced in the worst-case delay analysis. In the following chapters, we propose such an introduction in both the Network Calculus approach and the Trajectory approach.

1.7 Conclusion

In this chapter, we presented main characteristics of real-time full-duplex switched Ethernet networks using traffic shaping at each source node. The analysis of worst-case end-to-end delay has been illustrated. The state of the practice related to our work has been shortly recalled, as well as the Network Calculus approach and the Trajectory approach that are taken into account in our work.

The existing worst-case delay analysis considers only independent flows exchanged in the network. In this chapter, it has been shown that periodic flows with known offsets emitted by the same source node are dependent. There is a minimum duration between two frames of two dependent flows. These constraints of minimum duration can eliminate some pessimistic scenarios and therefore should be integrated in the worst-case delay analysis. In the following chapters, the integration of the minimum durations in the Network Calculus approach and in the Trajectory approach are implemented.

Chapter 2

Modified Network Calculus approach integrating minimum duration constraints

Contents

2.1 Introduction 31
2.2 Classical Network Calculus approach 32
2.2.1 Arrival curves 32
2.2.2 Service curves 33
2.2.3 Delay computation 34
2.2.4 Frame serialization 35
2.2.5 Limitation of the approach 36
2.3 A modified approach considering minimum duration constraints . 36
2.3.1 General idea 36
2.3.2 Computation overview 37
2.3.3 Computation of the arrival curve of dependent flows 38
2.4 Application on a small network example 41
2.4.1 Classical Network Calculus approach 42
2.4.2 Modified Network Calculus approach 44
2.5 Conclusion 48

2.1 Introduction

The existing Network Calculus approach allows the computation of an upper bound on the delay of flows transmitted on a switched Ethernet network with traffic shaping at each source node. This approach does not make any further assumptions on the generation instants of the frames. We have shown in Chapter 1 that, for periodic flows with known offsets emitted by the same source node, minimum duration constraints exist for the generation instants of the flows. The goal of Chapter 2 is to propose the integration of these minimum duration constraints in the Network Calculus approach.

Section 2.2 briefly reviews the classical Network Calculus approach applied on a real-time switched Ethernet network. Section 2.3 shows the integration of the minimum duration constraints in the Network Calculus approach. Section 2.4 compares the classical Network Calculus approach with the enhanced one on an example configuration. Section 2.5 concludes this chapter.

2.2 Classical Network Calculus approach

The existing Network Calculus approach for FIFO scheduling applied to a real-time switched Ethernet network is summarized in this section. A detailed presentation of the Network Calculus approach can be found in Appendix A.

We use the example in Figure 1.9 to illustrate the classical Network Calculus approach. This example is recalled in Figure 2.1. The transmission rate is $R = 100\ Mbit/s$ and the switching latency is $sl = 10\ \mu s$.

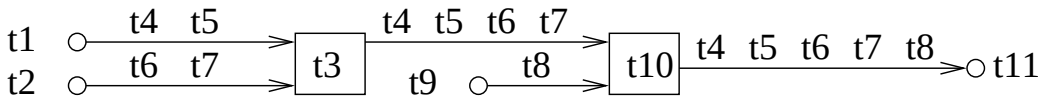


Figure 2.1: Recall of the network example for the Network Calculus approach

Five flows are transmitted over the network example. There are two periodic flows τ_1 and τ_2 generated at node N_1, and two periodic flows τ_3 and τ_4 generated at node N_2. Flow τ_5 is an independent flow emitted by node N_3. They have parameters recalled in Table 2.1.

τ_i	$T_i(\mu s)$	$C_i(\mu s)$	$O_i(\mu s)$	$J_i(\mu s)$
τ_1	2000	40	0	0
τ_2	4000	40	3500	0
τ_3	4000	40	0	0
τ_4	8000	40	1000	0
τ_5	16000	40	0	0

Table 2.1: Recall of flow parameters of the network example for the Network Calculus approach

2.2.1 Arrival curves

We consider real time flows shaped by leaky buckets. Each flow τ_i has a maximum frame size l_{max_i} and a minimum inter-frame distance T_i. Thus each flow is classically modeled by the arrival curve α_i defined by:

$$\alpha_i(t) = r_i t + b_i = \frac{l_{max_i}}{T_i} \times t + l_{max_i}$$

This arrival curve upper bounds the traffic generated by τ_i.

At each output port, the overall arrival curve is obtained by adding the arrival curves of all the input traffics. More precisely, when there are n input flows with arrival curves $\alpha_1, \alpha_2, ..., \alpha_n$ entering an output port, the arrival curve of the aggregated traffic is:

$$\alpha = \sum_{i \in [\![1,n]\!]} \alpha_i$$

As an example, let us consider the output port of N_1 in the example in Figure 2.1. It is crossed by two flows τ_1 and τ_2 with parameters: $l_{max_1} = 4000\ bits$, $T_1 = 2000\ \mu s$, $l_{max_2} = 4000\ bits$, $T_2 = 4000\ \mu s$. Thus, their arrival curves are:

$$\alpha_1(t) = 2t + 4000, \qquad \alpha_2(t) = t + 4000$$

They are illustrated in Figure 2.2.

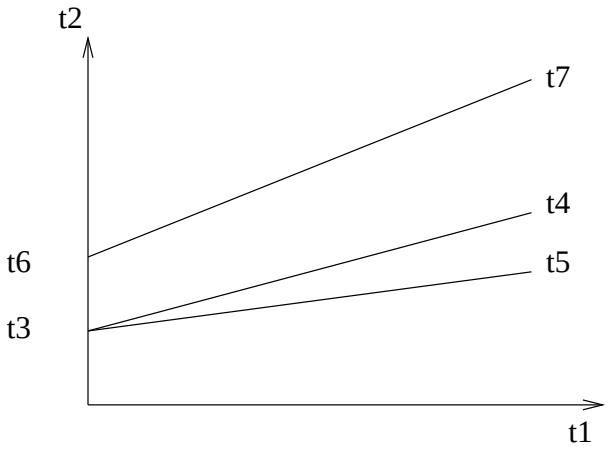


Figure 2.2: The arrival curves of flows τ_1 and τ_2 at N_1

Then the overall arrival curve at N_1 is:

$$\alpha(t) = \alpha_1(t) + \alpha_2(t) = 3t + 8000$$

It is shown in Figure 2.2.

2.2.2 Service curves

An output port of a source node transmits the arriving traffic with a transmission rate R. Thus it provides a service curve $\beta(t) = R(t)^+$. An output port of a switch transmits the arriving traffic with a transmission rate R after an upper bounded switching latency sl. Then it provides a service curve $\beta(t) = R(t - sl)^+$.

As an example, the output port of N_1 in the example in Figure 2.1 has a service curve $\beta(t) = 100(t)^+$, while the output port of switch S_1 has a service curve $\beta(t) = 100(t-10)^+$.

2.2.3 Delay computation

According to the Network Calculus approach, for an output port with an arrival curve α and providing a service curve β, the maximum delay generated at the output port is computed by $h(\alpha, \beta)$, which is the maximum horizontal distance between α and β.

For the output port of N_1, its arrival curve is $\alpha(t) = 3t + 8000$ and its service curve is $\beta(t) = 100(t)^+$, then the maximum delay is obtained by $h(\alpha, \beta) = 80\ \mu s$, as depicted in Figure 2.3.

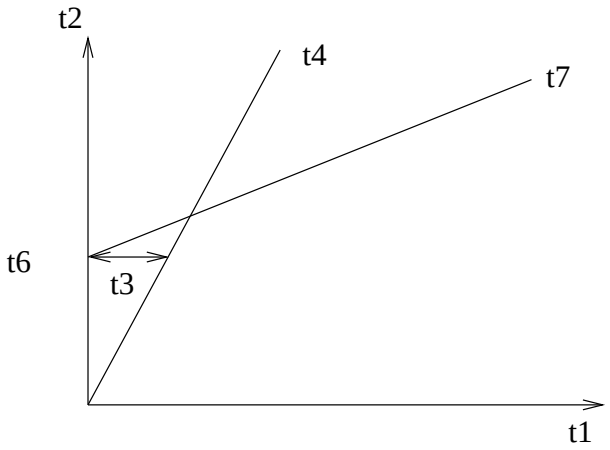


Figure 2.3: The maximum delay generated at the output port of N_1

The end-to-end delay upper bound of τ_i computed by the Network Calculus approach is the sum of the maximum delays computed at each output port visited by τ_i along its path $\mathcal{P}_i$. Then the computation has to be propagated from the source node to the following output port till the last visited output port.

The worst-case end-to-end delay R_1 of τ_1 in the example in Figure 2.1 is the sum of the maximum delays at the output ports of N_1, S_1 and S_2.

The propagation is done in the following manner. At each output port $h-1$ on the path of the considered flow, the output port curve α^h is computed from the overall arrival curve α^{h-1} and the service curve β^{h-1} of the output port. It is illustrated in Figure 2.4.

According to [Gri04], the arrival curve of τ_i at the output port h is constrained by:

$$\alpha_i^h = \alpha_i^{h-1}(t + J_i^{h-1}) \tag{2.1}$$

where J_i^{h-1} is the maximum delay jitter encountered by τ_i in the output port $h-1$.

This output curve becomes an arrival curve for the next output port h on the path of the

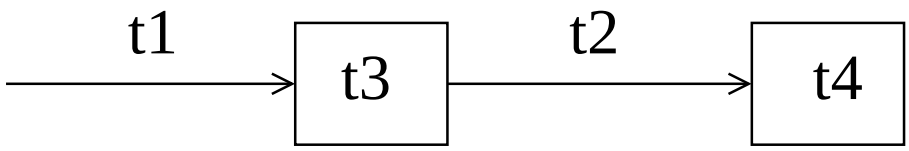


Figure 2.4: Propagation of the arrival curve

considered flow.

We illustrate this propagation of arrival curve of flow τ_1 in the example in Figure 2.1. Its arrival curve at its source node is $\alpha_1^{N_1}(t) = \alpha_1(t) = 2t + 4000$. As we have shown, it experiences a maximum delay 80 μs in the output port of N_1. Since its transmission delay is $C_1 = 40$ μs, its maximum delay jitter in the output port of N_1 is $J_1^{N_1} = 80 - 40 = 40$ μs. Then its arrival curve at the following output port of S_1 is:

$$\alpha_1^{S_1}(t) = \alpha_1^{N_1}(t + J_1^{N_1}) = 2(t + 40) + 4000 = 2t + 4080$$

as illustrated in Figure 2.5.

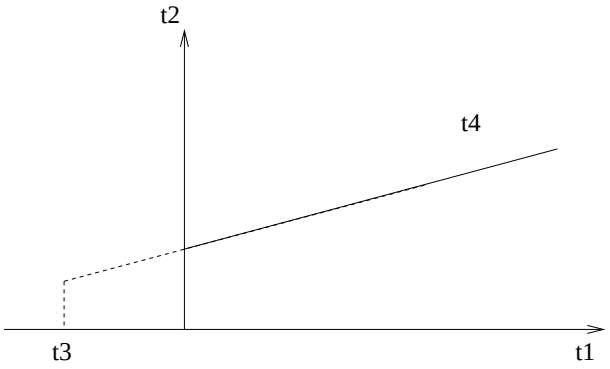


Figure 2.5: The arrival curves of flows

Similarly, for flows τ_2, τ_3 and τ_4 arriving at the output port of S_1 we have:

$$\begin{aligned} \alpha_2^{S_1}(t) &= t + 4040, \\ \alpha_3^{S_1}(t) &= t + 4040, \\ \alpha_4^{S_1}(t) &= 0.5t + 4020 \end{aligned}$$

2.2.4 Frame serialization

Frames coming from the same input link are serialized and they cannot arrive at the output port at the same time. This frame serialization is integrated in the approach by constraining the maximum traffic of each input link with the peak arriving rate R of link and the burst

workload which is the maximum workload arriving at one time.

For example, at the output port of S_1 in Figure 2.1, frames of τ_1 and τ_2 are transmitted by the same input link from N_1 and they are serialized. They cannot arrive at the same time at the output port of S_1 and their transmission rate is limited by the link rate R. Therefore, the sum of $\alpha_1^{S_1}$ and $\alpha_2^{S_1}$ is constrained by the burst workload $\max(b_1^{S_1}, b_2^{S_1}) = 4080\ bits$ and by the link rate R as shown in Figure 2.6. Similarly, the frame serialization of $\alpha_3^{S_1}$ and $\alpha_4^{S_1}$ at the output port of S_1 is illustrated in Figure 2.7.

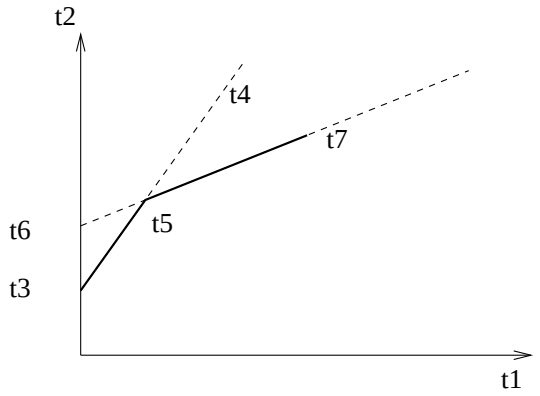


Figure 2.6: Serialization of $\alpha_1^{S_1}$ and $\alpha_2^{S_1}$

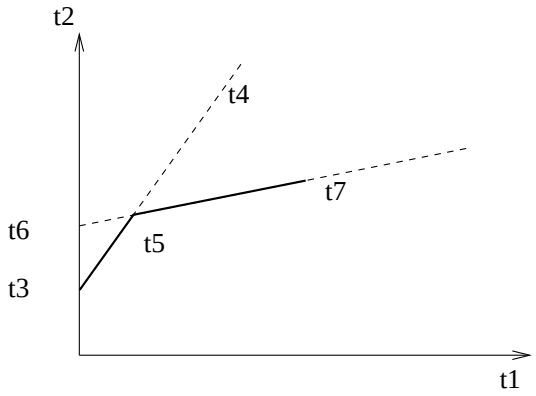


Figure 2.7: Serialization of $\alpha_3^{S_1}$ and $\alpha_4^{S_1}$

2.2.5 Limitation of the approach

The existing Network Calculus approach does not put any constraints between the flows. Typically, it assumes that frames of any two different flows can be generated at the same time. It has been shown in Section 1.6 that such a scenario might be impossible when the two flows are periodic and generated by the same node. Indeed, there can be a minimum duration between the frames of two such flows. In the rest of this chapter, we show how these minimum durations can be integrated in the Network Calculus approach.

2.3 A modified approach considering minimum duration constraints

2.3.1 General idea

The approach summarized in the previous section considers that all the flows are independent. Thus the arrival curves of all the flows crossing an output port are simply added in order to obtain the overall arrival curve at this port. It corresponds to the case where frames from each flow arrive at the output port at the same time. This is illustrated in Figure 2.8 on flows τ_1 and τ_2 at the output port of N_1 in the example in Figure 2.1.

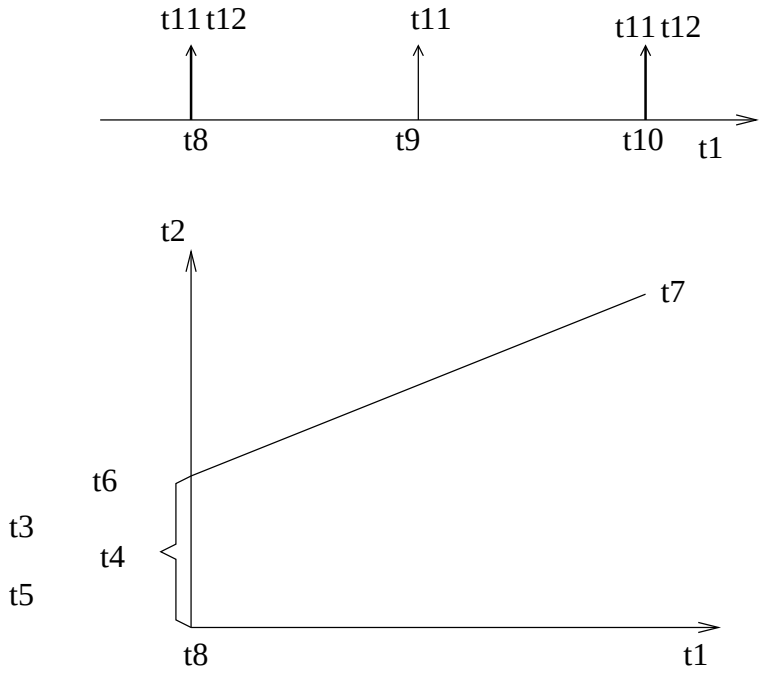


Figure 2.8: Illustration of the burst workload at N_1

The burst 8000 *bits* at time $t = 0$ corresponds to arrivals of one frame of τ_1 and one frame of τ_2 at the same time. However, it has been shown in Section 1.6 that there is at least 1500 μs from one frame of τ_1 to one frame of τ_2 and 500 μs from one frame of τ_2 to one frame of τ_1. Consequently, there will never be such a burst. Thus the idea of the approach is to build, for each set of dependent flows (i.e. flows with minimum durations between them), an aggregated arrival curve which takes into account these minimum durations.

2.3.2 Computation overview

Since the Network Calculus approach propagates the computation port by port along the considered path, the computation is presented for one output port h. Let us consider an output port where n flows compete. The computation of the overall arrival curve at port h processes in the following steps:

1. The n flows are classified in n_g subsets $\mathcal{G}_x$, $x \in [\![1, n_g]\!]$. Any two flows in a given subset have minimum duration constraints between them. Conversely, any two flows in different subsets have no minimum duration constraints between them.

2. An arrival curve $\alpha^h_{\mathcal{G}_x}$ is computed for each subset $\mathcal{G}_x$, $x \in [\![1, n_g]\!]$. When the subset $\mathcal{G}_x$ includes one single flow, $\alpha^h_{\mathcal{G}_x}$ is directly the arrival curve of this flow. When the subset $\mathcal{G}_x$ includes more than one flow, $\alpha^h_{\mathcal{G}_x}$ is an aggregation of the arrival curves of these flows.

3. The overall arrival curve at h which is obtained by adding the arrival curves of the n_g subsets:

$$\alpha^h = \sum_{x \in [\![1, n_g]\!]} \alpha^h_{\mathcal{G}_x}$$

4. The maximum delay at h is the maximum horizontal distance $h(\alpha^h, \beta^h)$ between the overall arrival curve α^h at h and the service curve β^h at h.

Thus, the difference with the approach presented in Section 2.2 is the computation of an aggregated arrival curve for a set of dependent flows. This computation is detailed in the following sections.

2.3.3 Computation of the arrival curve of dependent flows

In order to compute the worst-case delay upper bound, we need to upper bound the arrival traffic of each subset at each output port. For a subset $\mathcal{G}_x$, since the arrival traffic is the sum of all the flows in the subset, there can be a huge number of possibilities. The arrival curve $\alpha^h_{\mathcal{G}_x}$ constrains the maximum arrival traffic of all the flows in subset $\mathcal{G}_x$, and it is determined by the worst-case scenario of the subset $\mathcal{G}_x$ at the output port h. The number of possible arrival curves can be reduced by the following Lemma.

Lemma 1 *A source node has a subset $\mathcal{G}_x$ of dependent flows. The worst-case scenario of the arrival curve $\alpha^h_{\mathcal{G}_x}$ of this subset happens when at least one frame of any flow τ_i ($i \in \mathcal{G}_x$) arrives at h at time $t = 0$.*

Proof: For a subset $\mathcal{G}_x$, if its arrival curve at h happens when there is no frame of any flow τ_i ($i \in \mathcal{G}_x$) arriving at h at time $t = 0$, its burst workload is 0 at time $t = 0$. This burst workload is smaller than any case where there is one frame of any flow τ_i ($i \in \mathcal{G}_x$) arriving at h at time $t = 0$ which gives a burst workload of the size of the arriving frame. Since the arrival curve constrains the maximum arriving traffic, the case where there is no frame of any flow τ_i ($i \in \mathcal{G}_x$) arriving at time $t = 0$ cannot constrain the maximum arrival traffic, and therefore it is not the worst-case scenario for the set $\mathcal{G}_x$. Then the worst-case scenario of the arrival curve $\alpha^h_{\mathcal{G}_x}$ happens when at least one frame of flow τ_i ($i \in \mathcal{G}_x$) arrives at h at time $t = 0$. ■

Based on the Lemma 1, we compute the aggregated arrival curve $\alpha^h_{\mathcal{G}_x}$ at the output port h. We first consider two flows τ_i and τ_j generated by the same source node and classified in a subset $\mathcal{G}_x = \{i, j\}$. Their arrival curves at the output port h are illustrated in Figure 2.9. There are minimum durations $MD^h_{i,j}$ from τ_i to τ_j and $MD^h_{j,i}$ from τ_j to τ_i.

According to the Lemma 1, there is at least one frame from τ_i or τ_j arriving at h at time $t = 0$. We first consider the case when there is a frame of τ_i arriving at $t = 0$. Due to the minimum duration $MD^h_{i,j}$, the first frame arrival of τ_j is delayed to $t = MD^h_{i,j}$, which means

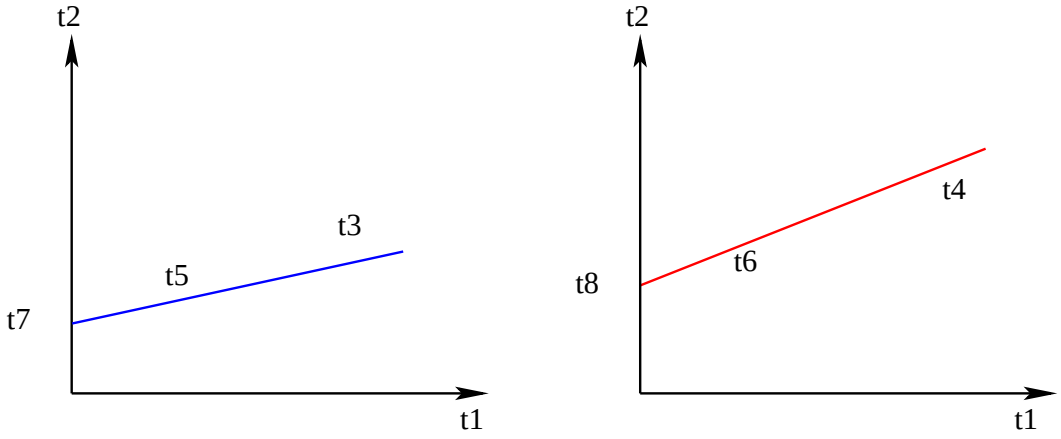
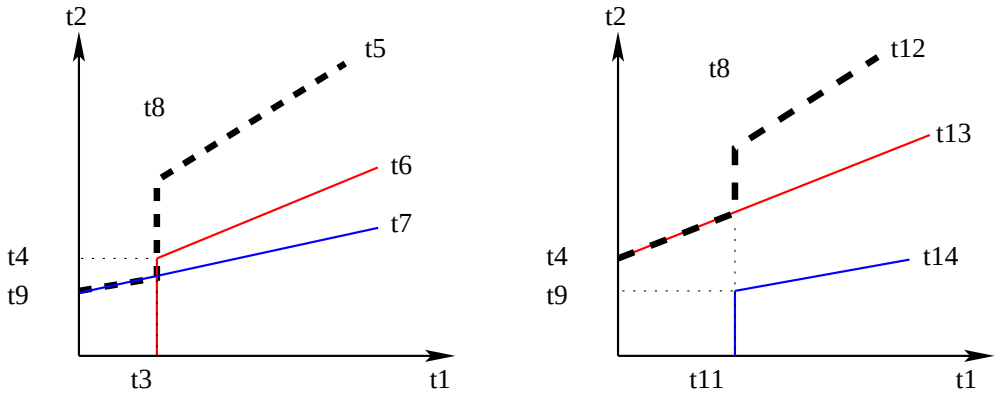


Figure 2.9: The arrival curves of α_i^h and α_j^h



(a) $\alpha^h_{\mathcal{G}_x,\tau_i}$ (b) $\alpha^h_{\mathcal{G}_x,\tau_j}$

Figure 2.10: Sum arrival curves of $\mathcal{G}_x$

that the arrival curve of τ_j is delayed by $t = MD^h_{i,j}$, as illustrated in Figure 2.10(a). This delayed arrival curve is given by:

$$\alpha^h_j(t - MD^h_{i,j})$$

The sum of these two arrival curves is denoted $\alpha^h_{\mathcal{G}_x,\tau_i}$. It indicates that it is the sum of the arrival curves of all the flows in the set $\mathcal{G}_x$ when considering that τ_i has a frame arriving at $t = 0$. Since τ_i and τ_j are the only flows in $\mathcal{G}_x$, we have:

$$\alpha^h_{\mathcal{G}_x,\tau_i}(t) = \alpha^h_i(t) + \alpha^h_j(t - MD^h_{i,j})$$

Similarly, as illustrated in Figure 2.10(b), when we consider that there is a frame of τ_j arriving at h at time $t = 0$, we have a delayed arrival curve $\alpha^h_i(t - MD^h_{j,i})$ of τ_i, and there is another arrival curve $\alpha^h_{\mathcal{G}_x,\tau_j}$ which is the sum of α^h_j and $\alpha^h_i(t - MD^h_{j,i})$.

The arrival curve $\alpha^h_{\mathcal{G}_x}$ of set $\mathcal{G}_x$ has to upper bound the traffic of all the flows in $\mathcal{G}_x$. Thus it has to upper bound all the sum arrival curves built in the previous steps ($\alpha^h_{\mathcal{G}_x,\tau_i}$ and $\alpha^h_{\mathcal{G}_x,\tau_j}$ in the example). It is obtained by taking the piecewise maximum of the sum arrival curves. It is called a *Safe Arrival Curve (SAC)* and denoted $\alpha^h_{\mathcal{G}_x}$.

Considering the subset $\mathcal{G}_x = \{i, j\}$, its *Safe Arrival Curve* is:

$$\alpha^h_{\mathcal{G}_x}(t) = \max(\alpha^h_{\mathcal{G}_x,\tau_i}(t), \alpha^h_{\mathcal{G}_x,\tau_j}(t))$$

The illustration is given in Figure 2.11.

When there are more than two flows in one subset, we have to explore all the possible sum arrival curves in order to build the *Safe Arrival Curve*. For example, there are three flows τ_i, τ_j and τ_k emitted by one source node. These three flows are in one subset $\mathcal{G}_x = \{i, j, k\}$. One possibility is that a frame of τ_i arrives at time $t = 0$ at the output port h, and the earliest frame arrival of τ_j is at time $MD^h_{i,j}$, and then the following earliest frame arrival of τ_k is at time $MD^h_{j,k}$. In this case, the frame arrival order is $\{\tau_i, \tau_j, \tau_k\}$ and it gives one possible sum arrival curve with the integration of $MD^h_{i,j}$ and $MD^h_{j,k}$. Similarly, the frame arrival order can also be $\{\tau_i, \tau_k, \tau_j\}$, $\{\tau_j, \tau_i, \tau_k\}$, $\{\tau_j, \tau_k, \tau_i\}$, $\{\tau_k, \tau_i, \tau_j\}$ and $\{\tau_k, \tau_j, \tau_i\}$. There are $3! = 6$ possibilities.

Therefore, for a subset $\mathcal{G}_x$ having n flows, there are $n!$ possible sum arrival curves and the *Safe Arrival Curve* is the piecewise max of all these curves. The computation time becomes prohibitive for large value of n. Thus, we propose that each time when we consider a frame arrival of τ_i at time $t = 0$, we consider the earliest frame arrival of any other flow τ_j in the subset arrives at time $MD^h_{i,j}$, and we ignore the minimum duration constraints between any other two flows. It means that the other flows in this set are considered independent to each other. Thus, for a subset $\mathcal{G}_x$ having n flows, there are n possible sum arrival curves. This simplified computation is pessimistic but it reduces the computation complexity.

More formally, for a subset $\mathcal{G}_x$ with n flows, n cases are considered. For the case where τ_i has a frame arrival at time $t = 0$, the arrival curve adds $\alpha^h_i(t)$ and other $\alpha^h_j(t - MD^h_{i,j})$

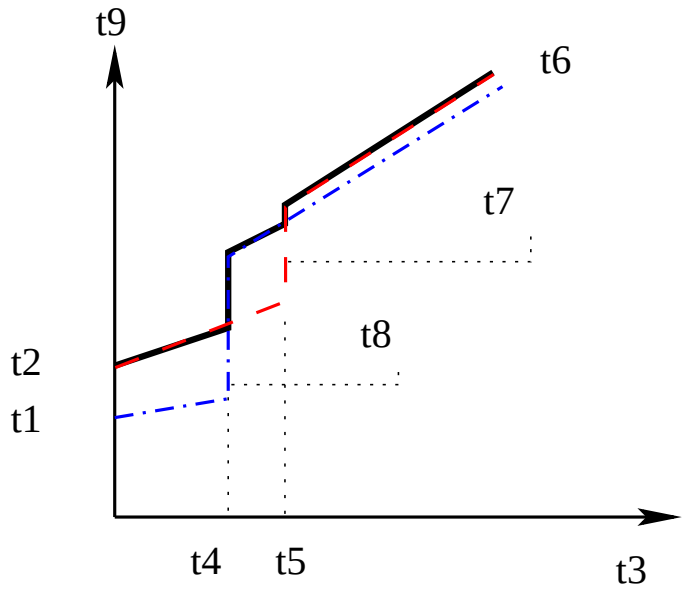


Figure 2.11: The *Safe Arrival Curve* of subset $\mathcal{G}_x$

$(j \in \mathcal{G}_x/\{i\})$ constrained by $MD^h_{i,j}$. It is given by:

$$\alpha^h_{\mathcal{G}_x,\tau_i}(t) = \alpha^h_i(t) + \sum_{j \in \mathcal{G}_x/\{i\}} \alpha^h_j(t - MD^h_{i,j})$$

Then the *Safe Arrival Curve* $\alpha^h_{\mathcal{G}_x}$ is computed by:

$$\alpha^h_{\mathcal{G}_x}(t) = \max_{i \in \mathcal{G}_x} \alpha^h_{\mathcal{G}_x,\tau_i}(t) \tag{2.2}$$

2.4 Application on a small network example

In the following paragraphs, the proposed approach is illustrated on the example in Figure 2.1. Based on the flow parameters in Table 2.1, the arrival curves of τ_1, τ_2, τ_3, τ_4 and τ_5 are:

$$\begin{aligned}
\alpha_1(t) &= 2t + 4000, \\
\alpha_2(t) &= t + 4000, \\
\alpha_3(t) &= t + 4000, \\
\alpha_4(t) &= 0.5t + 4000, \\
\alpha_5(t) &= 0.25t + 4000
\end{aligned}$$

They are shown in Figure 2.12.

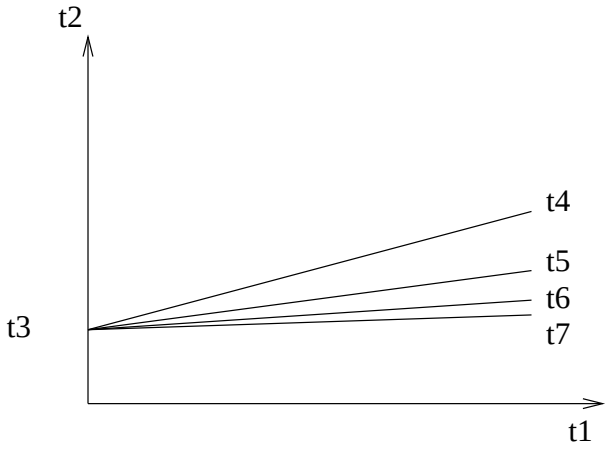


Figure 2.12: The arrival curves of flows

First we consider the classical Network Calculus approach. We illustrate the computation at the output port of S_1. Then the results of all the flows are given. Second, we do the same considering the modified Network Calculus approach.

2.4.1 Classical Network Calculus approach

In order to compute the worst-case delay at the output port of S_1, we first have to determine the arrival curves of flows τ_1, τ_2, τ_3 and τ_4 at S_1 (these flows compete at this port). It has been shown in Section 2.2.3 that:

$$\begin{aligned}
\alpha_1^{S_1}(t) &= 2t + 4080, \\
\alpha_2^{S_1}(t) &= t + 4040, \\
\alpha_3^{S_1}(t) &= t + 4040, \\
\alpha_4^{S_1}(t) &= 0.5t + 4020
\end{aligned}$$

Since frames are serialized on links $N_1 - S_1$ and $N_2 - S_1$, the overall arrival curve α^{S_1} is the sum of the arrival curve in Figure 2.6 and the curve in Figure 2.7, as illustrated in Figure 2.13. The output port of S_1 provides a service curve: $\beta^{S_1} = 100(t - 10)^+$. Then the maximum delay generated at the output port of S_1 is $h(\alpha^{S_1}, \beta^{S_1}) = 132.02\ \mu s$, which is illustrated in Figure 2.13.

Similarly, we can derive the arrival curve of the output port of S_2, and the maximum delay generated at the output port of S_2 is $h(\alpha^{S_2}, \beta^{S_2}) = 92.51\ \mu s$. For flow τ_1 crossing the output ports of N_1, S_1 and S_2, its worst-case end-to-end delay upper bound R_1 computed by the classical Network Calculus approach is the sum of the maximum delays generated at each

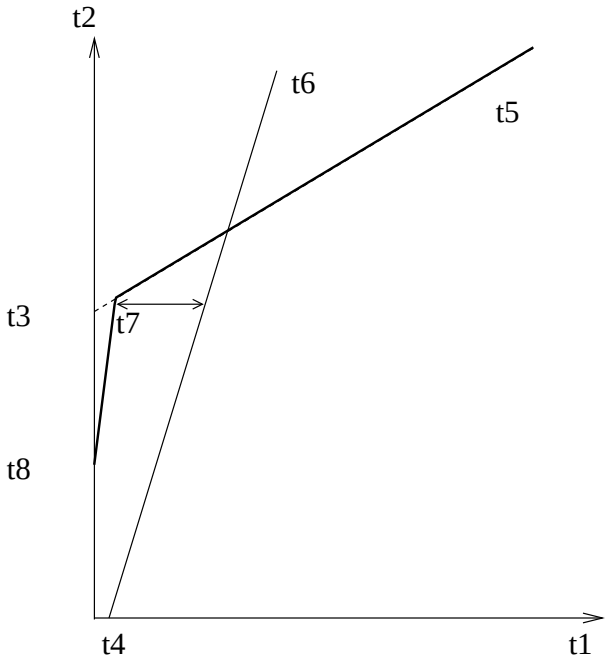


Figure 2.13: Maximum delay generated at S_1

crossed output port:

$$R_1 = 80 + 132.02 + 92.51 = 304.53\ \mu s$$

It is illustrated in Figure 2.14.

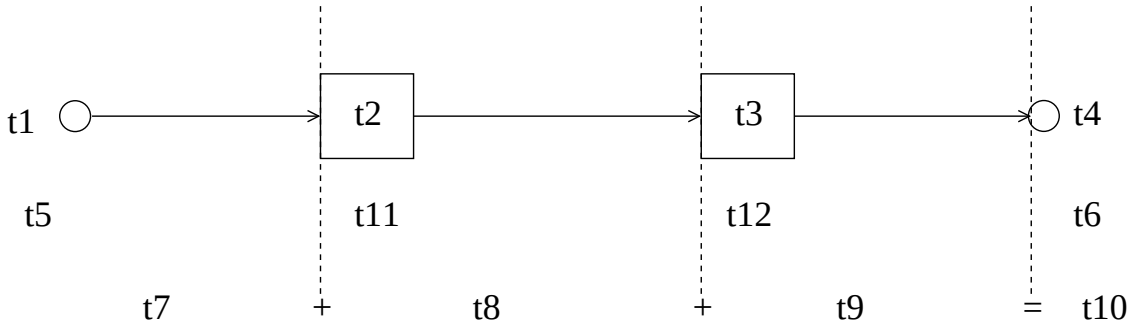


Figure 2.14: Worst-case delay of τ_1 with the classical Network Calculus approach

Results of all the flows of the example are given in Table 2.2 (column *NC*).

2.4.2 Modified Network Calculus approach

Two sets of flows are competing at the output port of S_1. $\mathcal{G}_1$ includes flows τ_1 and τ_2 which are generated by N_1, while $\mathcal{G}_2$ includes flows τ_3 and τ_4 which are generated by N_2. Indeed, these flows are periodic. In order to compute the aggregated arrival curves of these two sets at S_1, we have to determine the arrival curve of the flows they include. Due to the minimum duration constraints ($MD_{1,2}^{N_1} = 1500\ \mu s$ and $MD_{2,1}^{N_1} = 500\ \mu s$) at N_1, τ_1 and τ_2 do not suffer any delay jitter at N_1. Thus they have the same arrival curves as at N_1:

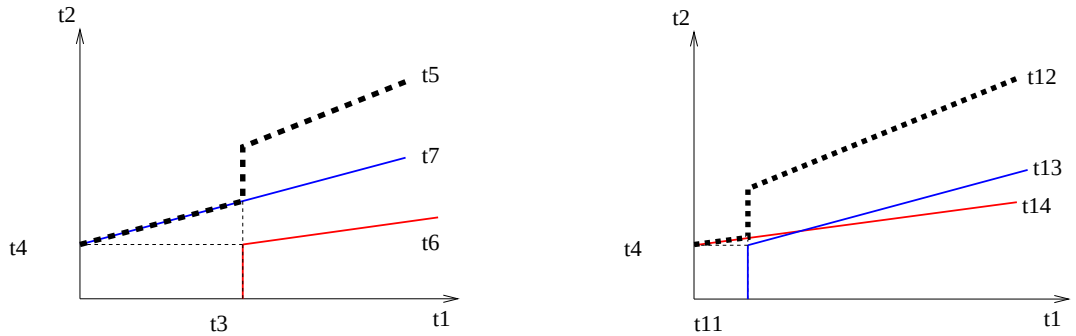
$$\begin{aligned} \alpha_1^{S_1}(t) &= 2t + 4000, \\ \alpha_2^{S_1}(t) &= t + 4000 \end{aligned}$$

It is the same for flows τ_3 and τ_4:

$$\begin{aligned} \alpha_3^{S_1}(t) &= t + 4000, \\ \alpha_4^{S_1}(t) &= 0.5t + 4000 \end{aligned}$$

The aggregated arrival curve of the set $\mathcal{G}_1 = \{1, 2\}$ is built from two sum arrival curves. The first one considers that a frame of τ_1 arrives at S_1 at time $t = 0$. Due to the minimum duration $MD_{1,2}^{S_1}$ from τ_1 to τ_2, the delayed arrival curve of τ_2 is $\alpha_2^{S_1}(t - MD_{1,2}^{S_1})$, which is depicted in Figure 2.15(a). The corresponding sum arrival curve $\alpha_{\mathcal{G}_1,\tau_1}^{S_1}$ is also shown in Figure 2.15(a).

The second one considers the scenario when a frame of τ_2 arrives at S_1 at time $t = 0$. Then another sum arrival curve $\alpha_{\mathcal{G}_1,\tau_2}^{S_1}$ of $\mathcal{G}_1$ is obtained by the sum of $\alpha_2^{S_1}$ and $\alpha_1^{S_1}(t - MD_{2,1}^{S_1})$ as depicted in Figure 2.15(b).



(a) $\alpha_{\mathcal{G}_1,\tau_1}^{S_1}$ (b) $\alpha_{\mathcal{G}_1,\tau_2}^{S_1}$

Figure 2.15: Sum arrival curves of $\mathcal{G}_1$

Finally the *Safe Arrival Curve* of $\mathcal{G}_1$, shown in Figure 2.16, is obtained by:

$$\alpha^{S_1}_{\mathcal{G}_1}(t) = \max(\alpha^{S_1}_{\mathcal{G}_1,\tau_1}(t), \alpha^{S_1}_{\mathcal{G}_1,\tau_2}(t))$$

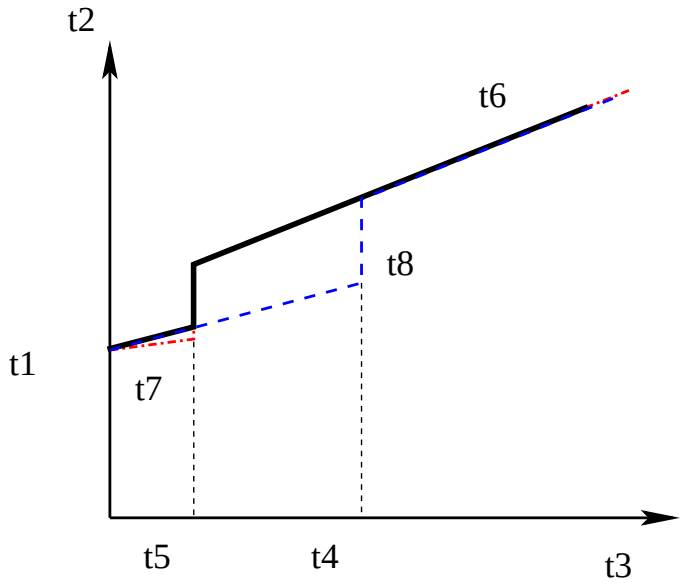


Figure 2.16: The *Safe Arrival Curve* of subset $\mathcal{G}_1$

Similarly, for the subset $\mathcal{G}_2 = \{3, 4\}$, the minimum duration constraints at S_1 are: $MD^{S_1}_{3,4} = 1000\ \mu s$ and $MD^{S_1}_{4,3} = 3000\ \mu s$. Their aggregated flow has also two sum arrival curves $\alpha^{S_1}_{\mathcal{G}_2,\tau_3}$ and $\alpha^{S_1}_{\mathcal{G}_2,\tau_4}$. The *Safe Arrival Curve* $\alpha^{S_1}_{\mathcal{G}_2}$ for $\mathcal{G}_2$ is obtained by taking the piecewise max of $\alpha^{S_1}_{\mathcal{G}_2,\tau_3}$ and $\alpha^{S_1}_{\mathcal{G}_2,\tau_4}$. It is shown in Figure 2.17.

Then by adding $\alpha^{S_1}_{\mathcal{G}_1}$ and $\alpha^{S_1}_{\mathcal{G}_2}$, the overall arrival curve α^{S_1} at the output port of S_1 is obtained:

$$\alpha^{S_1} = \alpha^{S_1}_{\mathcal{G}_1} + \alpha^{S_1}_{\mathcal{G}_2}$$

It is also depicted in Figure 2.17.

Figure 2.18 compares the overall arrival curve obtained by the sum of the *Safe Arrival Curves* in Figure 2.17 with the overall arrival curve considered by classical Network Calculus approach which does not take into account the minimum duration constraints.

α^{S_1} in Figure 2.18 represents the overall arrival curve at S_1 obtained by the modified Network Calculus approach, while $\alpha^{S_1'}$ is the overall arrival curve obtained by the classical Network Calculus approach. The upper bounded delay at this port is given by the maximum horizontal distance between the overall arrival curve and the service curve, which for α^{S_1} is:

$$h(\alpha^{S_1}, \beta^{S_1}) = 90.0\ \mu s$$

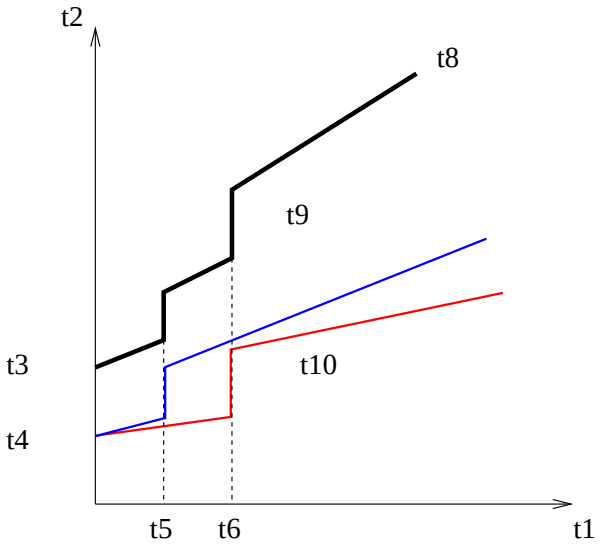


Figure 2.17: The overall arrival curve α^{S_1}

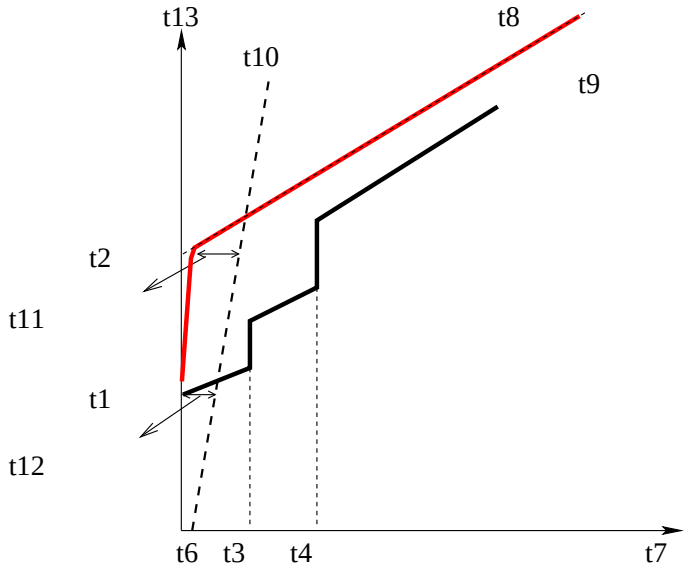


Figure 2.18: Comparison with the classical Network Calculus approach

As presented in Section 2.4.1, it is 132.02 μs for the classical Network Calculus approach. Thus the improvement is significant on this port for the considered example.

Similarly, we can derive the arrival curve α^{S_2} of flows τ_1, τ_2, τ_3, τ_4 and τ_5 at the output port of S_2. There are three subsets $\mathcal{G}_1 = \{1, 2\}$, $\mathcal{G}_2 = \{3, 4\}$ and $\mathcal{G}_3 = \{5\}$. The computation of $\alpha^{S_2}_{\mathcal{G}_1}$ and $\alpha^{S_2}_{\mathcal{G}_2}$ are similar to those at S_1, and therefore they are not detailed. Since the subset $\mathcal{G}_3$ has only one flow τ_5, $\alpha^{S_2}_{\mathcal{G}_3} = \alpha^{S_2}_5 = \alpha_5$.

Since the flows of τ_1, τ_2, τ_3 and τ_4 are transmitted from S_1 through one input link, their frame transmissions are serialized, and they cannot arrive at the output port of S_2 at the same time. These four flows are classified into two subsets, and then the sum of $\alpha^{S_2}_{\mathcal{G}_1}$ and $\alpha^{S_2}_{\mathcal{G}_2}$ are constrained by the frame serialization. Therefore the burst workload of the sum of the two arrival curves is the maximum burst workload among the four flows, i.e. $b^{S_2}_1 = 4100\ bits$, and the peak arriving rate is constrained by the link rate R. The constrained sum of arrival curves is obtained by:

$$\min(\alpha^{S_2}_{\mathcal{G}_1} + \alpha^{S_2}_{\mathcal{G}_2}, Rt + 4100)$$

Therefore the overall arrival curve α^{S_2} is obtained by:

$$\alpha^{S_2} = \min(\alpha^{S_2}_{\mathcal{G}_1} + \alpha^{S_2}_{\mathcal{G}_2}, Rt + 4100) + \alpha^{S_2}_5$$

The maximum delay generated at S_2 is computed by $h(\alpha^{S_2}, \beta^{S_2}) = 90.81\ \mu s$, as illustrated in Figure 2.19.

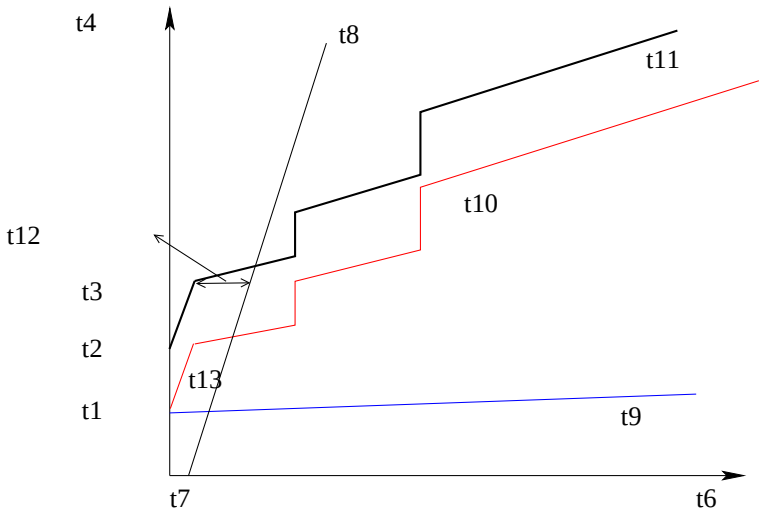


Figure 2.19: Illustration on the output port of S_2

From the maximum delays generated at the output ports of N_1, S_1 and S_2, we can compute the worst-case delay upper bound R_1 of τ_1:

$$R_1 = 40 + 90.0 + 90.81 = 220.81\ \mu s$$

It is illustrated in Figure 2.20.

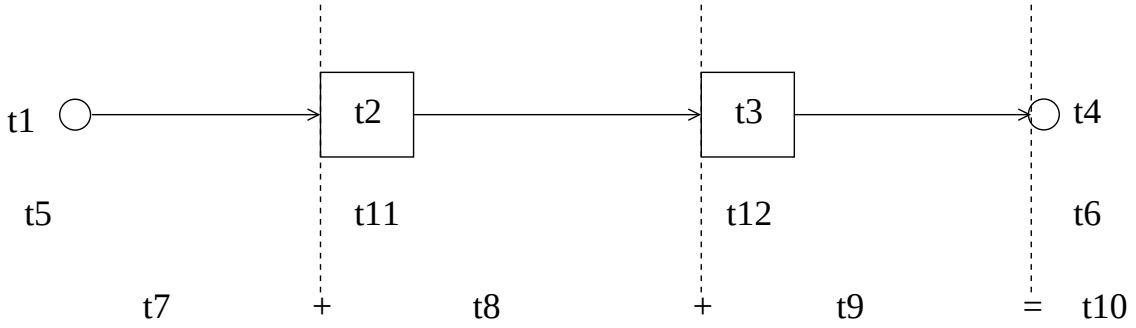


Figure 2.20: The illustration on R_1 computed by the modified Network Calculus approach

The results for all the flows of the configuration are given in Table 2.2 (column *ModifNC*). The column *Reduction* is the percentage of reduction when the results of *ModifNC* compared to that of *NC*. On this example, taking into account the minimum duration constraints brings a significant improvement on the delay upper bound (up to 27.5%). A more realistic case study will be presented in Chapter 4.

τ_i	NC (μs)	ModifNC (μs)	Reduction (%)
τ_1	304.53	220.81	27.5
τ_2	304.53	220.81	27.5
τ_3	304.53	220.81	27.5
τ_4	304.53	220.81	27.5
τ_5	132.50	130.81	1.3

Table 2.2: End-to-end delay upper bounds of the reference example

2.5 Conclusion

In this chapter, it has been illustrated that the classical Network Calculus approach leads to pessimistic computed delays without taking into account the constraints of minimum durations. Therefore, in this chapter a modified Network Calculus approach has been developed. This approach integrates the constraints of minimum duration at each output port along the considered path.

The proposed approach is illustrated on a small example of network. Each flow in the network example is computed and the results show that the modified Network Calculus approach brings significant improvements for four flows out of five on worst-case delay upper bounds compared to the classical Network Calculus approach. Further results of the modified Network

Calculus approach applied on an industrial avionics switched Ethernet network can be found in Chapter 4.

Chapter 3

Modified Trajectory approach integrating minimum duration constraints

Contents

3.1 Introduction 51
3.2 Classical Trajectory approach 52
3.3 A modified approach considering minimum duration constraints . 58
3.3.1 General idea 58
3.3.2 Computation overview 59
3.3.3 Computation of the maximum workload of dependent flows 59
3.4 Illustration on a small network example 65
3.4.1 Classical Trajectory approach 65
3.4.2 Modified Trajectory approach 66
3.5 Conclusion 70

3.1 Introduction

The Trajectory approach allows the computation of the worst-case delay of a flow transmitted on a switched Ethernet network. It has been first proposed for First In First Out (FIFO) scheduling in [MM06a], and expended for Fixed Priority (FP) scheduling in [MM06b]. This approach is not holistic since it considers the worst-case scenario that can happen to a frame along its trajectory. It has shown interesting properties of worst-case delay analysis on an avionics network as it can bring tighter delay upper bounds than the Network Calculus approach [BSF10].

The classical Trajectory approach does not consider the minimum duration constraints presented in Section 1.6. The goal of this chapter is to propose a modified Trajectory approach with the integration of these minimum duration constraints.

Section 3.2 summarizes the classical Trajectory approach applied on a real-time switched Ethernet network. Section 3.3 proposes a modified Trajectory approach which integrates the

minimum duration constraints. Section 3.4 applies the modified Trajectory approach to a network example. Section 3.5 concludes this chapter.

3.2 Classical Trajectory approach

This section gives a short overview of the approach. A more detailed presentation can be found in Appendix B.

The Trajectory approach has been proposed for the computation of worst-case response time in the context of distributed systems [MM06a]. It has been adapted and optimized for the computation of a worst-case delay on an avionics switched Ethernet network [BSF09, BSF10]. Both FIFO and FP schedulings have been considered. In this section, we focus on FIFO scheduling.

Let us consider flow τ_1 in the example in Figure 1.9. This example is recalled in Figure 3.1. The transmission rate is $R = 100\ Mbit/s$ and the switching latency is $sl = 10\ \mu s$.

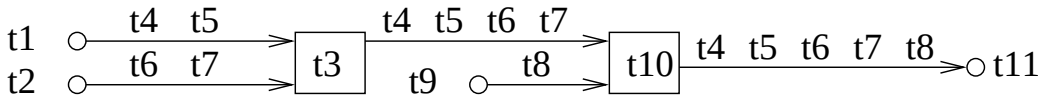


Figure 3.1: Recall of the network example for the Trajectory approach

Five flows are transmitted over the network example. There are two periodic flows τ_1 and τ_2 generated at node N_1, and two periodic flows τ_3 and τ_4 generated at node N_2. Flow τ_5 is an independent flow emitted by node N_3. They have parameters recalled in Table 3.1.

τ_i	$T_i(\mu s)$	$C_i(\mu s)$	$O_i(\mu s)$	$J_i(\mu s)$
τ_1	2000	40	0	0
τ_2	4000	40	3500	0
τ_3	4000	40	0	0
τ_4	8000	40	1000	0
τ_5	16000	40	0	0

Table 3.1: Recall of flow parameters of the network example for the Trajectory approach

Flow τ_1 follows path $\mathcal{P}_1 = \{N_1, S_1, S_2, N_4\}$. One possible scenario of one frame f_1 of flow τ_1 generated at time $t = 0$ is depicted in Figure 3.2. In this figure, $a^h_{f_1}$ is the arrival time of frame f_1 at the output port h.

Actually, the scenario in Figure 3.2 is the worst-case scenario for f_1. The delay of f_1 includes several parts (see Appendix B for details):

- The delay generated by frames competing with f_1 (the workload delay, blocks with oblique lines in Figure 3.2).
- The transition cost (one frame for each link, shadow blocks in Figure 3.2).

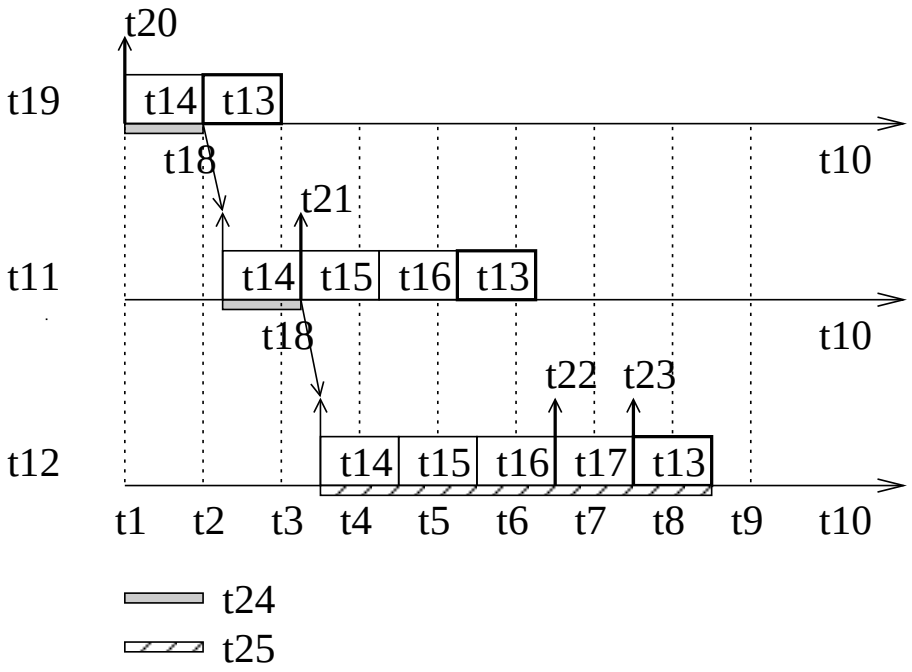


Figure 3.2: Worst-case scenario corresponding to the classical Trajectory approach

- The switching latency (oblique arrows in Figure 3.2).

The switching latency depends on the number of switches crossed by the frame and the latency in each switch. Since the latency in a switch is bounded by sl, the overall switching latency is:

$$(|\mathcal{P}_i| - 2) \cdot sl$$

The transition cost at one output port corresponds to the time needed to reach the following output port of the path. The largest competing frame is considered at each output port in order to avoid optimism in the computation. Thus the overall transition cost is:

$$\sum_{h \in \mathcal{P}_i / \{last_i, dest_i\}} \left(\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j) \right)$$

The output port $last_i$ does not have a following output port and $dest_i$ is the input port of the destination node.

In the example in Figure 3.2, f_2 is considered as the transition cost. Actually, any frame can be considered, since they all have the same frame size.

The last part of the delay (the workload delay) is more complex to determine. Indeed, this workload includes all the frames which can delay the frame under study. In order to do that,

we have to upper bound, for each flow τ_j competing with the flow under study, the number of frames which can delay the frame under study. This is achieved in two steps.

In the first step, the output port where flows τ_i and τ_j join is considered. This output port is denoted $first_{j,i}$. It has been shown in [MM06a] that a frame of τ_j can delay f_i only if it arrives at $first_{j,i}$ during the busy period when f_i is transmitted and no later than the arrival of f_i at $first_{j,i}$. Thus the earliest possible starting time of this busy period has to be determined. In [MM06a] an underestimation $M_i^{first_{j,i}}$ of this earliest possible starting time is proposed. It considers the earliest instant when a frame generated at time 0 at the source node of τ_i can reach the output port $first_{j,i}$. It is obtained by considering the transmission of the smallest possible frame at each output port between the source node of τ_i and $first_{j,i}$.

Figure 3.3 depicts the obtained time interval at $first_{j,i}$. It starts at $M_i^{first_{j,i}}$ and finishes at the latest possible arrival time of f_i at $first_{j,i}$, i.e. its generation time t plus its maximum delay $S_{max_i}^{first_{j,i}}$ between its source and $first_{j,i}$.

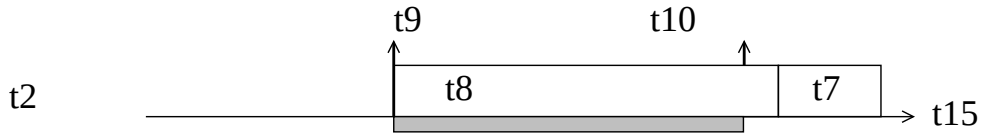


Figure 3.3: Illustration of the interval at $first_{j,i}$

In the second step, the maximum number of frames of τ_j which can join $first_{j,i}$ during this interval is determined. The delay of a frame of τ_j from its source node $first_j$ to $first_{j,i}$ is in the interval $[S_{min_j}^{first_{j,i}}, S_{max_j}^{first_{j,i}}]$. Thus a frame of τ_j has a chance to arrive at $first_{j,i}$ in the right interval only if it arrives at $first_j$ no earlier than $M_i^{first_{j,i}} - S_{max_j}^{first_{j,i}}$ and on later than $a_{f_i}^{first_{j,i}} - S_{min_j}^{first_{j,i}}$. These two limits on the arrival times of τ_j are indicated by upward arrows in Figure 3.4.

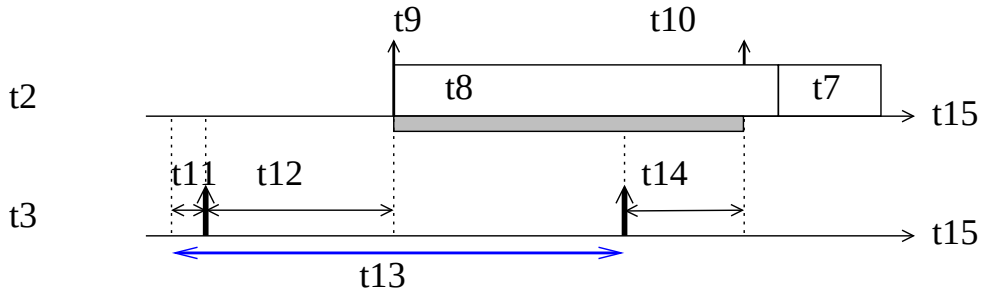


Figure 3.4: Illustration of the interval at $first_j$

Considering that the earliest possible frame arrival f_j can experience a maximum release jitter J_j, then the workload interval which maximizes the number of frame generations of τ_j is $t + A_{i,j}$ with:

$$A_{i,j} = S_{max_i}^{first_{j,i}} - S_{min_j}^{first_{j,i}} - M_i^{first_{j,i}} + S_{max_j}^{first_{j,i}} + J_j \tag{3.1}$$

Based on this workload interval for τ_j, the maximum number of frames which can delay f_i is:

$$(1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+$$

It leads to a maximum workload of:

$$(1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j$$

Thus the overall competing workload of n competing flows is:

$$\sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j$$

It has been shown in [BSF10] that, for some configurations, a part of this workload cannot delay f_i due to the serialization effect. This part is denoted $\Delta_{i,t}^h$ and its computation is detailed in Appendix B.3.

Now the whole computation of the Trajectory approach can be summarized. The latest starting time $W_{i,t}^{last_i}$ of frame f_i from a flow τ_i at its last visited output port $last_i$ is given by:

$$\begin{aligned}
W_{i,t}^{last_i} \quad = \quad & \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j && (3.2) \\
& + \sum_{h \in \mathcal{P}_i / \{last_i, dest_i\}} (\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j)) && (3.3) \\
& + (|\mathcal{P}_i| - 2) \cdot sl && (3.4) \\
& - \sum_{h \in \mathcal{P}_i / \{first_i, dest_i\}} (\Delta_{i,t}^h) && (3.5) \\
& - C_i && (3.6)
\end{aligned}$$

Term 3.6 means that the transmission time C_i of f_i at the last visited output port is subtracted because $W_{i,t}^{last_i}$ is the latest starting time.

The worst-case end-to-end delay upper bound of the flow τ_i is obtained by:

$$R_i = \max_{-J_i+\mathcal{B}_i \geq t \geq -J_i} \{W_{i,t}^{last_i} + C_i - t\} \tag{3.7}$$

where $\mathcal{B}_i = \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} \lceil \frac{\mathcal{B}_i}{T_j} \rceil \cdot C_j$.

Let us illustrate the whole computation process on a frame f_1 of τ_1 of the example in Figure 3.1. The switching latency is $2 * sl = 20\ \mu s$ since f_1 crosses two switches. The transition cost includes one frame of maximum size for the output ports of N_1 and S_1. Frame f_2 is taken in this example. Any competing frame can be chosen since they all have the same size. Thus the transition cost is:

$$C_2 + C_2 = 40 + 40 = 80\ \mu s$$

The computation of the workload delay considers all the flows competing with τ_1: τ_2, τ_3, τ_4, τ_5 and τ_1 itself. Figure 3.5 illustrates the computation of the workload generated by τ_3.

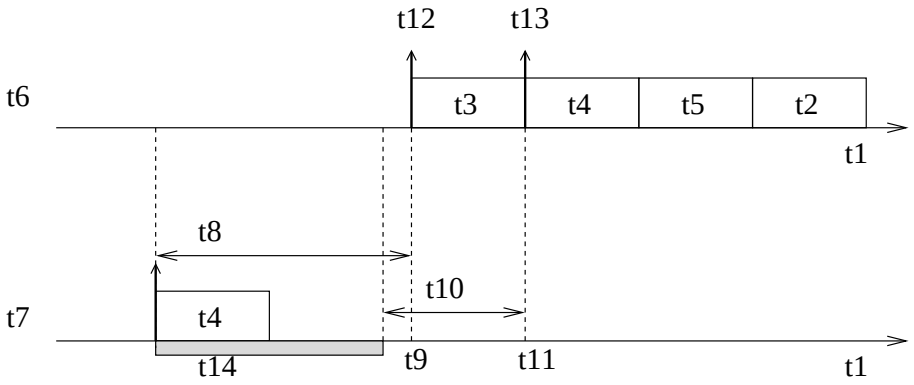


Figure 3.5: Illustration of the workload intervals of τ_3

The earliest starting of the busy period at S_1 is the time when the first frame (smallest one) transmitted from N_1 arrives. It is the sum of the transmission time of the smallest frame and a switching latency $sl = 10\ \mu s$. Since N_1 emits flows τ_1 and τ_2, the earliest starting time is $M_1^{S_1} = \min(C_1, C_2) + sl = 50\ \mu s$.

Frame f_1 is delayed by frame f_2 at the output port of N_1 and it experiences a switching latency $sl = 10\ \mu s$ when it crosses switch S_1. Thus, the maximum delay of f_1 from N_1 to S_1 is $S_{max_1}^{S_1} = C_1 + C_2 + sl = 90\ \mu s$. The latest arrival time of f_1 at the output port of S_1 is $a_{f_1}^{S_1} = t + S_{max_1}^{S_1}$.

Similarly, flow τ_3 can be delayed by flow τ_4 at the output port of N_2. Thus the maximum delay of flow τ_3 from N_2 to S_1 is $S_{max_3}^{S_1} = C_3 + C_4 + sl = 90\ \mu s$, while the minimum delay is $S_{min_3}^{S_1} = C_3 + sl = 50\ \mu s$.

Therefore, we have:

$$A_{1,3} = S^{S_1}_{max_1} - S^{S_1}_{min_3} - M^{S_1}_1 + S^{S_1}_{max_3} + J_3 = 90 - 50 - 50 + 90 + 0 = 80\ \mu s$$

Similarly, the workload intervals of the other flows are computed:

$$A_{1,1} = A_{1,2} = 0,\ A_{1,4} = 80\ \mu s,\ A_{1,5} = 120\ \mu s$$

Then the latest starting time $W^{S_2}_{1,t}$ of f_1 at its last visited output port S_2 is:

$$\begin{aligned} W^{S_2}_{1,t} &= \sum_{j\in[\![1,5]\!]} (1 + \lfloor \frac{t + A_{1,j}}{T_j} \rfloor)^+ \cdot C_j + C_2 + C_2 + 2 * sl - C_1 \\ &= (1 + \lfloor \frac{t}{2000} \rfloor)^+ \cdot 40 + (1 + \lfloor \frac{t}{4000} \rfloor)^+ \cdot 40 + (1 + \lfloor \frac{t+80}{4000} \rfloor)^+ \cdot 40 \\ &\quad + (1 + \lfloor \frac{t+80}{8000} \rfloor)^+ \cdot 40 + (1 + \lfloor \frac{t+120}{16000} \rfloor)^+ \cdot 40 + 60 \end{aligned}$$

Finally, the worst-case ETE delay of τ_1 is then bound by

$$R_1 = \max_{-J_1 + \mathcal{B}_1 \geq t \geq -J_1} (W^{S_2}_{1,t} + C_1 - t)$$

and it is obtained when $t = 0$, which is $R_1 = 300\ \mu s$ as illustrated in Figure 3.2. The results of other flows will be given in Table 3.3 (column *Traj*).

We illustrate the worst-case delay R_1 by showing the maximum vertical distance between $W^{S_2}_{1,t} + C_1$ and t, as depicted in Figure 3.6.

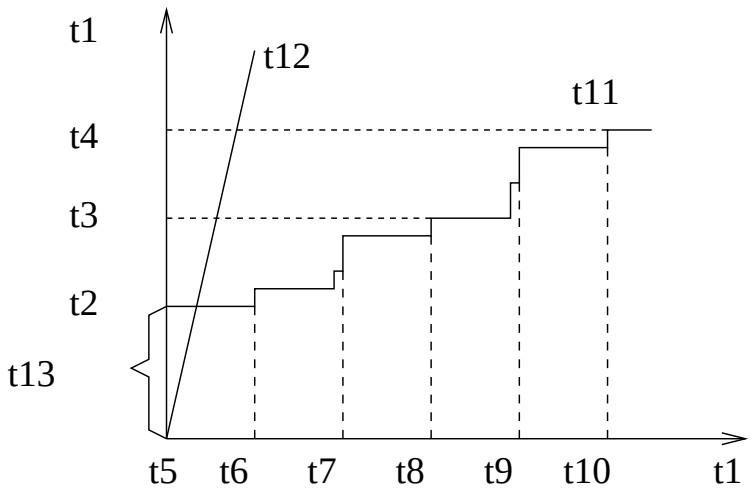


Figure 3.6: Illustration of the computation of R_1

The Trajectory approach does not consider the minimum duration constraints. It has been shown in Section 1.6 that for two periodic flows emitted by the same source node, there is a minimum duration between their frame arrivals. This constraint has been integrated in the Network Calculus approach in Chapter 2, and it improves the computation. In the following paragraphs, we show how the minimum duration constraints can be integrated in the Trajectory approach.

3.3 A modified approach considering minimum duration constraints

3.3.1 General idea

The classical Trajectory approach considers for the studied flow τ_i the maximum number of frames of each competing flow τ_j generated during the workload interval $t + A_{i,j}$. It leads to a worst-case scenario with each competing flow τ_j having a frame generation at the beginning of its workload interval $t+A_{i,j}$. For example, consider flows τ_3 and τ_4 in the example in Figure 3.1. These two flows delay flow τ_1 at the output port of S_1. In the previous section, it has been shown that for a frame f_1 of τ_1 generated at time $t = 0$, we have $M_1^{S_1} = 50\ \mu s$, $a_{f_1}^{S_1} = 90\ \mu s$, $S_{max_3} = S_{max_4} = 90\ \mu s$ and $S_{min_3} = S_{min_4} = 50\ \mu s$. Therefore $A_{1,3} = A_{1,4} = 80\ \mu s$. It is illustrated in Figure 3.7. It corresponds to the scenario when a frame of τ_3 and a frame τ_4 arrive at the output port of N_2 at the same time. However as shown in Section 1.6, there is a minimum duration from a frame of τ_3 to a frame of τ_4, which is 1000 μs and 3000 μs in the reverse order. Therefore, such a scenario will never happen.

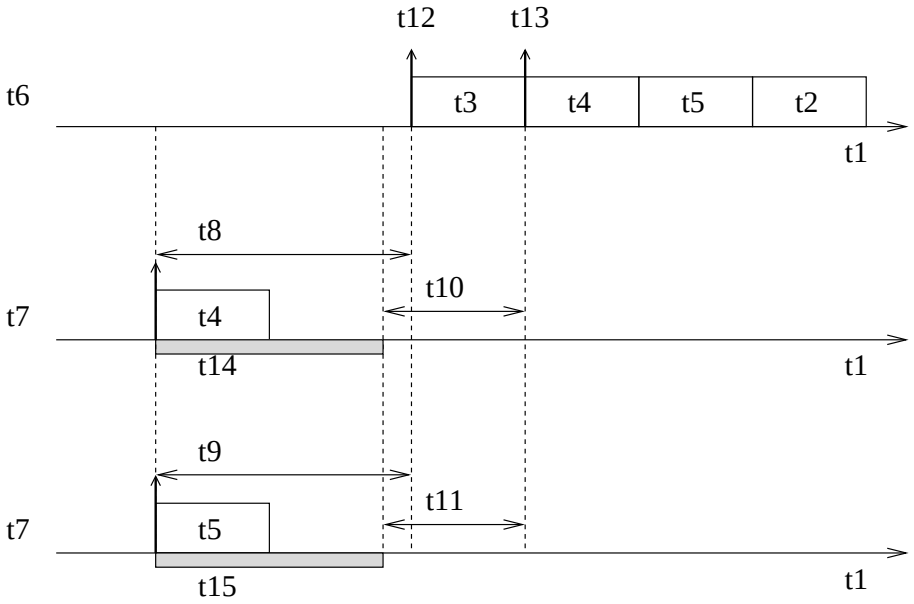


Figure 3.7: Illustration of the workload intervals of τ_3 and τ_4

Thus the idea is to integrate the minimum duration constraints in the workload intervals for all the dependent flows of each subset.

3.3.2 Computation overview

We consider a frame f_i of flow τ_i generated at time t. There are n flows crossing the path $\mathcal{P}_i$ of τ_i. The computation of the latest starting time $W_{i,t}^{last_i}$ of f_i at its last visited output port $last_i$ processes in the following steps:

1. The n flows are classified in n_g subsets $\mathcal{G}_x$, $x \in [\![1, n_g]\!]$. Any two flows in a given subset have minimum duration constraints between them. Conversely, any two flows in different subsets have no minimum duration constraints between them.
2. For the flows in a subset $\mathcal{G}_x$, we build all the combinations of their workload intervals which integrate the minimum duration constraints. When the subset $\mathcal{G}_x$ includes one single flow, the workload interval of the subset is obtained as in the classical Trajectory approach.
3. The maximum workload $RS_{i,t,x}$ for each subset $\mathcal{G}_x$ is obtained by the envelope of all the combinations of workload intervals.
4. The latest starting time $W_{i,t}^{last_i}$ of f_i at its last visited output port is:

$$\begin{aligned} W_{i,t}^{last_i} \quad = \quad & \sum_{x \in [\![1, n_g]\!]} RS_{i,t,x} \\ & + \sum_{h \in \mathcal{P}_i / \{last_i, dest_i\}} \left(\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j) \right) \\ & + (|\mathcal{P}_i| - 2) \cdot sl \\ & - \sum_{h \in \mathcal{P}_i / \{first_i, dest_i\}} (\Delta_{i,t}^h) \\ & - C_i \end{aligned} \tag{3.8}$$

5. The worst-case end-to-end delay upper bound R_i of flow τ_i is computed by Equation 3.7.

Thus, the difference with the approach presented in Section 3.2 is the computation of workload $RS_{i,t,x}$ for a set of dependent flows. This computation is detailed in the following sections.

3.3.3 Computation of the maximum workload of dependent flows

The computation is illustrated using the example in Figure 3.8. Flow τ_i is the only flow emitted by $first_i$, then it is the only flow in one subset. Flows τ_j and τ_k are periodic with known offsets

and emitted by their source node $first_j = first_k$. There is a minimum duration $MD_{j,k}^{first_k}$ from a frame arrival of τ_j to a frame arrival of τ_k and a minimum duration $MD_{k,j}^{first_k}$ in the reverse order. These two flows are classified in one subset.

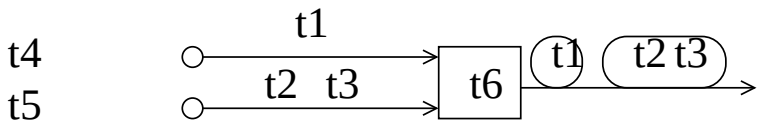


Figure 3.8: An illustrative example

We consider a frame f_i of flow τ_i generated at time t. Flows τ_j and τ_k delay flow τ_i at the output port h. Each subset $\mathcal{G}_x$ generates a maximum workload $RS_{i,t,x}$ to delay τ_i. This maximum workload has to be determined. For the example in Figure 3.8, a subset of two flows: $\mathcal{G}_x = \{j, k\}$ is considered.

The delay of a frame f_i of flow τ_i includes a workload delay. In the classical Trajectory approach the workload intervals of flows τ_j and τ_k are determined independently. Such intervals are shown in Figure 3.9.

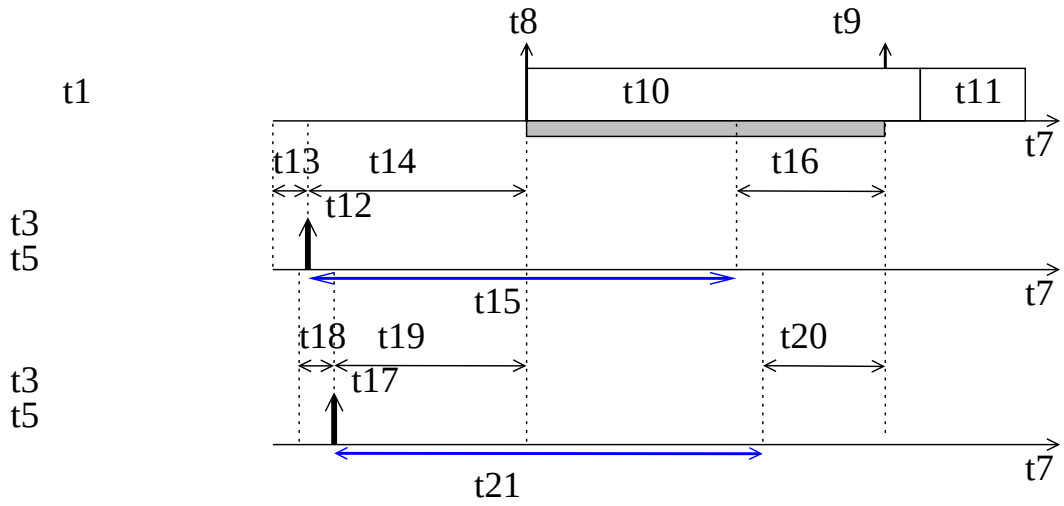


Figure 3.9: Illustration of the workload intervals $A_{i,j}$ and $A_{i,k}$

In order to take into account the minimum duration constraints, we consider two combinations of intervals. In the first combination, we consider that one frame f_j from τ_j is generated at the beginning of the workload interval of τ_j, and ready at time WS_j. Then no frames of flow τ_k can be ready before $WS_j + MD_{j,k}^{first_k}$. Indeed the minimum duration from one frame arrival of τ_j to one frame arrival of τ_k is $MD_{j,k}^{first_k}$. Thus in this case, the workload interval of τ_k cannot start before $WS_j + MD_{j,k}^{first_k}$. This combination of intervals is illustrated in Figure 3.10. The constrained workload interval of τ_k is denoted $A_{i,j,k}$.

The second combination is built in a similar manner, considering that a frame f_k of flow τ_k

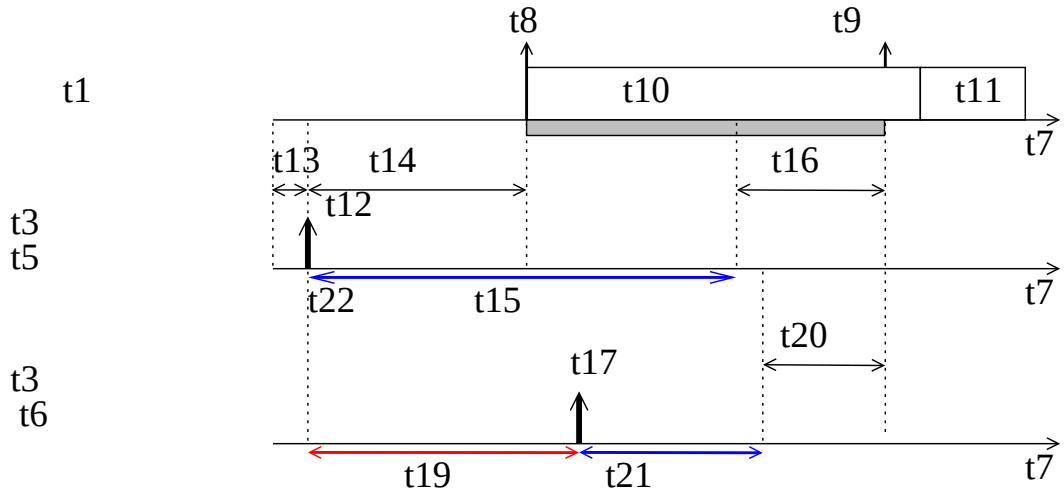


Figure 3.10: Illustration of the constrained workload intervals $A_{i,j,k}$

is generated at the beginning of its workload interval and ready at time WS_k. The the start of the workload interval of τ_j is delayed, as shown in Figure 3.11. The constrained workload interval of τ_j is denoted $A_{i,k,j}$.

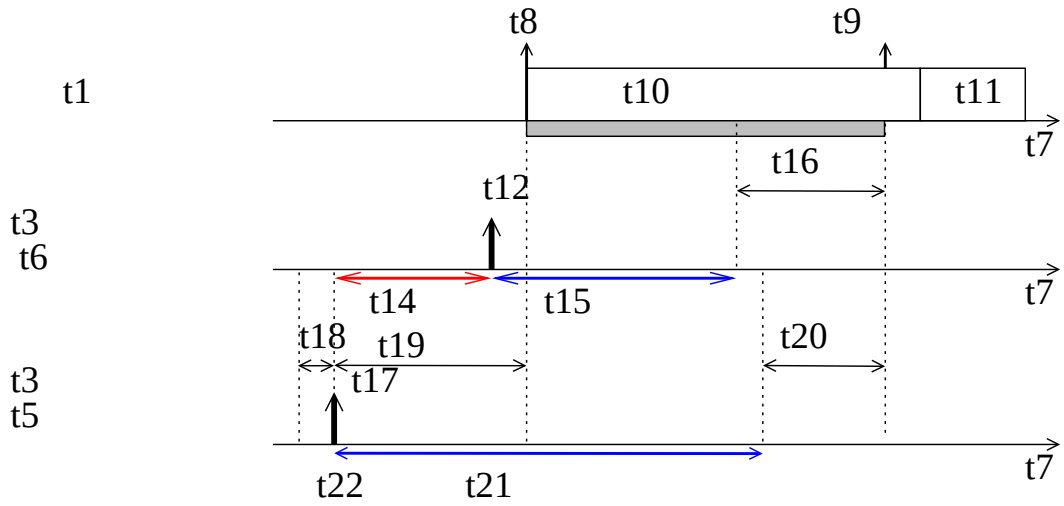


Figure 3.11: Illustration of the constrained workload intervals $A_{i,k,j}$

Taking into account these two combinations is sufficient for a subset of two flows, since it captures the worst-case workload of τ_j and τ_k. A scenario where a frame f_j of flow τ_j is not generated at the beginning of its workload interval cannot provide more workload. This scenario is illustrated in Figure 3.12. In this scenario, the release jitter J_j is null. The starting time of the workload interval $t+A_{i,j}$ is WS_j. The first frame f_j of flow τ_j arrives at its workload interval at time $WS_j + I_j$. Due to the minimum duration $MD_{j,k}^{first_k}$, the constrained workload

interval $t + A_{i,j,k}$ of flow τ_k starts at time $WS_j + MD_{j,k}^{first_k}$, and the first frame f_k of flow τ_k arrives at time $WS_j + I_j + MD_{j,k}^{first_k}$.

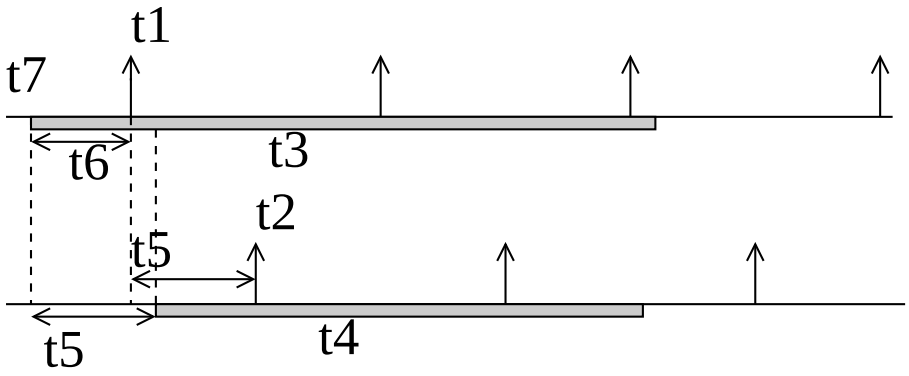


Figure 3.12: Illustration of a possible scenario of workload interval

Now we shift all the frame arrivals of flows τ_j and τ_k during their corresponding workload intervals to the left by temporal distance I_j. Thus the first frame arrival f_j of flow τ_j is at time WS_j, which is the beginning of its workload interval. The first frame f_k of flow τ_k is moved to time $WS_j + MD_{j,k}^{first_k}$, which is the beginning of its constrained workload interval. This scenario is illustrated in Figure 3.13. There is no frame moved out of its corresponding workload interval, thus the overall workload of flows τ_j and τ_k is not reduced. Therefore, a scenario where a frame is not generated at the beginning of its workload interval will not bring more workload than the scenario where a frame is generated at the beginning of its workload interval.

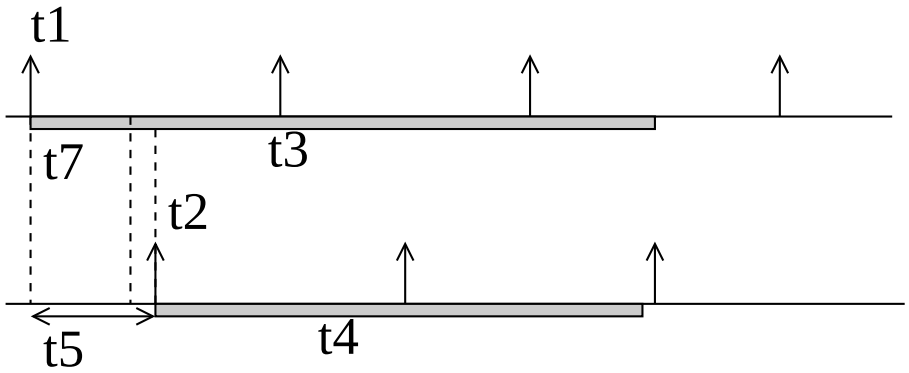


Figure 3.13: Illustration of the frame shift

In that case, for two flows τ_j and τ_k in one subset, the scenarios in Figure 3.10 and Figure 3.11 include the worst-case scenario.

Now let us formalize the computation. First let us consider the case where τ_j has a frame generated at the beginning of its workload interval $t + A_{i,j}$. Thus, the workload generated by

τ_j is computed by:

$$(1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j$$

As illustrated in Figure 3.10, since there exists a minimum duration $MD_{j,k}^{first_k}$, the workload interval $t + A_{i,j,k}$ is lower bounded by:

$$M_i^{first_{k,i}} - S_{max_j}^{first_{k,i}} + MD_{j,k}^{first_k}$$

We do not know if the modified lower bound is lower than the classical one $M_i^{first_{k,i}} - S_{max_k}^{first_{k,i}} - J_k$ (without minimum duration constraint). In order to avoid pessimism, the larger value is taken between them. Then the lower bound of the constrained workload interval $t + A_{i,j,k}$ is:

$$\max(M_i^{first_{k,i}} - S_{max_j}^{first_{k,i}} + MD_{j,k}^{first_k}, M_i^{first_{k,i}} - S_{max_k}^{first_{k,i}} - J_k)$$

The upper bound of the constrained workload interval is not related to $MD_{j,k}^{first_k}$. It is the same as the one in the classical workload interval, which is:

$$t + S_{max_i}^{first_{k,i}} - S_{min_k}^{first_{k,i}}$$

Therefore this constrained workload interval $t + A_{i,j,k}$ of τ_k is determined with:

$$A_{i,j,k} = S_{max_i}^{first_{k,i}} - S_{min_k}^{first_{k,i}} - \max(M_i^{first_{k,i}} - S_{max_j}^{first_{k,i}} + MD_{j,k}^{first_k}, M_i^{first_{k,i}} - S_{max_k}^{first_{k,i}} - J_k) \quad (3.9)$$

With the constrained workload interval, the maximum workload of flow τ_k to delay τ_i is computed by:

$$(1 + \lfloor \frac{t + A_{i,j,k}}{T_k} \rfloor)^+ \cdot C_k$$

Thus, the maximum workload of one combination of the subset $\mathcal{G}_x$ is the sum of the the maximum workloads of τ_j and τ_k. It is denoted by $RS_{i,t,x}^j$ which indicates the scenario where flow τ_j has a frame generated at the beginning of its workload interval $t + A_{i,j}$, and flow τ_k has a constrained workload interval $t + A_{i,j,k}$. Thus $RS_{i,t,x}^j$ is computed by:

$$RS_{i,t,x}^j = (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j + (1 + \lfloor \frac{t + A_{i,j,k}}{T_k} \rfloor)^+ \cdot C_k$$

There is another scenario where flow τ_k has a frame generated at the beginning of its

workload interval $A_{i,k}$. This scenario is illustrated in Figure 3.11. Similarly, for flow τ_j there is a constrained workload interval $t + A_{i,k,j}$ where:

$$A_{i,k,j} = S_{max_i}^{first_{j,i}} - S_{min_j}^{first_{j,i}} - \max(M_i^{first_{j,i}} - S_{max_k}^{first_{j,i}} + MD_{k,j}^{first_j}, M_i^{first_{j,i}} - S_{max_j}^{first_{j,i}} - J_j)$$

With the constrained workload interval, the maximum delay of flow τ_j to delay τ_i is computed by:

$$(1 + \lfloor \frac{t + A_{i,k,j}}{T_j} \rfloor)^+ \cdot C_j$$

Correspondingly, accounting for the scenario where τ_k has a frame generated at the beginning of its workload interval $t + A_{i,k}$, and flow τ_j has a constrained workload interval $t + A_{i,k,j}$, the maximum workload $RS_{i,t,x}^k$ of the subset $\mathcal{G}_x$ is obtained by:

$$RS_{i,t,x}^k = (1 + \lfloor \frac{t + A_{i,k}}{T_k} \rfloor)^+ \cdot C_k + (1 + \lfloor \frac{t + A_{i,k,j}}{T_j} \rfloor)^+ \cdot C_j$$

As illustrated before, there are two possible maximum workloads ($RS_{i,t,x}^j$ and $RS_{i,t,x}^k$) generated by the subset $\mathcal{G}_x$ corresponding to two scenarios of combinations. The maximum workload generated by the subset $\mathcal{G}_x$ is denoted $RS_{i,t,x}$. It considers the envelope of the two scenarios of combinations above. It is then obtained by:

$$RS_{i,t,x} = \max(RS_{i,t,x}^j, RS_{i,t,x}^k)$$

When there are more than two competing flows in one subset $\mathcal{G}_x$, each time one case where τ_j, $j \in \mathcal{G}_x$ has a frame at the beginning of its workload interval is considered. As we have shown in Section 2.3.3, in order to avoid the problem of prohibitive computation time, only the minimum duration constraints between flow τ_j and any other flow τ_k in $\mathcal{G}_x/\{j\}$ are considered. The minimum duration constraints between any other two flows in the subset are ignored. Then for a subset $\mathcal{G}_x$ having n flows, the computation goes through n times.

Thus the maximum workload $RS_{i,t,x}^j$ for one scenario of combination where flow τ_j has a frame generated at the beginning of its workload interval is computed by:

$$RS_{i,t,x}^j = (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j + \sum_{k \in \mathcal{G}_x/\{j\}} ((1 + \lfloor \frac{t + A_{i,j,k}}{T_k} \rfloor)^+ \cdot C_k)$$

And the maximum workload of the subset $\mathcal{G}_x$ is obtained by taking the maximum values of all the combinations:

$$RS_{i,t,x} = \max_{j \in \mathcal{G}_x}(RS^j_{i,t,x}) \tag{3.10}$$

3.4 Illustration on a small network example

The approach proposed in Section 3.3 is illustrated on the same example we used in Chapter 2. This example is depicted in Figure 3.1.

Flow τ_1 which follows the path $\mathcal{P}_1 = \{N_1, S_1, S_2, N_4\}$ is under study. Based on the configuration of the example, there are three subsets which are $\mathcal{G}_1 = \{1, 2\}$, $\mathcal{G}_2 = \{3, 4\}$ and $\mathcal{G}_3 = \{5\}$. The minimum durations between any couple of dependent flows have been computed in Section 1.6 and recalled in Table 3.2.

$MD^{N_1}_{1,2}$	$MD^{N_1}_{2,1}$	$MD^{N_2}_{3,4}$	$MD^{N_2}_{4,3}$
1500 μs	500 μs	1000 μs	3000 μs

Table 3.2: Minimum durations of flows τ_1, τ_2, τ_3 and τ_4 at their source nodes

3.4.1 Classical Trajectory approach

For the purpose of comparison, the classical Trajectory approach is first applied to the example. The computation has been detailed in Section 3.2. And the worst-case ETE delay of τ_1 is then bounded by $R_1 = 300$ μs, which is recalled in Figure 3.14.

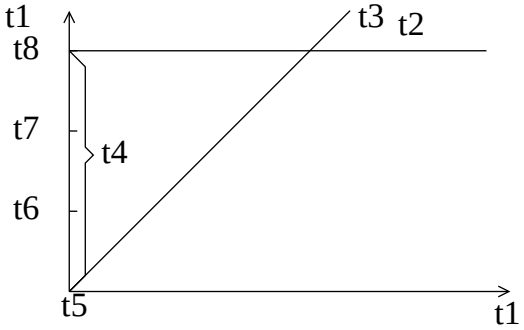


Figure 3.14: Classical approach

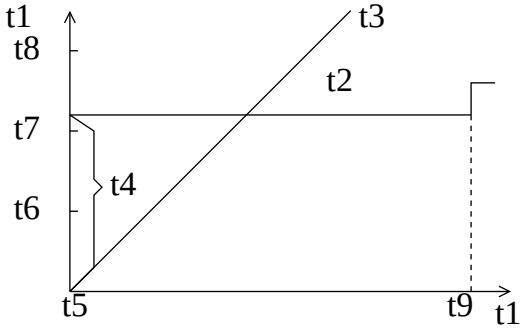


Figure 3.15: Modified approach

The classical Trajectory approach does not consider the minimum duration constraints listed in Table 3.2, and therefore it considers that a frame f_1 is delayed by a frame f_2 of τ_2 at N_1, by a frame f_3 of τ_3 and a frame f_4 of τ_4 at S_1 and by a frame f_5 of τ_5 at S_2. This corresponds to the worst case depicted in Figure 3.2.

3.4.2 Modified Trajectory approach

Now the modified Trajectory approach is applied to the example with the integrations of the minimum duration constraints. The delay generated by each subset is computed. Let us take the subset $\mathcal{G}_2 = \{3, 4\}$ as an illustration. Since there are two flows in this set, two scenarios are considered. One scenario is when flow τ_3 has a frame generation at the beginning of its workload interval $t + A_{1,3}$. The other scenario is when flow τ_4 has a frame generation at the beginning of its workload interval $t + A_{1,4}$.

Due to the minimum duration constraints ($MD_{3,4}^{N_2} = 1000\ \mu s$ and $MD_{4,3}^{N_2} = 3000\ \mu s$) at N_2, flows τ_3 and τ_4 do not suffer any delay jitter at N_2. They cross the switch S_1 and experience a switching latency $sl = 10\ \mu s$. Thus, we have:

$$\begin{aligned} S_{max_3}^{S_1} &= S_{min_3}^{S_1} = 50\ \mu s, \\ S_{max_4}^{S_1} &= S_{min_4}^{S_1} = 50\ \mu s \end{aligned}$$

The earliest starting time of the busy period at the output port of S_1 is the time when the first frame (smallest one) transmitted from N_1 arrives. The source node N_1 emits two flows τ_1 and τ_2 to the output port of S_1. This frame suffers a minimum transmission delay of $\min(C_1, C_2) = 40\ \mu s$ and a switching latency of 10 μs. Thus $M_1^{S_1} = 50\ \mu s$.

Due to the minimum duration $MD_{1,2}^{N_1} = 1500\ \mu s$, frame f_1 of flow τ_1 is not delayed by a frame f_2 of flow τ_2. It experiences a switching latency of $10\mu s$ when it crosses the switch S_1. Thus its maximum delay from N_1 to S_1 is $S_{max_1}^{S_1} = 50\ \mu s$. Its latest arrival time at the output port of S_1 is $a_{f_1}^{S_1} = t + S_{max_1}^{S_1} = t + 50\ \mu s$.

Let us first consider the scenario where τ_3 has a frame generation at the beginning of its workload interval. The workload interval of flow τ_3 is computed by:

$$\begin{aligned} A_{1,3} &= S_{max_1}^{S_1} - S_{min_3}^{S_1} - M_1^{S_1} + S_{max_3}^{S_1} + J_3 \\ &= 50 - 50 - 50 + 50 + 0 \\ &= 0 \end{aligned}$$

For τ_4 which is constrained by $MD_{3,4}^{N_2}$, its workload interval is $t + A_{1,3,4}$ with:

$$\begin{aligned} A_{1,3,4} &= S_{max_1}^{S_1} - S_{min_4}^{S_1} - \max(M_1^{S_1} - S_{max_3}^{S_1} + MD_{3,4}^{N_2}, M_1^{S_1} - S_{max_4}^{S_1} - J_4) \\ &= 50 - 50 - 50 + 50 - \max(1000, -40) \\ &= -1000 \end{aligned}$$

Then the maximum workload generated by the subset $\mathcal{G}_2$ when τ_3 has a frame generation

at the beginning of its workload interval is:

$$\begin{aligned} RS^3_{1,t,2} &= (1+\lfloor\frac{t+A_{1,3}}{T_3}\rfloor)^+ \cdot C_3 + (1+\lfloor\frac{t+A_{1,3,4}}{T_4}\rfloor)^+ \cdot C_4 \\ &= (1+\lfloor\frac{t}{4000}\rfloor)^+ \cdot 40 + (1+\lfloor\frac{t-1000}{8000}\rfloor)^+ \cdot 40 \end{aligned}$$

It is illustrated in Figure 3.16.

Similarly, when τ_4 has a frame generation at the beginning of its jitter, τ_3 has a constrained workload interval $t + A_{1,4,3} = t - 3000$, and the maximum workload generated by the subset $\mathcal{G}_2$ in this scenario is:

$$\begin{aligned} RS^4_{1,t,2} &= (1+\lfloor\frac{t+A_{1,4}}{T_4}\rfloor)^+ \cdot C_4 + (1+\lfloor\frac{t+A_{1,4,3}}{T_3}\rfloor)^+ \cdot C_3 \\ &= (1+\lfloor\frac{t}{8000}\rfloor)^+ \cdot 40 + (1+\lfloor\frac{t-3000}{4000}\rfloor)^+ \cdot 40 \end{aligned}$$

It is illustrated in Figure 3.16.

In order to guarantee the worst case, the maximum workload generated by the subset $\mathcal{G}_2$ is determined by:

$$RS_{1,t,2} = \max(RS^3_{1,t,2}, RS^4_{1,t,2})$$

It is illustrated in Figure 3.16.

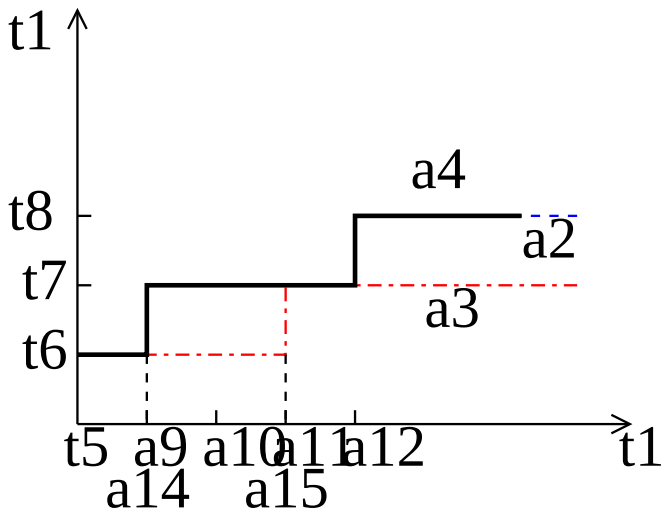


Figure 3.16: Illustration on maximum workload curve $RS_{1,t,2}$

In a similar process, the maximum workload generated by the subset $\mathcal{G}_1$ is determined by:

$$RS_{1,t,1} = \max(RS^1_{1,t,1}, RS^2_{1,t,1})$$

It is illustrated in Figure 3.17.

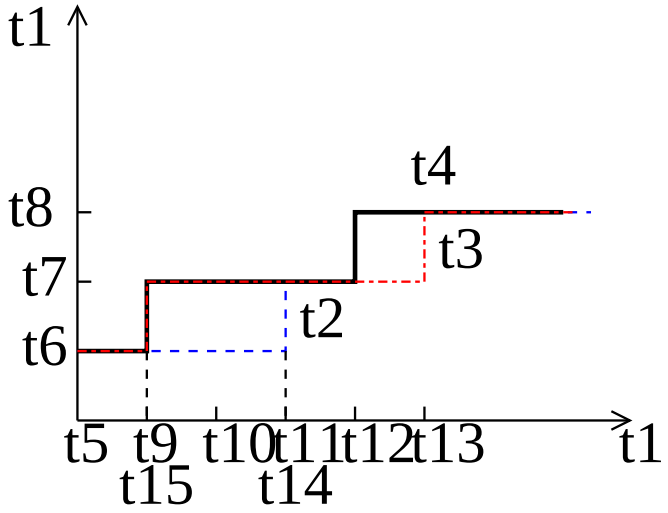


Figure 3.17: Illustration on maximum workload curve $RS_{1,t,1}$

The subset $\mathcal{G}_3$ has one single flow τ_5. Thus the maximum workload generated by this set is obtained by:

$$\begin{aligned} RS_{1,t,3} &= (1 + \lfloor \frac{t + A_{1,5}}{T_5} \rfloor)^+ \cdot C_5 \\ &= (1 + \lfloor \frac{t + 40}{16000} \rfloor)^+ \cdot 40 \end{aligned}$$

It is illustrated in Figure 3.18.

Since all the flows have the same frame size, any frame can be chosen as the transition cost at the output ports of N_1 and S_1. In the example frame f_1 is taken, thus the transition cost is:

$$C_1 + C_1 = 40 + 40 = 80\ \mu s$$

Since frame f_1 crosses two switches, the switching latency is $2 * sl = 20\ \mu s$.

Hence, based on the modified Trajectory approach of Equation 3.8, the latest starting time of frame f_1 of τ_1 at its last visited output port of S_2 is:

$$\begin{aligned} W^{S_2}_{1,t} &= RS_{1,t,1} + RS_{1,t,2} + RS_{1,t,3} + C_1 + C_1 - 2 * sl - C_1 \\ &= RS_{1,t,1} + RS_{1,t,2} + RS_{1,t,3} + 60 \end{aligned}$$

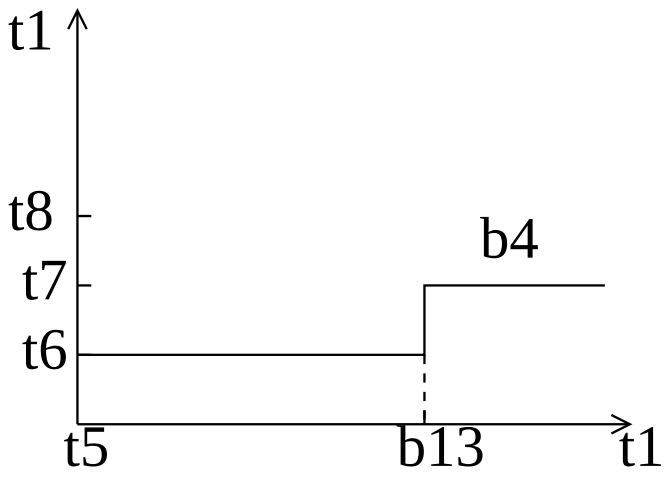


Figure 3.18: Illustration on maximum workload curve $RS_{1,t,3}$

The worst-case end-to-end delay upper bound of τ_1 is then bounded by Equation 3.7 and equal to $R_1 = 220\ \mu s$ when $t = 0$. We illustrate the delay by considering the maximum vertical distance from curve $W_{1,t}^{S_2} + C_1$ to curve t, which is shown in Figure 3.15. The corresponding worst-case scenario is shown in Figure 3.19.

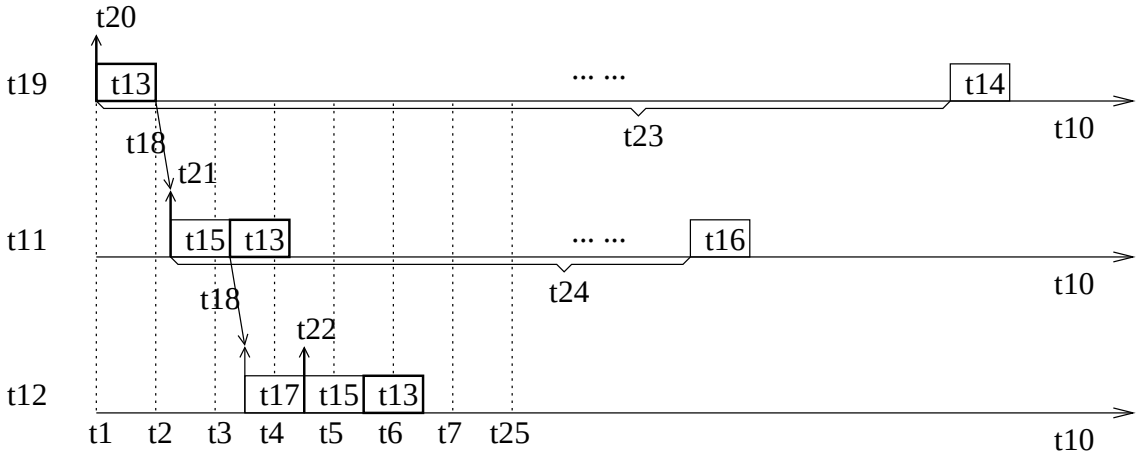


Figure 3.19: Worst-case scenario corresponding to the modified Trajectory approach

Compared to the computed result obtained by the classical Trajectory approach, there is a clear reduction of R_1 from 300 μs to 220 μs. The difference 80 μs between two computed results are contributed by the minimum duration constraints. Actually, due to the existing $MD_{1,2}^{N_1}$, f_1 is not delayed by f_2 at N_1. Similarly, due to the existing $MD_{3,4}^{N_2}$ ($MD_{4,3}^{N_2}$), at the output port of S_1 only one frame (lager one) of f_3 and f_4 can arrive during the busy period bp^{S_1} and delay f_1. f_3 is taken as an illustration in Figure 3.19. A different of 80 μs is the transmission time of f_2 and f_4 which do not delay f_1 but are taken into account in the classical approach.

The results for all the flows of the configuration are given in Table 3.3 (column *ModifTraj*). The column *Reduction* is the percentage of reduction when the results of *ModifTraj* compared to that of *Traj*. On this example, taking into account the minimum duration constraints brings a significant improvement on the delay upper bound (up to 26.7%). A more realistic case study will be presented in Chapter 4.

τ_i	Traj (μs)	ModifTraj (μs)	Reduction (%)
τ_1	300	220	26.7
τ_2	300	220	26.7
τ_3	300	220	26.7
τ_4	300	220	26.7
τ_5	130	130	0

Table 3.3: End-to-end delay upper bounds of the reference example

3.5 Conclusion

In this chapter, it has been illustrated that the classical Trajectory approach, which does not consider the constraints of minimum duration, introduces pessimistic computation of end-to-end delay upper bounds. Indeed, there exist minimum durations between two frames of two periodic flows emitted by the same source node. Therefore, a modified Trajectory approach has been developed which integrates the constraints of minimum duration in this chapter.

The proposed approach is applied to a network example. The results show that the modified Trajectory approach brings improvement (up to 26.7%) on worst-case delay upper bounds compared to the classical Trajectory approach. Further results of the modified Trajectory approach applied on an industrial avionics switched Ethernet network can be found in Chapter 4. Moreover, the comparison of the two modified approaches, the Network Calculus approach and the Trajectory approach, will be conducted on this industrial avionics switched Ethernet network in Chapter 4.

Chapter 4

A real-time switched Ethernet example: the AFDX network

Contents

4.1 Introduction **71**
4.2 AFDX context **72**
4.3 Example of an offset assignment algorithm **74**
4.4 Worst-case delay analysis based on the modified Network Calculus approach **75**
4.5 Worst-case delay analysis based on the modified Trajectory approach **78**
4.6 A comparison of the two modified approaches **81**
4.7 Near-optimal offset assignment for the industrial AFDX network **82**
4.7.1 Existing offset assignments 83
4.7.2 Optimal scenario of dependent flows over the AFDX network 84
4.7.3 Offset assignment algorithms in the context of AFDX network 86
4.7.4 Results 92
4.8 Conclusion **94**

4.1 Introduction

In previous chapters, two approaches, a modified Network Calculus approach and a modified Trajectory approach, have been proposed to upper bound the end-to-end delays of a real-time switched Ethernet network with dependent flows. The evaluation of these two modified approaches on an industrial application is important in order to demonstrate the improvement brought by considering the minimum duration constraints. In this chapter, an industrial real-time switched Ethernet (Avionic Full DupleX Switched Ethernet, AFDX [ACC08]) is presented. AFDX has been proposed to satisfy the growing requirements of avionics application and developed for modern aircraft such as Airbus A380. This chapter concentrates on the evaluations on an industrial AFDX configuration of both modified approaches presented in the previous two chapters.

Different offset assignments of flows can lead to different combinations of flow arrival times at one source node. Therefore, it is interesting to find out which offset assignment leads to

tighter delay upper bounds of a given configuration. This issue has been addressed in the context of uniprocessor in [Goo03, GGN06] as well as in the context of automotive network in [GHN08]. Therefore, it is interesting to consider existing offset assignment methods to find an algorithm that minimizes the delay upper bounds of the industrial AFDX network. Moreover, the existing algorithms can be adapted in order to take into account specific characteristics of an AFDX network.

In this chapter, we evaluate offset assignment algorithms for industrial AFDX network. The goal of the evaluation is to measure the gap between offset assignments based on heuristics and the optimal assignment. For a given flow, this gap can be defined as the difference between the worst-case ETE delays obtained by, on one hand, the *Optimal offset assignment*, and on the other hand, the offset assignment based on a heuristic. Since the optimal scenario is intractable on the industrial AFDX network, it is impossible to obtain the exact gap between the *Optimal offset assignment* and each offset assignment heuristic. In order to evaluate it, one way is to upper bound the gap from an underestimated delay (a reachable value) to an overestimated delay (a worst-case delay upper bound). An underestimated delay is achieved by an optimal scenario.

This chapter is organized as follows. Section 4.2 introduces the AFDX network model and an illustrative industrial configuration. Section 4.3 presents an offset assignment example for the purpose of evaluation. Section 4.4 shows the evaluation on the AFDX industrial configuration with the modified Network Calculus approach. Section 4.5 shows the evaluation on the AFDX industrial configuration with the modified Trajectory approach. Section 4.6 compares the results obtained by the two modified approaches. Section 4.7 introduces the existing offset assignments and proposes new heuristics integrating the avionics characteristics. The offset assignments are compared to an optimal assignment for the AFDX industrial configuration. Section 4.8 concludes this chapter.

4.2 AFDX context

Avionics Full DupleX Switched Ethernet (AFDX) [ACC08] is a switched Ethernet network which has been defined for the avionics context and developed for modern aircraft such as Airbus A380. AFDX is one of the industrial applications of real-time switched Ethernet networks, since this network is based on full-duplex switched Ethernet architecture. In this architecture, a static routing at each switch is implemented and flow shaping at each source node is provided by the concept of $Virtual\ Link$ (VL). The descriptions of the network and its industrial configuration are given in the following paragraphs.

Network and flow model

The inputs and outputs of the networks are called *end systems* (ES). All the end systems are interconnected by switches. Each end system is connected to exactly one port of an AFDX switch and each port of an AFDX switch can be connected at most to one end system. All the output ports of end systems and switches support a scheduling strategy. All the links in the

network are full-duplex.

A *Virtual Link* (VL) standardized by ARINC-664 is a concept of virtual communication channel, which is the basis of AFDX flows (mono-emitter and multi-receivers). A Virtual Link is characterized by the *Bandwidth Allocation Gap* (BAG), which is the minimum delay between two consecutive frames of the corresponding VL, as well as l_{min} and l_{max} , which are the minimum and maximum frame lengths. As shown in Section 1.3.3, each VL v_i can be described by $\{BAG_i, C_i, J_i, O_i\}$, where the BAG_i value corresponds to its period, C_i is the frame transmission time, J_i is the maximum release jitter and O_i is the offset.

A connection defined by a Virtual Link is unidirectional, including one source end system and one or more paths leading to different destination end systems (multicast). For the purpose of determinism, Virtual Links are statically defined.

There is no synchronization between the end systems. Each end system transmits frames on a set of VLs. Thus each end system has to schedule the transmissions of all its emitting VLs since they are multiplexed on the same physical Ethernet link.

An AFDX industrial configuration

The industrial AFDX network interconnects aircraft functions in the avionics domain. It is a large scale network and composed of two redundant networks. A typical industrial AFDX network architecture is illustrated in Figure 4.1. It supports FIFO scheduling.

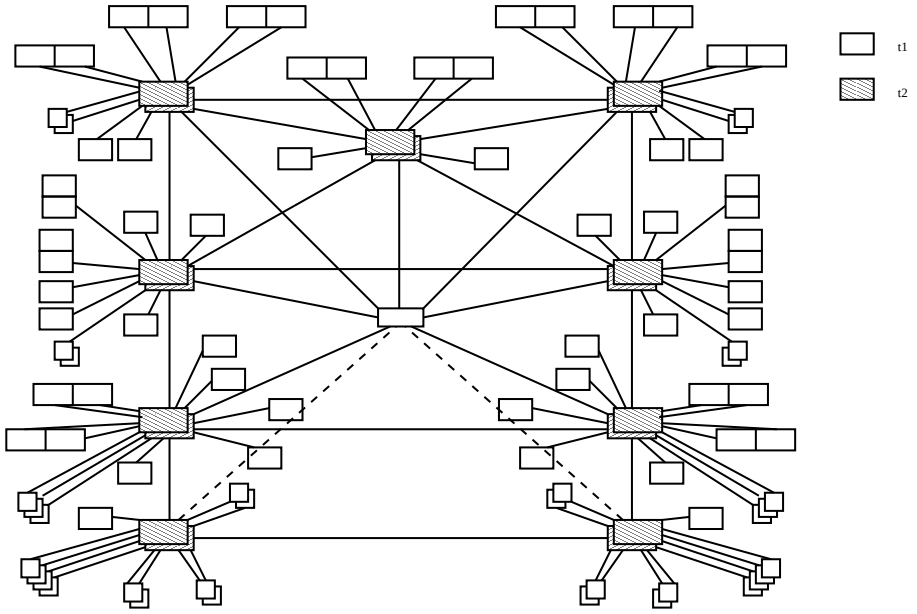


Figure 4.1: An illustrative industrial AFDX network architecture

The network is composed of 123 end systems, 8 switches per network, 984 Virtual Links and 6412 VL paths (due to VL multicast characteristics). The left part of Table 4.1 gives the

dispatching of VLs among BAGs. The right part of Table 4.1 gives the dispatching of VLs among the maximum frame length values l_{max}. The majority of VLs uses short frames.

BAG (ms)	Number of VL
2	20
4	40
8	78
16	142
32	229
64	220
128	255

Frame length (bytes)	Number of VL
0-150	561
151-300	202
301-600	114
601-900	57
901-1200	12
1201-1500	35
> 1500	3

Table 4.1: BAGs and frame lengths

Table 4.2 shows the number of VL paths per length (i.e. the number of crossed switches).

Nb of crossed switches	Number of paths
1	1797
2	2787
3	1537
4	291

Table 4.2: VL paths lengths

This industrial AFDX network works at 100 $Mbit/s$ and the technological latency (switching latency) of an AFDX switch is 16 μs. The overall workload (utilization) of the industrial network is about 10%. Actually, the industrial AFDX network is lightly loaded in order to guarantee that buffers will never overflow. Both sporadic VLs and periodic VLs exist on the AFDX network, and offsets can be assigned to periodic VLs. There is no global clock in an AFDX network. Consequently, frame release times at different end systems are independent.

4.3 Example of an offset assignment algorithm

To evaluate an industrial AFDX configuration, an offset assignment to model flow behaviors is required. Therefore, an algorithm proposed for automotive CAN networks in [GHN08] is to provide an offset allocation example. The aim of this algorithm is to separate frames of one flow as far as possible from the frames of the other flows emitted by the same source node.

This algorithm considers a discrete time system based on a time interval granularity $gran$. A periodic flow τ_i has a period T_i. For n periodic flows emitted by the same source node, the offsets are distributed over time interval $[0, \frac{T_{max}}{gran})$, where $T_{max} = \max_{i \in [\![1,n]\!]}(T_i)$, and the offset

for each flow τ_i is chosen in the interval $[0, T_i)$. The sequence of frame transmissions is stored in an ordered array R, which has $\frac{T_{max}}{gran}$ elements. The n flows are numbered by increasing value of their periods and proceeded from τ_1 to τ_n. The first flow τ_1 is assigned with the offset $O_1 = 0$. The offset assignment of τ_i ($i \in [\![2, n]\!]$) is achieved by the following steps:

1. look for one of the longest idle interval in $[0, T_i)$. The first and last possible release time of the interval are noted by B_i and E_i respectively.
2. then set the offset O_i in the middle of the selected interval. The corresponding possible release time is r_i.
3. finally store the frames of τ_i in the array R in the following way:

$$\forall k \in \mathbb{N} \; and \; r_i + k \cdot \frac{T_i}{gran} \leq \frac{T_{max}}{gran},$$
$$do \; R[r_i + k \cdot \frac{T_i}{gran}] = R[r_i + k \cdot \frac{T_i}{gran}] \cup f_{i,k+1}$$

This algorithm is illustrated by the following small example. Four periodic flows τ_1, τ_2, τ_3, τ_4 emitted at N_1 with periods of 4 ms, 8 ms, 16 ms, 16 ms respectively are sorted by increasing periods and $T_{max} = 16$ ms. τ_1 is assigned with offset $O_1 = 0$ which means $R[1] = f_{1,1}$. According to step 3, $R[3] = f_{1,2}$, $R[5] = f_{1,3}$ and $R[7] = f_{1,4}$. Then for τ_2, $B_2 = 2$ and $E_2 = 3$ (step 1), thus $r_2 = 2$. The array R is updated with $R[2] = f_{2,1}$ and $R[6] = f_{2,2}$ (step 3). The same process is done for τ_3 and τ_4. The R array is listed in Table 4.3 from which we can see that $O_2 = 2$ ms, $O_3 = 6$ ms and $O_4 = 14$ ms.

Time	0	2	4	6	8	10	12	14
$PRT\ i$	1	2	3	4	5	6	7	8
$R[i]$	$f_{1,1}$	$f_{2,1}$	$f_{1,2}$	$f_{3,1}$	$f_{1,3}$	$f_{2,2}$	$f_{1,4}$	$f_{4,1}$

PRT_i is the abbreviation of *Possible release time i*
$R[i]$ records the released frames

Table 4.3: The example of offset assignment

4.4 Worst-case delay analysis based on the modified Network Calculus approach

The industrial avionics network in Figure 4.1 is configured with the example offset assignment presented in 4.3. We assume that all the VLs are periodic. The worst-case end-to-end delay

upper bound of each VL path is computed by the classical Network Calculus approach and the modified Network Calculus approach. A normalization is used for the purpose of comparison. For two computed results of one path $\mathcal{P}_x$, one is considered as the reference value (denoted rf_x) and normalized as 100, while the other (denoted cp_x) is taken as the comparison value and normalized as Ncp_x:

$$Ncp_x = 100 + (\frac{cp_x - rf_x}{rf_x} \times 100) \tag{4.1}$$

The delay upper bound computed by classical Network Calculus approach is taken as the reference value rf_x. The delay upper bound computed by modified Network Calculus approach is taken as the comparison value cp_x. Both rf_x and Ncp_x are shown in Figure 4.2, where $ClassicalNC$ refers to the classical Network Calculus approach, and $ModifiedNC$ refers to the modified Network Calculus approach. The VL paths in Figure 4.2 are sorted by increasing values of their corresponding normalized Ncp_x.

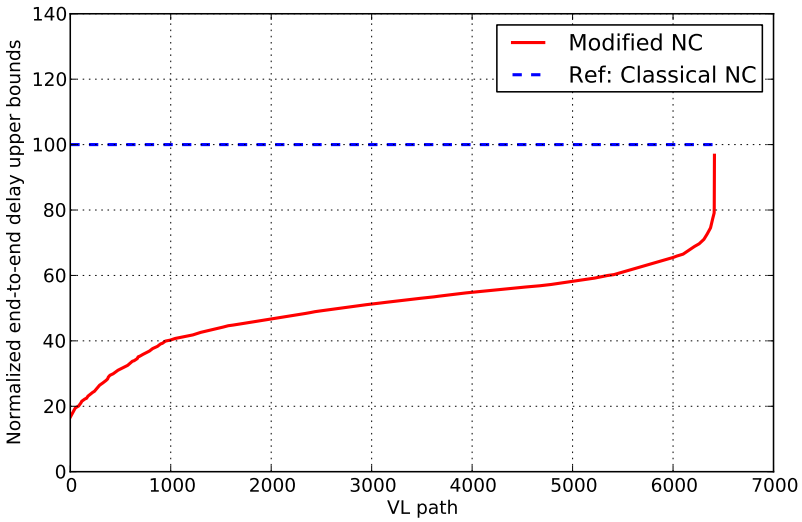


Figure 4.2: Improvement of the modified Network Calculus approach

The average improvement brought by modified Network Calculus approach in terms of ETE delay upper bound is of 49.7%, which reveals the gain of integrating the minimum duration constraints of flows at AFDX end systems. The maximum improvement is up to 83.3%. The main reason of this improvement is that the flows are separated by the minimum durations at each source node, and therefore lead to a reduction of burst workload at the output ports. Since these minimum duration constraints propagate along the path, they also lead to more

even workload at the output ports of switches.

Besides the case that all the flows transmitted in the network are constrained by minimum durations, it is also possible that only part of the flows are dependent while the other flows are independent. It means that only part of the flows are periodic and with known offsets. In order to evaluate this case, it is assumed that 2964 randomly chosen VL paths (almost half of the total VL paths) in the industrial AFDX network are periodic with known offsets. The other 3448 VL paths are either sporadic or periodic with unknown offsets, and they are considered independent. The delay upper bound of each flow is computed by both the classical Network Calculus approach (rf_x) and the modified Network Calculus approach (cp_x). The comparison is shown in Figure 4.3 where VL paths are sorted by increasing values of their corresponding normalized Ncp_x.

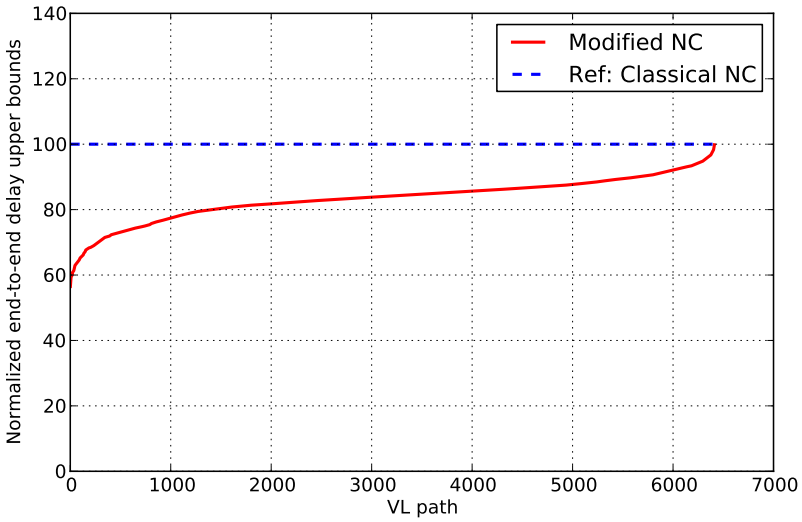


Figure 4.3: Improvement of the modified Network Calculus approach with partial dependencies

From Figure 4.3, there is still an average improvement of 16.7% brought by the modified Network Calculus approach even if only less than half of the flows are dependent. Therefore, it is important to integrate offsets in the computation in order to improve the results.

The offsets separate the dependent flows and improves the delay upper bounds. It is interesting to know if the improvement is similar for both independent and dependent flows. Therefore, the results of delay upper bounds for dependent flows and for independent flows are depicted in Figure 4.4 and in Figure 4.5, respectively.

The results show that for both dependent flows and independent flows, the integration of

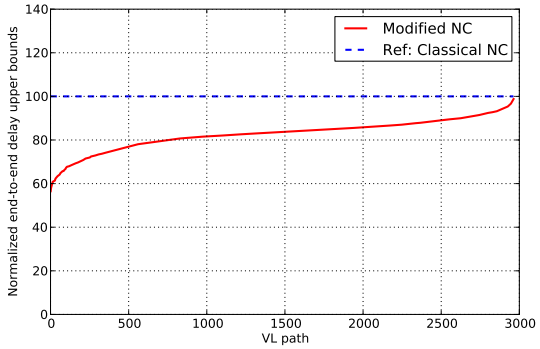


Figure 4.4: Improvement of dependent flows of the modified Network Calculus

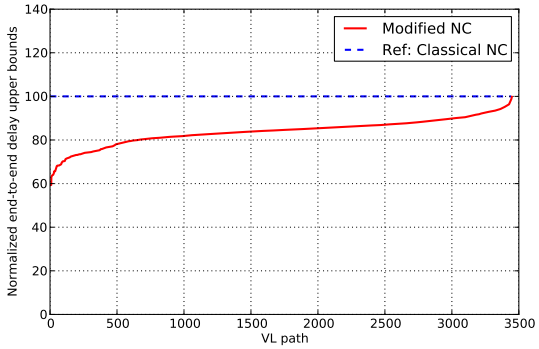


Figure 4.5: Improvement of independent flows of the modified Network Calculus

offsets brings improvement on end-to-end delay upper bounds. The average improvement for dependent flows is of 17.23% which is slightly higher than the 16.15% for independent flows.

4.5 Worst-case delay analysis based on the modified Trajectory approach

In this evaluation, the improvement brought by the integration of the minimum duration constraints is measured for the modified Trajectory approach. The worst-case end-to-end delay upper bound of each VL path is computed by the classical Trajectory approach and the modified Trajectory approach. The normalization presented by Equation 4.1 is used for the purpose of comparison. The results of the comparisons are shown in Figure 4.6, where $ClassicalTraj$ refers to the classical Trajectory approach, and $ModifiedTraj$ refers to the modified Trajectory approach. The VL paths in Figure 4.6 are sorted by increasing values of their corresponding normalized values Ncp_x.

The gap between the obtained Ncp_x and the reference reflects the improvement for each VL path. There is a clear improvement of computed end-to-end delay upper bounds as minimum duration constraints are considered. The maximum improvement is up to 82% and the average improvement is about 50%. The main reason of this improvement is that the flows are separated by the minimum durations at each source node, and therefore lead to a reduction of burst workload at the output ports, not only of source nodes, but also of switches. The following table Table 4.4 shows the distribution of improvements on output port burst workload of both end systems and switches. This improvement means the percentage of burst workload reduction when considering offsets compared to the case where no offsets are accounted for.

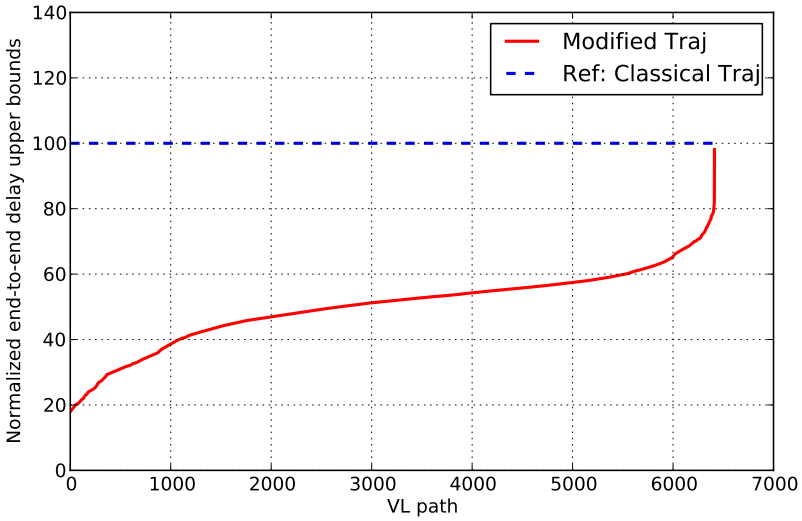


Figure 4.6: Improvement of the modified Trajectory approach

Improvement (%)	Number of ES output ports	Number of switch output ports
0-10	0	72
10-20	0	44
20-30	0	26
30-40	3	4
40-50	5	0
50-60	10	0
60-70	10	0
70-80	42	0
80-90	25	0
90-100	1	0

Table 4.4: Improvement on burst workload at output ports of ESs and switches

From the table, we can see that most end systems have a reduced burst workload of 70%-80% due to the offsets. From the viewpoint of switch output ports, even if the input flows at the switch output ports come from different end systems which are not synchronized, the reduction on burst workload of most switch output ports is still of about 10%. Therefore for such an industrial configuration with the average workload of about 10%, accounting for minimum duration constraints can effectively reduce burst workload, and therefore gives tighter computed end-to-end delay upper bounds.

The scenario with partial dependent flows presented in the Network Calculus evaluation is also presented here for the Trajectory approach. It is assumed that 2964 randomly chosen VL paths in the industrial AFDX network are dependent and the other 3448 VL paths are considered independent. The delay upper bound of each VL path is computed by both the classical Trajectory approach (rf_x) and the modified Trajectory approach (cp_x). The comparison is shown in Figure 4.7 where VL paths are sorted by increasing values of their corresponding normalized value Ncp_x.

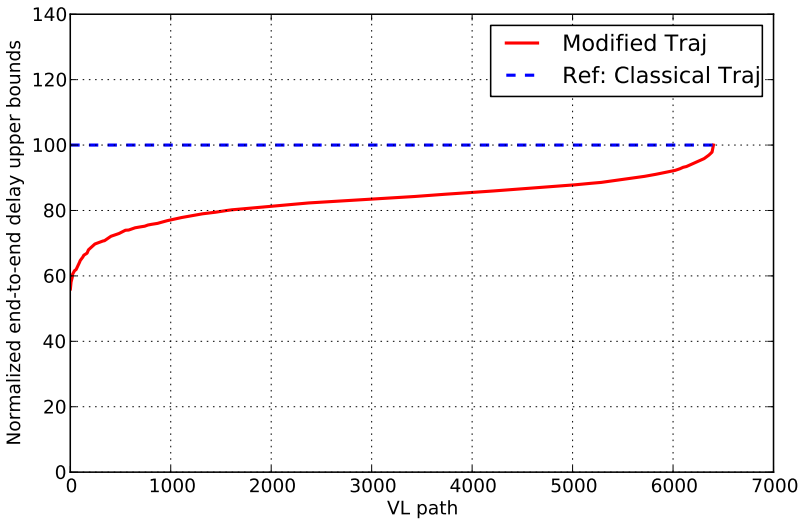


Figure 4.7: Improvement of the modified Trajectory approach with partial dependent flows

From Figure 4.7, it can be seen that there is still an average improvement of 17% brought by the modified Trajectory approach even if only less than half of the flows are dependent. This result complies with the results obtained by the modified Network Calculus approach.

Similarly, the improvements on dependent flows and on independent flows are evaluated

and depicted in Figure 4.8 and in Figure 4.9, respectively. The average improvement on dependent flows is 17.72% which is slightly higher than the 15.99% of independent flows. Thus, these results are in line with the results obtained by the modified Network Calculus approach. Integrating the temporal constraints imposed by the offsets, even partially, in the computation improves the results on end-to-end delay upper bounds as well.

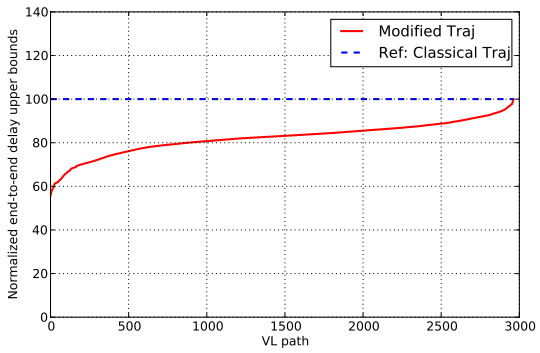


Figure 4.8: Improvement of dependent flows based on the Trajectory approach

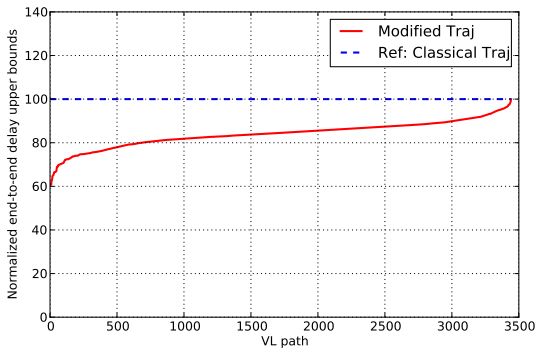


Figure 4.9: Improvement of independent flows based on the Trajectory approach

4.6 A comparison of the two modified approaches

The modified Network Calculus approach takes into account the frame serialization effect on link. It is still holistic since the computation considers the worst case scenario for a frame at each output port it visits. The computation of the proposed Trajectory approach, which integrates offsets, considers the worst case scenario of a frame along its path. In previous paragraphs, both approaches are applied to the industrial AFDX network. This section compares the results of the modified Network Calculus approach to the ones of the proposed Trajectory approach. The same normalization method is adopted with the reference value rf_x chosen as the end-to-end delay upper bounds computed by the modified Network Calculus approach, while the comparison value cp_x chosen as the end-to-end delay upper bounds computed by the proposed Trajectory approach. Since both two modified approaches are pessimistic computations, an optimized approach (denoted *Opt*) can be achieved by taking the tighter (smaller) value of the two approaches for each VL path. The result is shown in Figure 4.10.

The modified Trajectory approach brings tighter end-to-end delay upper bounds than the modified Network Calculus approach to 5956 VL paths which takes up 92% of the VL path total number and correspondingly it brings looser upper bounds to 456 VL paths. The maximum

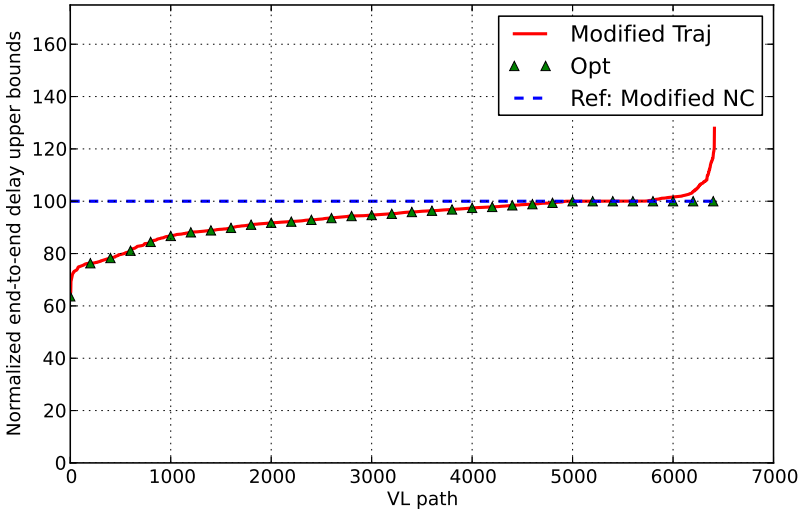


Figure 4.10: Comparative results of two approaches

gain and the maximum loss are of 36% and 28% respectively, and the average gain is of 6%. The modified Trajectory approach works better than the modified Network Calculus approach on this network for the bigger portion of VL paths. The optimized approach provides a further improved worst-case delay analysis. It brings on average 50.2% improvement on end-to-end delay upper bounds which is slightly better than the modified Trajectory approach.

4.7 Near-optimal offset assignment for the industrial AFDX network

Each source node of the switched network emits its flows according to a local clock. It assigns an offset to each flow. It means that different offset assignments of flows can lead to different combinations of flow arrival times at one source node. Therefore, it is interesting to find out which offset assignment leads to a tighter delay upper bound for a given system. This issue has been addressed in the context of uniprocessor task scheduling in [Goo03, GGN06] as well as in the context of automotive networking in [GHN08]. In previous paragraphs, only the offset assignment originally designed for the CAN network in [GHN08] has been applied to the industrial AFDX network. It is interesting to consider other existing offset assignment algorithms [Goo03, GGN06] in order to find the algorithm that minimizes the delay upper bounds of flows in the industrial AFDX network. Moreover, the existing algorithms can be

adapted in order to take into account specific characteristics of an AFDX network.

4.7.1 Existing offset assignments

The offset assignment has been studied in [Goo03] in the context of periodic task sets executed on a unique processor. Each task τ_i is characterized by a period T_i, a hard deadline D_i, a processing time C_i and an offset O_i. In the context of uniprocessor, the systems can be classified into three classes with respect of their offsets:

- Synchronous system: all the tasks have the same fixed offsets $O_i = 0$, i.e., at time 0, all the tasks generate one request;
- Asynchronous system: a different offset is allocated to each task due to application constraints;
- Offset free system: any offset can be allocated to each task in order to improve the system schedulability.

A key point of an offset free system is the choice of an offset assignment. However, the number of possible offset assignments is exponential.

In [Goo03], an *Optimal offset assignment* is proposed to exhaust all possible non-equivalent offset assignments. Although this method reduces significantly the number of combinations, the number remains exponential. Therefore this offset algorithm is intractable for large scale network.

Dissimilar offset assignment is then defined in [Goo03] in order to reduce computational complexity in the comparison with the *Optimal offset assignment* by providing a single offset assignment for a task set. This method tries to move from the synchronous case as much as possible. It considers a set of pairs of tasks. Each pair has two tasks (τ_i, τ_j). Then it orders task pairs (τ_i, τ_j) by decreasing value of $gcd(T_i, T_j)$, where $gcd(T_i, T_j)$ is the greatest common divisor of T_i and T_j. When O_i is known, this algorithm assigns an offset O_j to τ_j by separating τ_j from τ_i by a minimal distance $\lfloor \frac{gcd(T_i,T_j)}{2} \rfloor$ between two requests of τ_i and τ_j. *Dissimilar offset assignment* is denoted GCD in this chapter.

Near-optimal offset assignment heuristics are derived in [GGN06] based on the study of GCD. When GCD fails to generate a schedulable asynchronous situation, this algorithm orders task pairs by considering one of four alternative criteria:

- *RateAdd*: $\frac{C_i}{T_i} + \frac{C_j}{T_j}$,
- *RAGCD*: $(\frac{C_i}{T_i} + \frac{C_j}{T_j}) \times gcd(T_i, T_j)$,
- *RMGCD*: $max(\frac{C_i}{T_i}, \frac{C_j}{T_j}) \times gcd(T_i, T_j)$;
- *GCDMinus*: $-gcd(T_i, T_j)$.

Then it assigns offsets to each task pair in the same way as *GCD*. Since both *Dissimilar offset assignment* and *Near-optimal offset assignment heuristics* assign offsets to tasks pair by pair, they are called *PairAssign* in this Chapter.

In [GHN08], the authors show that the offset assignments mentioned above are not efficient when applied to the scheduling of automotive message, and an offset assignment algorithm is tailored for automotive CAN networks. This algorithm aims at choosing offsets to maximize the distance between frames. It is presented in 4.3, and it is denoted *SingleAssign* and recalled briefly here. For n flows emitted by one source node, sort them by increasing value of their periods and calculate $T_{max} = \max_{i \in [\![1,n]\!]}(T_i)$. The assignment starts with the flow having the smallest period and processes one flow after the other. The first flow is assigned with the offset 0. For a flow τ_i ($i \in [\![2, n]\!]$), its offset O_i is derived as follows:

- first, search for the longest idle interval in $[0, T_i)$;
- then set O_i in the middle of this interval;
- finally record all the frames of τ_i released in $[0, T_{max})$.

It is noted that different network applications have various temporal characteristics, then the efficiency of an offset assignment highly depends on the specific network application. In this chapter, we specially consider the industrial AFDX configuration presented in Section 4.2. In the following paragraphs, an optimal scenario is developed and different offset assignments are then evaluated based on the optimal offset scenario for the industrial AFDX configuration.

4.7.2 Optimal scenario of dependent flows over the AFDX network

Considering the industrial AFDX configuration (presented in Section 4.2) with about 1000 flows, the *Optimal offset assignment* proposed in [Goo03] is intractable. Thus other offset assignment algorithms have to be used. Then, the evaluation of the gap between the *Optimal offset assignment* and the assignment generated by each heuristic is an important issue. For a

given flow, this gap is the difference between the worst-case ETE delay obtained by *Optimal offset assignment* and the one obtained by a chosen offset assignment algorithm. On a whole configuration, the gap is the average of the gaps obtained for all the flows. Indeed, it is not possible to compute such a gap for an industrial AFDX configuration, since *Optimal offset assignment* is intractable. Then, a first idea is to compute an upper bound on this gap.

This upper bound can be obtained by considering an ideal offset assignment, which minimizes the worst-case ETE delay for all the flows. This ideal assignment may not exist for a given configuration, but it is sure that it gives worst-case delays which are not higher than the ones obtained by the *Optimal offset assignment*. This ideal assignment, denoted *IdealAssign*, minimizes the maximum waiting delay of every frame in each output port that it crosses by considering that all the frames emitted by the same end system are separated by an infinite distance. It corresponds to the following scenario:

- At a source ES, a frame f_i of a VL v_i is not delayed by any other frames emitted by the same ES, i.e., the frame f_i is transmitted immediately after its release;
- At each switch output port of its path, the frame f_i encounters frames generated by several ESs. f_i can be delayed by exactly one frame coming from each of these ESs. The frame with the largest size is considered. The delay experienced by f_i at each switch output port takes into account the serialization effect (i.e., two frames cannot be received at the same time from an input link).

Indeed, since there is no common clock among the end systems, there is no relationship between the releases of two frames from different end systems. Consequently, there exist scenarios where the two frames arrive at their first common switch output port at the same time.

Let us illustrate this scenario on Example 1 depicted in Figure 4.11.

Example 1 *There are four flows transmitted: v_1 and v_2 emitted by the end system e_1 as well as v_3 and v_4 emitted by the end system e_2. The temporal characteristics of each VL are listed in Table 4.5 The sample AFDX network works at 100 $Mbit/s$. The technological latency is null.*

We first focus on VL v_1. The *IdealAssign* leads to the scenario illustrated in Figure 4.12, where $a^h_{f_i}$ is the arrival time of frame f_i on the output port h. At e_1, the frame f_1 is transmitted as soon as it is released due to the separation from v_2. Since the ESs are not synchronized, at

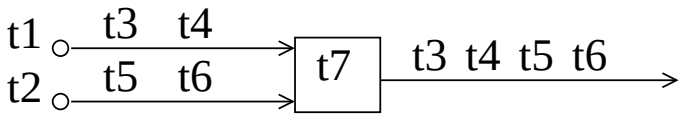


Figure 4.11: A sample AFDX network of Example 1

v_i	BAG_i (μs)	l_{max_i} (bit)	C_i (μs)	J_i (μs)
v_1	8000	8000	80	0
v_2	8000	8000	80	0
v_3	8000	8000	80	0
v_4	8000	4000	40	0

Table 4.5: The Configuration of flows of Example 1

the output port of the switch S_1, the frame f_1 of v_1 can arrive at the same time as the frame f_3 of v_3 and it is delayed by f_3, i.e. $a_{f_1}^{S_1} = a_{f_3}^{S_1}$. Only one frame from e_2 delays the frame f_1 at the output port of S_1 since v_3 and v_4 are separated far away from each other, and f_3 is considered due to the frame size $l_{max_3} > l_{max_4}$.

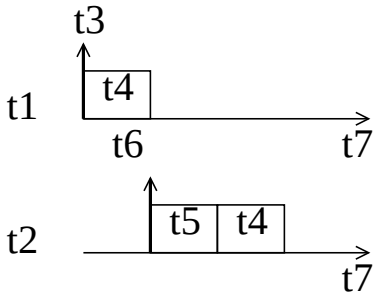


Figure 4.12: Scenarios of the VL v_1

IdealAssign gives an upper bound on the delay reduction which can be obtained by an offset assignment algorithm. The following paragraphs propose some offset assignment heuristics tailored for the AFDX network.

4.7.3 Offset assignment algorithms in the context of AFDX network

In the context of a uniprocessor, a set of tasks shares a unique resource, i.e., the processor. The situation is different in the context of a switched Ethernet network, like the AFDX network, where a set of flows shares a set of output ports. Actually, each port is shared by a subset of all the flows. Consequently, the load can be different for each output port. The worst-case waiting

time of a frame in an output port increases when the load of the output port increases. Then, it could be interesting to take into account the load of the output port in the offset assignment.

This is illustrated on Example 2 in Figure 4.13,

Example 2 *There are six flows transmitted:* v_1, v_2 *and* v_3 *emitted by the end system* e_1 *as well as* v_4, v_5 *and* v_6 *emitted by the end system* e_2. *The temporal characteristics of each VL are listed in Table 4.6. The sample AFDX network supports FIFO scheduling and works at* $R = 100$ $Mbit/s$. *The technological latency is null.*

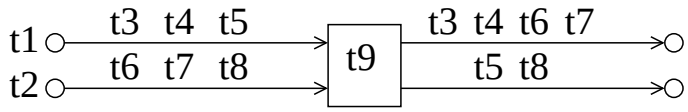


Figure 4.13: A small example of AFDX network of Example 2

v_i	BAG_i (μs)	l_{max_i} (bit)	C_i (μs)	J_i (μs)
v_1	400	4000	40	0
v_2	800	6000	60	0
v_3	400	6000	60	0
v_4	400	4000	40	0
v_5	800	6000	60	0
v_6	400	6000	60	0

Table 4.6: The Configuration of flows of Example 2

The offset assignment *SingleAssign* is applied to this example network. The three VLs emitted by e_1 are considered. Since *SingleAssign* assigns offsets to VLs in the order of increasing BAG values and $BAG_1 = BAG_3 < BAG_2$, the offsets are assigned to these three VLs in order:

$$O_1 = 0\ \mu s,\ O_3 = 200\ \mu s,\ O_2 = 100\ \mu s$$

This case is shown for the chronogram of e_1 in Figure 4.14. Similar case at the end system e_2 is depicted for the chronogram of e_2 in Figure 4.14. We focus on v_2 whose first frame f_2 is released at $O_2 = 100\ \mu s$. At the output port of the switch S_1 that v_2 visits, v_4 and v_5 from e_2 join the path of v_2 while v_3 has left. Then one possible scenario at this output port is depicted for the chronogram of S_1 in Figure 4.14. It can be seen that when the frames f_1 and f_2 arrive at S_1, they are still separated long enough to avoid delaying each other. Similarly, the frames f_4 and f_5 from v_4 and v_5 are separated long enough when they arrive at S_1, consequently only one

frame f_5 delays the studied frame f_2. Since the frame f_2 is released at e_1 at time $O_2 = 100\ \mu s$ and the transmission of frame f_2 is finished at the switch S_1 at time 280 μs, the end-to-end delay (shadow block in Figure 4.14) of the frame f_2 is:

$$R_2 = 280 - 100 = 180\ \mu s$$

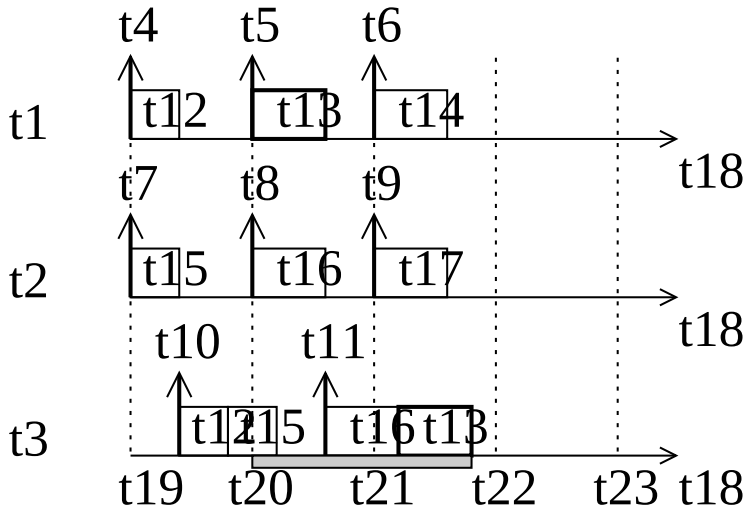


Figure 4.14: Illustration of the *SingleAssign* with low workload

The illustration in Figure 4.14 shows an example where the offset assignment *SingleAssign* succeeds to distribute the workload even in the output port of a switch. It is interesting to demonstrate the case when the workload increases. The example AFDX network in Figure 4.13 is under study and the maximum frame sizes of v_1 and v_4 are increased to:

$$l_{max_1} = l_{max_4} = 6000\ bits\ (C_1 = C_4 = 60\ \mu s)$$

According to *SingleAssign*, the release of frames at e_1 and e_2 is depicted in Figure 4.15 (same as in Figure 4.14). One possible scenario at the output port of S_1 is exhibited for the chronogram of S_1 in Figure 4.15, where the studied frame f_2 finishes its transmission at time 300 μs. The delay of the frame f_2 is:

$$R_2 = 300 - 100 = 200\ \mu s$$

This delay is higher than the one obtained in the case of Figure 4.14 (180 μs). It increases

due to the fact that when the frame f_2 arrives at S_1 at time $a_{f_2}^{S_1} = 160\ \mu s$, the transmission of frame f_4, delayed by the transmission of frame f_1, is not completed. It thus delays the transmission of frame f_2. For this case *SingleAssign* could not separate frames at an output port of a switch.

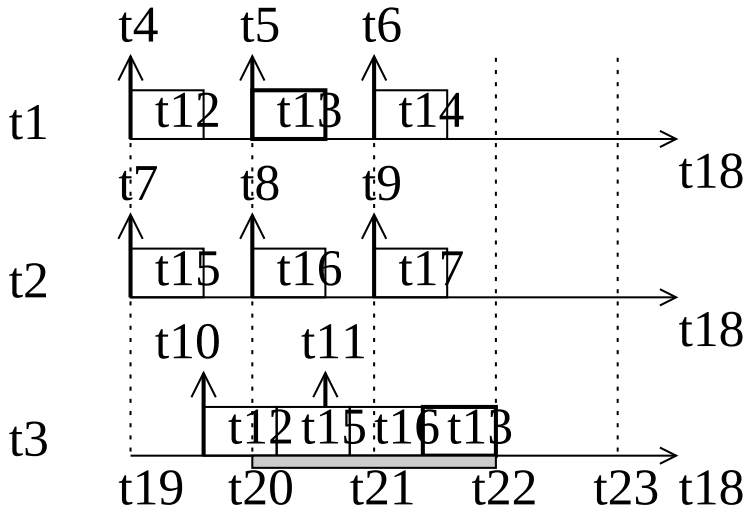


Figure 4.15: Illustration of *SingleAssign* with increased workload

Note that v_1, v_2 and v_3 emitted by e_1 visit three output ports: the output port of e_1, the upper output port of S_1 and the lower output port of S_1. The output port of e_1 is shared by VLs v_1, v_2 and v_3, creating a utilization of:

$$U_{e_1} = \sum_{\{v_1,v_2,v_3\}} (\frac{C_1}{BAG_1} + \frac{C_2}{BAG_2} + \frac{C_3}{BAG_3}) = 0.375$$

The upper output port of S_1 is shared by VLs v_1, v_2, v_4 and v_5, creating a utilization of:

$$U_{S_1} = \sum_{\{v_1,v_2,v_4,v_5\}} (\frac{C_1}{BAG_1} + \frac{C_2}{BAG_2} + \frac{C_4}{BAG_4} + \frac{C_5}{BAG_5}) = 0.45$$

The lower output port of S_1 is shared by VLs v_3 and v_6, creating a utilization of:

$$U_{S_1'} = \sum_{\{v_3,v_6\}} (\frac{C_3}{BAG_3} + \frac{C_6}{BAG_6}) = 0.3$$

Consequently, for these three VLs, the most loaded port is the upper output port of S_1, followed by e_1 and the lower output port of S_1. We could first assign offsets to v_1 and v_2

which visit the most loaded port of S_1, then go on with v_3, leading to the offsets: $O_1 = 0\ \mu s$, $O_2 = 200\ \mu s$ and $O_3 = 100\ \mu s$. This case is illustrated in part e_1 in Figure 4.16. Similar case for the VLs emitted by e_2 is shown for e_2 in Figure 4.16. Then one possible scenario for the frame f_2 at S_1 is identified for S_1 in Figure 4.16, indicating that the delay of this frame is:

$$R_2 = 380 - 200 = 180\ \mu s$$

It is smaller than the one obtained by *SingleAssign* (200 μs). The reason is that at the most loaded output port of S_1 the workload is further evenly distributed to reduce the waiting time in the buffer.

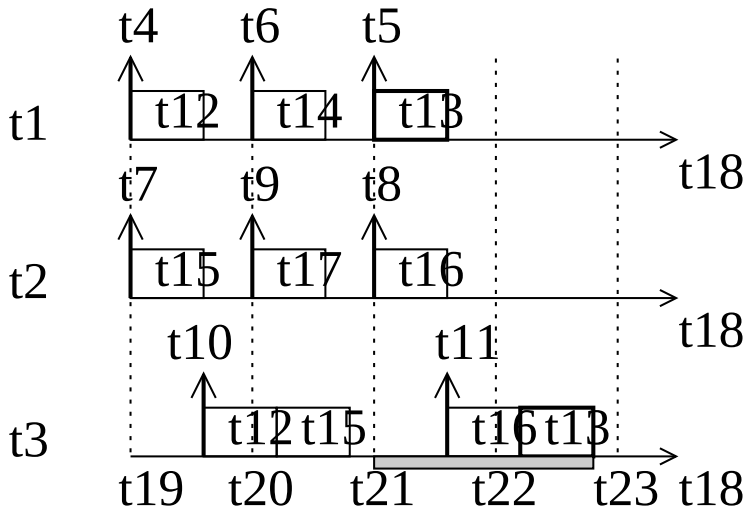


Figure 4.16: Illustration of *MostLoadSA* with high workload

A proposed algorithm considers separating the VLs by assigning offsets to VLs emitted by the same ES in the order of decreasing utilization of the output ports they share. The offsets are first assigned to the flows visiting the most loaded port using the assignment *SingleAssign*, then to the flows which are not yet handled in the secondly most loaded port till all the flows of one ES are assigned with offsets. This algorithm is developed based on the assignment *SingleAssign* and denoted *MostLoadSA*. This offset assignment is illustrated in previous paragraphs and in Figure 4.16.

For the *PairAssign*, a similar heuristic is proposed to consider the load of the output ports. This heuristic, denoted $MostLoad$, sorts the VL pairs (v_i, v_j) by decreasing values of $Ld_i + Ld_j$, where Ld_i is the workload (utilization) of most loaded output port crossed by v_i. We illustrate this offset assignment using the same example shown in Figure 4.13 and Table 4.6, except that

we consider $C_1 = C_4 = 60\ \mu s$. Three VL pairs are considered for v_1, v_2 and v_3 emitted by e_1: (v_1, v_2), (v_1, v_3) and (v_2, v_3). For the first pair (v_1, v_2), the most loaded output port crossed by v_1 and v_2 is the upper output port of S_1, and then $Ld_1 + Ld_2 = 0.45 + 0.45 = 0.9$. Then, for the second pair (v_1, v_3), the most loaded output ports crossed by v_1 and v_3 are the upper output port of S_1 and the output port of e_1 respectively, and then $Ld_1 + Ld_3 = 0.45 + 0.375 = 0.825$. Similarly, for the third pair (v_2, v_3), we have $Ld_2 + Ld_3 = 0.45 + 0.375 = 0.825$. Therefore, the offsets are first assigned to v_1 and v_2, i.e., $O_1 = 0\ \mu s$ and $O_2 = 200\ \mu s$. Then the offset is assigned to v_3 which is $O_3 = 100\ \mu s$. It is the same process for VLs v_4, v_5 and v_6 emitted by e_2. The assigned offsets are listed in Table 4.7.

VL	SingleAssign	MostLoadSA	MostLoad	CrossedSSA	CrossedS
v_1	0 μs	0 μs	0 μs	0 μs	0 μs
v_2	100 μs	200 μs	200 μs	200 μs	200 μs
v_3	200 μs	100 μs	100 μs	100 μs	100 μs
v_4	0 μs	0 μs	0 μs	0 μs	0 μs
v_5	100 μs	200 μs	200 μs	200 μs	200 μs
v_6	200 μs	100 μs	100 μs	100 μs	100 μs

Table 4.7: The assigned offsets based on different offset assignments

Due to the nature of switched Ethernet, flows in one set can share several output ports. When flows share several common output ports of switches, the minimum interval between two frames decreases, which can increase the waiting time of a frame in the output port. Then the number of crossed switches can be considered in the offset assignments. For the *SingleAssign*, a heuristic, denoted *CrossedSSA*, is proposed. *CrossedSSA* first orders the VLs emitted by one end system by decreasing values of maximum number of crossed switch output ports, and then orders the VLs by *SingleAssign*. Consider the example in previous paragraph. v_1, v_2 and v_3 emitted by e_1 all cross one switch output port, then they are assigned offsets in the order of v_1, v_2 and v_3. Similar process happens for v_4, v_5 and v_6 emitted by e_2. The assigned offsets are listed in Table 4.7. For the *PairAssign*, a similar heuristic, denoted *CrossedS*, is proposed. It sorts the VL pairs (v_i, v_j) by decreasing values of $cs(v_i, v_j)$, where $cs(v_i, v_j)$ is the number of common switch output ports crossed by v_i and v_j. For the same example, three VL pairs are considered for v_1, v_2 and v_3 emitted by e_1: (v_1, v_2), (v_1, v_3) and (v_2, v_3). For the first pair (v_1, v_2), v_1 and v_2 share one switch output port of S_1. For the second pair (v_1, v_3), v_1 and v_3 do not share any switch output port. For the third pair (v_2, v_3), v_2 and v_3 do not share any switch output port. Therefore, offsets are assigned to v_1 and v_2, and then to v_3, which are also listed in Table 4.7.

Besides the four new proposed heuristics, the existing offset assignment heuristics presented

in Section 4.7.1 are applied to the AFDX network with the value of BAG as the period. The evaluation on each offset assignment is presented in the next section.

4.7.4 Results

The existing and proposed offset assignments introduced in Section 4.7.3 are applied to the industrial AFDX network presented in 4.2. In this evaluation, all the VLs are assumed to be periodic. The computation is processed using the modified Network Calculus approach integrating the offsets. The computed ETE delay upper bounds for each offset assignment are compared with those obtained from the network without offset constraints. The statistic reductions on ETE delay upper bounds of each algorithm are listed in Table 4.8. The columns *Average*, *Max* and *Min* give the average, maximum and minimum reductions, respectively.

Heuristics	Average %	Max %	Min %
IdealAssign	53.48	83.29	21.00
GCD	23.00	70.24	4.01
RateAdd	32.89	73.50	5.08
RAGCD	32.51	72.99	8.85
RMGCD	32.29	70.77	9.99
GCDMinus	32.95	70.06	8.83
MostLoad	32.12	70.06	8.84
CrossedS	32.32	73.03	8.90
SingleAssign	49.67	83.29	18.84
MostLoadSA	51.32	82.94	18.84
CrossedSSA	51.29	82.94	18.84

Table 4.8: Statistic reduction on end-to-end delay upper bounds for each offset algorithm

SingleAssign as well as its extended algorithms *MostLoadSA* and *CrossedSSA* outperform the *PairAssign* heuristics. Indeed, the average reductions obtained with the *PairAssign* heuristics are 23% (*GCD*) and 32% (*RateAdd*, *RAGCD*, *RMGCD*, *GCDMinus*, *MostLoad* and *CrossedS*). It is 49% for the *SingleAssign* and 51% for the *SingleAssign* based algorithms adapted to the AFDX network. On the considered example, the *SingleAssign* based algorithms are close to the *IdealAssign* with an average reduction of 53%.

The *PairAssign* heuristics are not efficient in the studied context due to the limited different values of BAG, which lead to few values of $gcd(BAG_i, BAG_j)$ for different VL pairs. It means that different VL pairs can have the same value of $gcd(BAG_i, BAG_j)$. A small example is given in Figure 4.17. Considering VLs v_1, v_2 and v_3 with $BAG_i = 4\ ms$ ($i \in [\![1, 3]\!]$) of e_1, there are three pairs: (v_1, v_2), (v_1, v_3) and (v_2, v_3). They have the same value of $gcd(BAG_i, BAG_j) =$

4 ms ($i \in [\![1,3]\!]$). *GCD* leads to $O_1 = 0\ ms$, $O_2 = O_1 + \frac{gcd(BAG_1,BAG_2)}{2} = 2\ ms$, and $O_3 = O_1 + \frac{gcd(BAG_1,BAG_3)}{2} = 2\ ms$ ($O_2 = O_3$). The releases of the first frames for both v_2 and v_3 overlap, and the frames have to wait in the queue. This case is depicted in Figure 4.18.

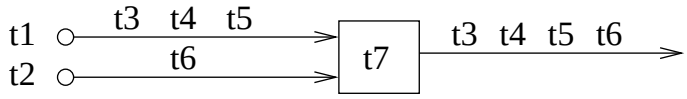


Figure 4.17: A small example of AFDX

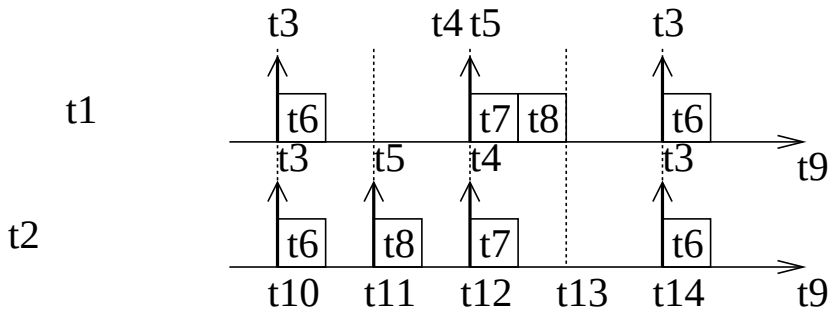


Figure 4.18: Comparison of GCD and SingleAssign

The situation is different when applying the offset assignment *SingleAssign* (Figure 4.18). With the same configuration, the offsets are set in order: $O_1 = 0\ ms$, $O_2 = 2\ ms$ and $O_3 = 1\ ms$. In this way, no frame has to wait in the output queue of e_1.

The analyzed problem of *GCD* for the industrial AFDX network exists for all the *PairAssign* based heuristics because the computation of offsets mainly concerns the value of $gcd(BAG_i, BAG_j)$ even if the order of pairs varies based on different criteria.

The results are further studied by the normalized method in Equation 4.1. For one path $\mathcal{P}_x$, the computed ETE delay upper bound without offset assignment is considered as the reference value (denoted rf_x) and normalized as 100. The computed result with one offset assignment (denoted cp_x) is taken as the comparison value and normalized as Ncp_x. All the 6412 VL paths are sorted by increasing order of Ncp_x. Three offset assignments are taken into account: *IdealAssign*, *SingleAssign*, and *MostLoadSA*. The comparative results are presented in Figure 4.19.

It can be seen in Figure 4.19 that the *MostLoadSA* curve is close to the *IdealAssign* curve, which reveals that the algorithm which takes into account the AFDX properties works well on this industrial AFDX configuration. The gap between the *SingleAssign* curve and the

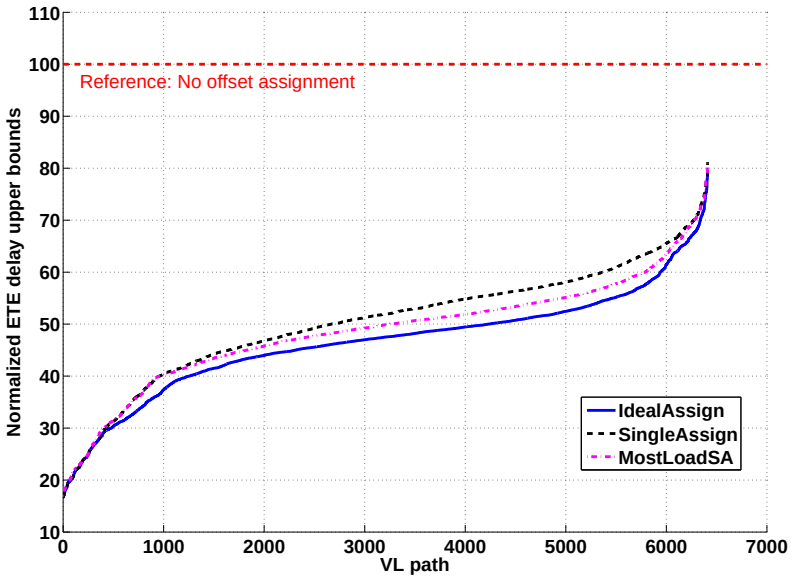


Figure 4.19: Comparative results of *IdealAssign*, *SingleAssign* and *MostLoadSA*

IdealAssign curve is also small (although bigger than the gap with *MostLoadSA* curve). It suggests that a simple algorithm could be efficient to separate the flows of the industrial AFDX network.

Further evaluations have been conducted, leading to the same conclusions. They consider the same industrial AFDX architecture and a same overall workload of 10%. For each VL, the l_{min} and l_{max} are randomly chosen from 72 *bytes* to 1526 *bytes*, and the BAG value is randomly chosen from 1 ms to 128 ms as the powers of 2. The results show that the average ETE delay reduction brought by the $IdealAssign$ is 45%. The $PairAssign$ heuristics bring average reductions ranging from 24% to 31%, which are far from the $IdealAssign$. The algorithms based on the $SingleAssign$ bring average reductions ranging from 39% to 40%, which are closer to the $IdealAssign$.

4.8 Conclusion

In this chapter, we have presented the AFDX network as a real-time switched Ethernet network example. An AFDX industrial configuration which contains about 1000 flows is introduced. Evaluations on such an industrial configuration have been conducted for the modified Network

Calculus approach and for the modified Trajectory approach. The results have shown that considering the minimum duration constraints improves significantly the computed worst-case delay upper bounds by 49.7% for the modified Network Calculus approach and by 50% for the modified Trajectory approach. The comparison of results obtained by both approaches has shown that the modified Trajectory approach brings improvement for 92% of flows compared to the modified Network Calculus approach. An optimized approach based on both two modified approaches has been proposed which further improves the computation on end-to-end delay upper bounds.

The offset assignments for the industrial AFDX network have also been studied in this chapter. Since the *Optimal offset assignment* is intractable in this context, an optimal scenario has been built based on a presumed optimal assignment in order to upper bound the gap between the *Optimal offset assignment* and each offset assignment heuristic. New heuristics considering the AFDX characteristics have been proposed. Using the Network Calculus approach, the improvement on ETE delay upper bound bought by each heuristic has been compared to the optimal algorithm. It is demonstrated that *PairAssign* heuristics are not efficient when applied to the industrial AFDX network due to the limited amount of different BAG values. The *SingleAssign* turns out to be a near optimal algorithm in the studied context. Although the heuristics integrating specific AFDX characteristics bring slight improvements in contrast to the *SingleAssign*, they are of increased complexity.

Chapter 5

Pessimism analysis

Contents

5.1 Introduction 97
5.2 Pessimism analysis based on the classical Trajectory approach . . 99
5.2.1 Review of the classical Trajectory approach 99
5.2.2 Pessimism analysis of computing flows 101
5.2.3 Pessimism analysis of busy period transition 105
5.2.4 Pessimism analysis of serialization effect 108
5.2.5 Analytical method for underestimated delays 115
5.2.6 Analytical method for maximum potential pessimism 115
5.3 Integration of the minimum duration constraints 116
5.3.1 Review of the modified Trajectory approach 117
5.3.2 Illustration on the overestimation of dependent flows 117
5.3.3 Pessimism analysis of the workload of dependent flows 120
5.4 Application on the AFDX network 121
5.4.1 Upper bounding the pessimism of the Network Calculus approach . . 122
5.4.2 Upper bounding the pessimism of the Trajectory approach 122
5.5 Conclusion 124

5.1 Introduction

In previous chapters, two approaches, a modified Network Calculus approach and a modified Trajectory approach, are proposed for the worst-case delay analysis of real-time switched Ethernet networks considering minimum duration constraints. In order to guarantee the delay upper bounds on real-time switched Ethernet networks, some pessimism is introduced in the worst-case delay analysis. The introduced pessimism of a computed delay upper bound is the gap between the exact worst-case delay and the computed value. The percentage of the introduced pessimism out of the computed delay upper bound indicates how pessimistic the computation is. Thus, an evaluation of this pessimism is an importance issue. Since the exact worst-case delay is unknown, the difference between an underestimated value of worst-case delay and an

upper bounded delay (overestimated) is considered. The maximum difference between the underestimated delay and the overestimated delay gives a metric of the introduced pessimism. The pessimism analysis is an essential part of worst-case delay analysis, since it evaluates the credibility of the computed delay upper bounds.

Similar work about pessimism analysis includes [BB08, FJJ09, BSF10]. In [BB08], a quantitative metric is proposed to compare the deviation between a given scheduling algorithm and an optimal scheduling in the context of uniprocessor with FP scheduling. In [FJJ09], a Network Calculus based approach is proposed to guarantee real-time communication of a switched network with FIFO scheduling. The pessimism introduced by such an approach is measured by the difference between the computed worst-case delay and a simulated worst-case delay which is obtained by simulation. However, no further details about how to build the simulation is provided.

In [BSF10] an empirical pessimism analysis based on an unfavorable scenario was built in the context of avionics switched Ethernet with FIFO scheduling. In this unfavorable scenario, only one frame for each sporadic flow is considered and all the considered frames are sorted based on the following criteria:

- At one output port, the frames are sorted first based on their arrival times according to FIFO scheduling;
- If they have the same arrival time, they are sorted by increasing number of shared output ports along the path of the considered flow from the current output port;
- If they have the same number of shared output ports, they are sorted by decreasing frame size.

Pessimism can be obtained only based on a simulation since the first criterion is the frame arrival times which are decided by simulation. Therefore, such a pessimism analysis is empirical, which means that a scenario corresponding to each configuration has to be built. Recently, this work has been extended for FP/FIFO scheduling policies for the avionics switched Ethernet network in [BSF12].

This chapter proposes an analytical approach, which is based on the Trajectory approach, to evaluate the maximum potential pessimism introduced by the worst-case delay analysis of real-time switched Ethernet supporting FIFO scheduling. Therefore it is first necessary to identify the sources of pessimism of the proposed Trajectory approach, and then to derive an

analytical approach to evaluate the maximum potential pessimism by integrating minimum duration constraints for periodic flows with known offsets. The pessimism introduced by both modified Network Calculus approach and modified Trajectory approach is evaluated on the industrial AFDX configuration.

This chapter is organized as follows. Section 5.2 presents the pessimism analysis of a real-time switched Ethernet network based on the classical Trajectory approach. It first identifies all the possible sources of pessimism and then gives an analytical method to upper bound the maximum potential pessimism. Section 5.3 integrates the minimum duration constraints and develops a pessimism analysis based on the modified Trajectory approach. Section 5.4 applies the developed pessimism analysis on the industrial avionics switched Ethernet network. Section 5.5 concludes this chapter.

5.2 Pessimism analysis based on the classical Trajectory approach

5.2.1 Review of the classical Trajectory approach

The Trajectory approach considers for a frame f_i of flow τ_i the worst-case scenario along its trajectory. According to Chapter 3, for frame f_i generated at time t, the latest starting time $W_{i,t}^{last_i}$ at its last visited output port $last_i$ is computed by Term 3.2, Term 3.3, Term 3.4, Term 3.5 and Term 3.6 for FIFO scheduling. The computation is recalled by Term 5.1, Term 5.2, Term 5.3, Term 5.4 and Term 5.5:

$$
\begin{aligned}
W_{i,t}^{last_i} = & \sum_{\substack{j\in[\![1,n]\!] \\ \mathcal{P}_i\cap\mathcal{P}_j\neq\emptyset}} (1+\lfloor\frac{t+A_{i,j}}{T_j}\rfloor)^+\cdot C_j && (5.1)\\
& + \sum_{h\in\mathcal{P}_i/\{last_i,dest_i\}} (\max_{\substack{j\in[\![1,n]\!] \\ h\in\mathcal{P}_j}}(C_j)) && (5.2)\\
& +(|\mathcal{P}_i|-2)\cdot sl && (5.3)\\
& - \sum_{h\in\mathcal{P}_i/\{first_i,dest_i\}} (\Delta_{i,t}^h) && (5.4)\\
& -C_i && (5.5)
\end{aligned}
$$

- Term 5.1 is the delay generated by competing flows which delay τ_i along its trajectory as

well as the transmission delay generated by τ_i itself.

- Term 5.2 is the transition cost from one busy period to the next one.
- Term 5.3 is the switching latencies along the considered path.
- Term 5.4 considers the frame serialization effect.
- Term 5.5 is subtracted because $W_{i,t}^{last_i}$ is the latest starting time at $last_i$, but not the delay.

Then the delay upper bound of τ_i is computed by Equation 3.7, and recalled by Equation 5.6:

$$R_i = \max_{-J_i+\mathcal{B}_i \geq t \geq -J_i} \{W_{i,t}^{last_i} + C_i - t\} \tag{5.6}$$

Thus, the delay of the considered flow τ_i is composed of several parts. Term 5.3 gives a tight delay upper bound since the path length $|\mathcal{P}_i|$ and the switching latency sl are both constant. Term 5.5 is constant because C_i is the transmission time of the considered frame f_i. However, Term 5.1, Term 5.2 and Term 5.4 could introduce pessimism in the computation. This pessimism will be analyzed in this Chapter.

Besides the demonstration of potential pessimism introduced by the three terms of the classical Trajectory approach computation, a key issue is to evaluate this pessimism. Such an evaluation has been conducted in [BSF10], based on the generation of a reachable unfavorable scenario. The difference between the guaranteed delay upper bound and the delay experienced with the unfavorable scenario gives an estimation of the pessimism (maximum potential pessimism). Nevertheless, this unfavorable scenario must be established for each analyzed configuration. Therefore, a formula is proposed which underestimates the worst-case ETE delay. This underestimation is obtained by modifying pessimistic terms (Term 5.1, 5.2, 5.4) in the Trajectory approach computation. An analytical method for underestimated delays is developed based on these new terms which are never pessimistic (they can be optimistic). Therefore an analytical method for maximum potential pessimism is the discrepancy between the classical Trajectory approach and the analytical method for underestimated delays.

5.2.2 Pessimism analysis of computing flows

Overestimation of competing flows

For a flow τ_i following its path $\mathcal{P}_i$, its frame f_i generated at time t is considered. At each node along its path $\mathcal{P}_i$, there are competing flows from other input links which could delay the frame f_i. Thus, one part of delay is contributed by the frame transmissions of the competing flows, which corresponds to the Term 5.1.

Let us consider the Example 3 in Figure 5.1.

Example 3 *There are eleven flows $\tau_1, ..., \tau_{11}$ emitted by the source node N_1 and one flow τ_{12} emitted by the source node N_2. The flow τ_1 following the path $\mathcal{P}_1 = \{N_1, S_1, N_3\}$ is considered, and flow τ_{12} first enters the path $\mathcal{P}_1$ at the switch S_1, i.e., $first_{12,1} = S_1$. The parameters of the flows are as follows: $C_1 = ... = C_{11} = C_{12} = 10\ \mu s$, $T_1 = ... = T_{11} = 200\ \mu s$, $T_{12} = 100\ \mu s$, $J_1 = ... = J_{12} = 0$. The network supports FIFO scheduling and works at $R = 100\ Mbit/s$. The switching latency is $sl = 0$.*

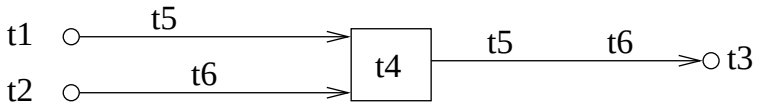


Figure 5.1: Example 3

The arrival time of a frame f_i of flow τ_i at the output port h is denoted $a^h_{f_i}$. The worst case scenario of a frame f_1 of flow τ_1 at the source node N_1 occurs when each competing flow τ_j ($j \in [\![2, 11]\!]$) has a frame arriving at the source node N_1 at the same time as the frame f_1 (*critical time*) and the frame f_1 is the last one transmitted. For the sake of simplicity, this arrival time is considered as the time origin, i.e., $t = a^{N_1}_{f_1} = ... = a^{N_1}_{f_{11}} = 0\ \mu s$. This scenario is depicted in Figure 5.2. Then $S^{S_1}_{max_1} = 110\ \mu s$ and $M^{S_1}_1 = 10\ \mu s$. Since τ_{12} is the only flow emitted by the source node N_2, $S^{S_1}_{max_{12}} = S^{S_1}_{min_{12}} = C_{12} = 10\ \mu s$.

According to the Equation 3.1 , the maximum workload interval of flow τ_{12} at node S_1 is:

$$\begin{aligned} A_{1,12} &= S^{S_1}_{max_1} - S^{S_1}_{min_{12}} - M^{S_1}_1 + S^{S_1}_{max_{12}} + J_{12} \\ &= 110 - 10 - 10 + 10 + 0 \\ &= 100\ \mu s \end{aligned}$$

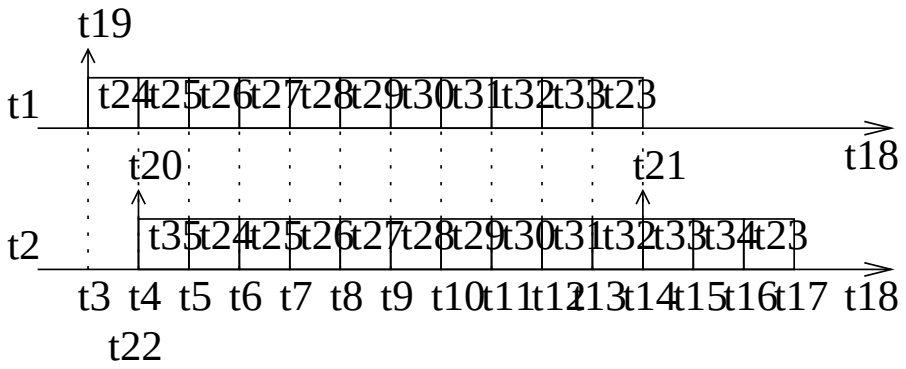


Figure 5.2: Worst-case scenario for τ_1 of the Example 3

Then Term 5.1 is equal to

$$(1 + \lfloor \frac{t + A_{1,12}}{T_{12}} \rfloor)^+ \cdot C_{12} = (1 + \lfloor \frac{t + 100}{100} \rfloor)^+ \cdot 10 = 20 \ \mu s$$

when $t = 0 \ \mu s$.

It means that there are at most two frames, denoted by f_{12} and f'_{12}, of flow τ_{12} delaying the studied frame f_1 at node S_1.

Specifically, the kth frame of flow τ_j is represented by $f_{j,k}$. The minimum temporal inter-arrival time between the arrival times of two consecutive frames $f_{j,k}$, $f_{j,k+1}$ from the same flow τ_j at the output port h is achieved when frame $f_{j,k}$ is delayed as much as possible and frame $f_{j,k+1}$ is not delayed by any other frame, as illustrated in Figure 5.3.

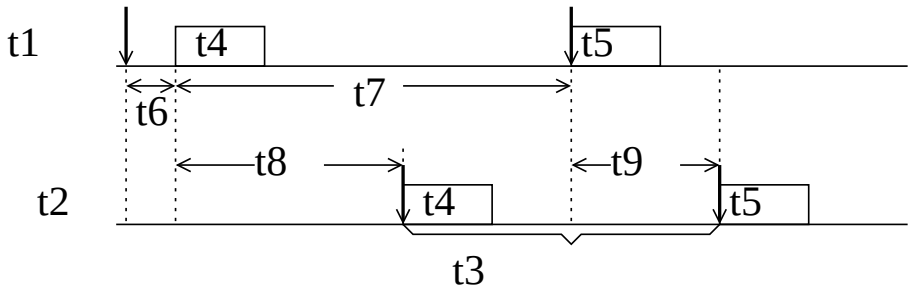


Figure 5.3: Illustration on minimum inter-arrival time between two frames of the same flow

Then the minimum inter-arrival time is:

$$T_j - S^h_{max_j} + S^h_{min_j} - J_j$$

Back to Example 3, the minimum inter-arrival time between f_{12} and $f_{12}^{'}$ at the node S_1 is $100 - 10 + 10 - 0 = 100\ \mu s$. In order to delay f_1, f_{12} and $f_{12}^{'}$ arrive at S_1 no earlier than $M_1^{S_1} = 10\ \mu s$ and no later than $t + S_{max_1}^{S_1} = 110\ \mu s$ with a minimum interval $100\ \mu s$ between their arrivals. Thus f_{12} and $f_{12}^{'}$ arrive at $110\ \mu s$ and $10\ \mu s$, respectively. This scenario is also depicted in Figure 5.2. The computed delay of flow τ_1 is then $140\ \mu s$.

Note that it is the interval lower bounded by $M_1^{S_1}$ and upper bounded by $t + S_{max_1}^{S_1}$ which determines the maximum number of frames of flows τ_{12} delaying the studied frame f_1. However, the length of this interval could be overestimated. Let us consider the Example 4 drawn in Figure 5.4.

Example 4 *Compared to the Example 3 in Figure 5.1, flows $\tau_2, ..., \tau_{10}$ have left the path $\mathcal{P}_1$ when the studied frame f_1 arrives at the node S_1. All the parameters of the network in the Example 4 are the same as flows in the Example 3.*

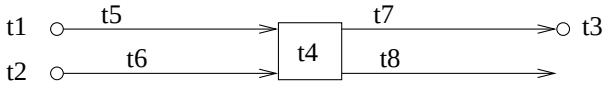


Figure 5.4: Example 4

The worst case scenario for the frame f_1 at the output port of source node N_1 is shown in Figure 5.5 (the same as that in the Example 3 in Figure 5.2), which gives $S_{max_1}^{S_1} = 110\ \mu s$ and $M_1^{S_1} = 10\ \mu s$. The interval $[M_1^{S_1}, t + S_{max_1}^{S_1}]$ decides that two frames f_{12} and $f_{12}^{'}$ of flow τ_{12} could arrive at the node S_1 during this interval. As analyzed in the previous paragraphs, these two frames arrive at $M_1^{S_1} = 10\ \mu s$ and $t + S_{max_1}^{S_1} = 110\ \mu s$ respectively, which is depicted in Figure 5.5. The computed delay of the frame f_1 is $140\ \mu s$ which is the same as that in Figure 5.5. However, since frames $f_2, ..., f_{10}$ have left the path $\mathcal{P}_1$ when frame f_1 arrives at the node S_1, only frame f_{12} arriving at $t + S_{max_1}^{S_1} = 110\ \mu s$ can delay frame f_1, resulting in the delay of flow τ_1 equal to $130\ \mu s$ (Figure 5.5) which is $10\ \mu s$ less than the computed delay $140\ \mu s$. This $10\ \mu s$ is exactly the transmission time of frame $f_{12}^{'}$ which actually does not delay frame f_1. It indicates that the computation considering two frames of flow τ_{12} introduces pessimism.

For both Example 3 and Example 4, the results of ETE delay computation for the flow τ_i are $140\ \mu s$, which is reachable in Example 3 but pessimistic in Example 4. The reason of the introduced pessimism is the idle time between the time $M_1^{S_1}$ and the time $t + S_{max_1}^{S_1}$ due to the workload that has left the considered path at the output port of S_1 in Example 4. This idle time results in the immediate transmission of frame $f_{12}^{'}$ just after its arrival at S_1 without

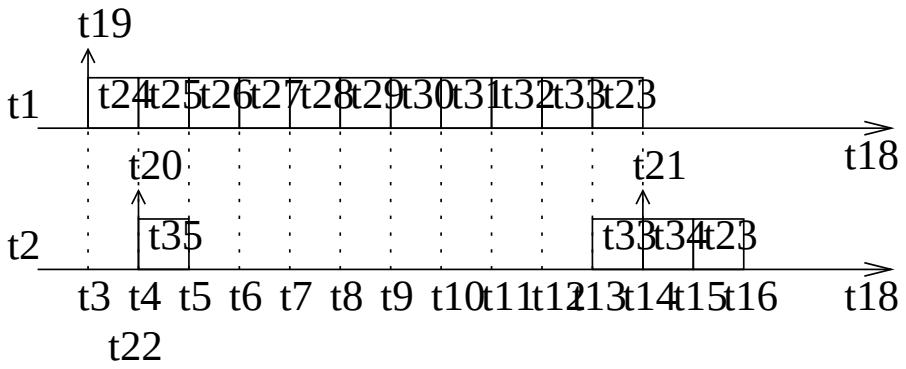


Figure 5.5: Worst-case scenario for τ_1 of the Example 4

delaying any frame in the busy period bp^{S_1}.

Maximum potential pessimism of competing flows

In order to evaluate the introduced pessimism of an overestimated delay, it is necessary to calculate an underestimated value of the delay generated by competing flows. We focus on a frame f_i of flow τ_i which is generated at time t. According to the Term 5.1, the pessimism comes from the overestimated number of frames for each competing flow. Thus the potential pessimism can be removed by considering that there is only one frame per flow. It can be optimistic. Then the lower bounded delay generated by competing flows is denoted $UR_{CF_i^{last_i}}$ and computed by:

$$UR_{CF_i^{last_i}} = \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} (C_j) \tag{5.7}$$

The upper bound of the pessimism caused by the computation of this delay is denoted by $P_{CF_{i,t}^{last_i}}$. It is the difference between Term 5.1 and Term 5.7, which is:

$$P_{CF_{i,t}^{last_i}} = \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} (\lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j \tag{5.8}$$

For Example 4 in Figure 5.4, a reachable delay of τ_1 computed by Term 5.7 is 130 μs which

is the exact worst-case delay of τ_1 in this example.

The maximum pessimism introduced by the classical Trajectory approach for τ_1 is computed by Term 5.8, which is 10 μs.

5.2.3 Pessimism analysis of busy period transition

Overestimation of busy period transition

Based on the classical Trajectory approach, the transmission time of the first frame of a busy period is counted twice as the busy period transition. In order to be safe, the largest frame crossing each output port of nodes is counted. This can be pessimistic. Let us consider the Example 5 depicted in Figure 5.6.

Example 5 *Flows τ_1 and τ_2 are emitted by the source node N_1, and flows τ_3 and τ_4 are emitted by the source node N_2. The parameters of the flows are as follows: $C_1 = C_2 = C_3 = C_4 =$ 100 μs, $T_1 = T_2 = T_3 = T_4 = 8000$ μs, $J_1 = J_2 = J_3 = J_4 = 0$. The network supports FIFO scheduling and works at $R = 100$ $Mbit/s$. The switching latency is $sl = 0$.*

Flow τ_1 following the path $\mathcal{P}_1 = \{N_1, S_1, N_3\}$ is focused on. Flows τ_3 and τ_4 first enter the path $\mathcal{P}_1$ at the node S_1, i.e., $first_{3,1} = first_{4,1} = S_1$.

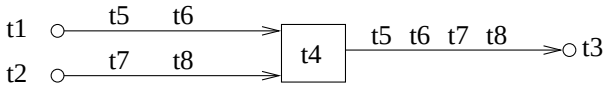


Figure 5.6: Sample network of Example 5, 6, 7

The worst-case ETE delay of flow τ_1 is computed using the classical Trajectory approach and it equals to 500 μs which corresponds to the worst-case scenario depicted in Figure 5.7. Note that in this scenario all the frames are counted once except the frame f_2 which is the first and largest frame coming from the busy period bp^{N_1} to the busy period bp^{S_1}. Indeed, the transmission time of the frame f_2 (marked as a shadow block in Figure 5.7) is the transition cost from the busy period bp^{N_1} to the busy period bp^{S_1} which corresponds to the Term 5.2.

However, Term 5.2 could introduce some pessimism in the computation when the first frame of the busy period is not the largest frame transmitted from the previous busy period. Let us illustrate this case on Example 6.

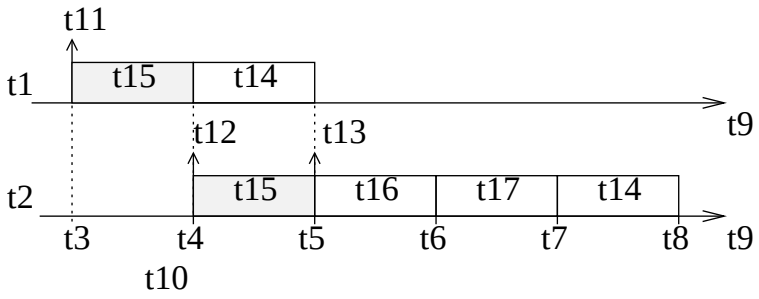


Figure 5.7: Worst-case scenario for τ_1 of the Example 5

Example 6 *This example has the same network architecture as the Example 5 in Figure 5.6, but smaller frame transmission time for the flow τ_2, i.e., $C_2 = 40\ \mu s$.*

According to the Trajectory approach, Term 5.2 is equal to

$$\max(C_1, C_2) = \max(100, 40) = 100\ \mu s$$

The latest starting time of a frame f_1 of flow τ_1 at node S_1 is computed as $W_{1,t}^{S_1} = 340\ \mu s$. Therefore, the ETE delay upper bound of the flow τ_1 is obtained by:

$$R_1 = \max_{-J_1+\mathcal{B}_1 \geq t \geq -J_1} (W_{1,t}^{S_1} + C_1 - t) = 440\ \mu s$$

when $t = 0$.

In this computation, the frame f_1 is counted twice since $C_1 > C_2$. However, this result is pessimistic.

The worst-case scenario of the frame f_1 at the source node N_1 is identified in Figure 5.8. Based on the computation, a frame f_3 of flow τ_3 and a frame f_4 of flow τ_4 can arrive at node S_1 no earlier than the time $M_1^{S_1} = 40\ \mu s$ and no later than the time $t + S_{max_1}^{S_1} = 140\ \mu s$. Since both f_3 and f_4 come from the same input link, their transmissions are constrained by the serialization. It means that the temporal interval between their arrivals is at least $\min(C_4, C_5) = 100\ \mu s$. Assume that f_3 arrives earlier than f_4 (it is the same if f_4 arrives earlier since f_3 and f_4 have exactly the same parameters). In order to guarantee the worst case, the arrival time of f_4 is delayed till the arrival of the considered frame f_1, which is $140\ \mu s$. Therefore, the latest arrival time of f_3 is $40\ \mu s$.

One possible scenario giving the worst-case delay in the node S_1 is that $a_{f_2}^{S_1} = a_{f_3}^{S_1}$ and $a_{f_1}^{S_1} = a_{f_4}^{S_1}$ as shown in Figure 5.8. In this scenario, frame f_2 which delays the frame f_1 is the first frame transmitted in the busy period bp^{S_1}. It means that the frame f_2 should be counted twice in the computation and counting twice the frame f_1 actually incurs pessimism. The tight ETE delay of the flow τ_1 is then 380 μs. Compared to the delay 440 μs computed by the classical Trajectory approach, there is a pessimism of 60 μs, which corresponds to the difference between the transmission times of C_1 and C_2.

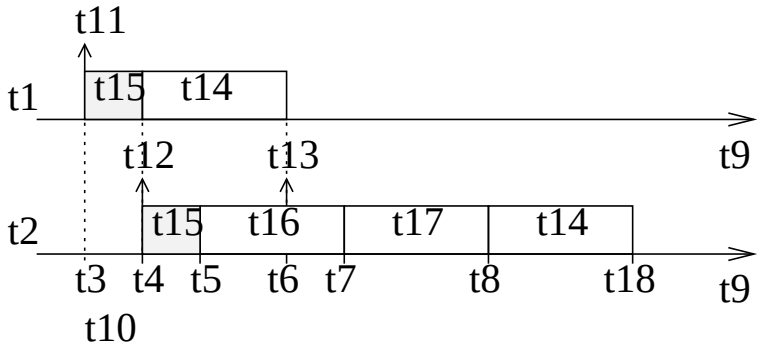


Figure 5.8: Worst-case scenario for τ_1 of the Example 6

The computation is pessimistic because the frame f_2 is not the largest frame transmitted from the busy period bp^{N_1}. It indicates that since the order of frames transmitted in the busy period is not arbitrary, the assumption that the first frame coming from the previous busy period is the largest one can bring some pessimism into the computation.

Maximum potential pessimism of busy period transition

Term 5.2 considers that first frame transmitted in the busy period bp^h is the largest frame coming from the previous busy period bp^{h-1} and it is counted twice as the busy period transition. As shown in Example 6 in Figure 5.8, Term 5.2 can introduce pessimism due to the non-arbitrary frame transmission order. It is sure that the transition cost is equal to or larger than the minimum frame transmitted from the previous busy period. Then the lower bounded delay of busy period transition is denoted $UR_{CT_i^{last_i}}$ and computed by:

$$UR_{CT_i^{last_i}} = \sum_{h \in \mathcal{P}_i / \{last_i, dest_i\}} \left(\min_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j) \right) \tag{5.9}$$

The upper bound of the pessimism caused by the twice counted frame is denoted $P_{CT_i^{last_i}}$. It is the difference between Term 5.2 and Term 5.9, which is:

$$P_{CT_i^{last_i}} = \sum_{h \in \mathcal{P}_i / \{last_i, dest_i\}} \left(\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j) - \min_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j) \right) \tag{5.10}$$

For Example 6 in Figure 5.6, a reachable delay of τ_1 computed by Term 5.9 is 380 μs which is the exact worst-case delay of τ_1 in this example.

The maximum pessimism introduced by the classical Trajectory approach for τ_1 is computed by Term 5.10, which is 60 μs.

5.2.4 Pessimism analysis of serialization effect

Underestimation of serialization effect

As an optimization in [BSF10], Term 5.4 considers the frame serialization of input link, which refers to the fact that the frames coming from the same input link cannot arrive at the output port at the same time due to the physical constraint. The flows crossing the output port h are coming from $k_h + 1$ input links. For the studied flow τ_i, at each visited output port h in the path $\mathcal{P}_i$, IP_0^h represents the input link of the flow τ_i. There are k_h others input links, denoted by IP_k^h ($k \in [\![1, k_h]\!]$). For each input link IP_k^h, there is a frame sequence seq_k^h composed of continuous frame transmissions of each competing flow from the input link IP_k^h. The duration of sequence seq_k^h without its first frame is denoted l_k^h ($k \in [\![0, k_h]\!]$).

According to [BSF10], the serialization factor $\Delta_{i,t}^h$ at the output port h is the maximum value between 0 and:

$$\max_{k \in [\![1,k_h]\!]} (\min(l_k^h)) - \max(l_0^h) \tag{5.11}$$

Details can be found in Appendix B.3.

The serialization effect is illustrated using the Example 7 which has the network architecture shown in Figure 5.6.

Example 7 *This example has the network architecture shown in Figure 5.6. The parameters*

of the flows are as follows: $C_1 = C_2 = 100\ \mu s$, $C_3 = C_4 = 40\ \mu s$ $T_1 = T_2 = T_3 = T_4 = 8000\ \mu s$, $J_1 = J_2 = J_3 = J_4 = 0$. *The network supports FIFO scheduling and works at* $R = 100\ Mbit/s$. *The switching latency is* $sl = 0$.

Compared to Example 6, the transmission times of flows τ_3 and τ_4 are reduced to 40 μs.

The ETE delay upper bound of the flow τ_1 computed by the Trajectory approach is 320 μs. The computation of $\Delta_{1,t}^{S_1}$ is illustrated in Figure 5.9. According to [BSF10], $\Delta_{1,t}^{S_1}$ is calculated by:

$$\begin{aligned}\Delta_{1,t}^{S_1} &= \max(0, \min(l_1^{S_1}) - \max(l_0^{S_1})) \\ &= \max(0, C_4 - C_1) \\ &= \max(0, -60) \\ &= 0\end{aligned}$$

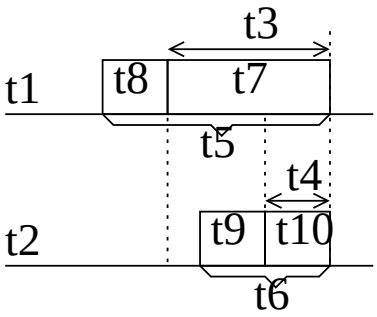


Figure 5.9: Illustration of $\Delta_{1,t}^{S_1}$

The worst-case scenario of frame f_1 at the source node N_1 is identified in Figure 5.10. When f_1 arrives at the output port of S_1, the interval during which the other competing frames can delay f_1 is $[M_1^{S_1}, a_{f_1}^{S_1}] = [40, 140]$. Since frames f_3 and f_4 are transmitted by the same link from N_2 to S_1, they are serialized which means that their inter-arrival time at S_1 is at least one frame transmission time 40 μs. Suppose that frame f_4 arrives at the output port of S_1 at the same time as frame f_1, i.e., $a_{f_4}^{S_1} = a_{f_1}^{S_1} = 140\ \mu s$, and frame f_3 is transmitted before frame f_4. Thus in order to delay f_1, f_3 arrives at S_1 no earlier than $t = 40\ \mu s$ and no later than $t = 100\ \mu s$ due to the serialization.

If f_3 arrives as early as at time $t = 40\ \mu s$ (the same arrival time as f_2), the transmission

of these two frames is finished at time $40 + C_2 + C_3 = 120\ \mu s$ which is before the arrival of f_1. Thus f_3 does not delay f_1. This case is illustrated in the chronogram of S_1 (a) in Figure 5.10. If f_3 arrives as late as at time $t = 100\ \mu s$, the transmission of f_3 is finished at time 140 μs before the arrival of f_1, and f_3 does not delay f_1. This case is depicted in the chronogram of S_1 (b) in Figure 5.10. It indicates that f_3 cannot delay frame f_1 at the output port of S_1 even if it arrives during the time interval $[40, 100]$. Since the flows τ_3 and τ_4 have the same parameters, it is the same case if f_4 arrives before f_3.

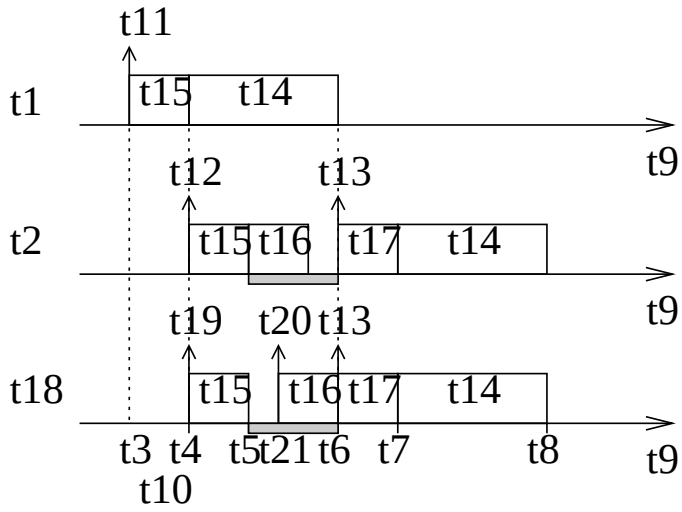


Figure 5.10: Worst-case scenario for τ_1 of the Example 7

Frame f_3 does not delay frame f_1 because it has been transmitted during the idle time generated between the arrivals of frames f_2 and f_1 (indicated by a shadow block in Figure 5.10). This idle time can not be ignored due to the different transmission times of the serialized frames f_1 and f_2. Indeed, the busy period bp^{S_1} starts at time $a^{S_1}_{f_1}$ instead of the time $a^{S_1}_{f_2}$, which is different from the case presented in Figure 5.8 where the busy period bp^{S_1} starts at time $a^{S_1}_{f_2} = M^{S_1}_1$. Then in Example 7, if frame f_2 is still considered in the frame sequence $\text{seq}^{S_1}_0$, it leads to an overestimated workload of $\text{seq}^{S_1}_0$. Therefore, f_2 should not be counted in $\text{seq}^{S_1}_0$ in Example 7, which results in $l^{S_1}_0 = 0$ and the serialization factor is computed by:

$$\Delta^{S_1}_{1,t} = \min(l^{S_1}_1) - \max(l^{S_1}_0) = C_4 - 0 = 40\ \mu s$$

Then ETE delay upper bound of the flow τ_1 is $320 - 40 = 280\ \mu s$ as shown in Figure 5.10.

Comparing Example 7 with Example 6, frame f_2 is a part of $\text{seq}^{S_1}_0$ in Example 6 in Figure 5.8, while it does not belong to $\text{seq}^{S_1}_0$ in Example 7 in Figure 5.10. The pessimistic

estimation in Example 7 is due to the idle time at the output port of S_1 which separates f_2 from the frame sequence $\text{seq}_0^{S_1}$.

Maximum potential pessimism of serialization effect

As illustrated in Example 7, the pessimism of serialization factor is introduced due to the idle time at the output port. In order to avoid the pessimism, one way is to build a reachable scenario at each output port where the frame sequence of each input link will not lead to idle time at the output port. Then a reachable serialization factor is computed based on this scenario. In order to achieve that, first we will show how to build a continuous frame sequence at each input link. Actually, the way to build continuous frame sequences of input links of competing flows and the way to build a continuous frame sequence of the input link of the considered flow are different. They will be presented respectively. Then, the computation of the reachable serialization factor is formalized. The difference between the reachable serialization factor and the one of Term 5.4 leads to the maximum potential pessimism.

We first consider a reachable input frame sequence seq_k^h ($k \in [\![1, k_h]\!]$) of competing flows crossing the output port h. Since we consider one frame per flow, one reachable continuous frame sequence seq_k^h is composed of one frame of each competing flow transmitted from the input link IP_k^h with the largest one at first of the sequence. It is proved by Lemma 2, which is given later, that a continuous frame sequence with the largest frame transmitted first on the input link is still continuous when the frames arrive at the output link. Therefore, such a frame sequence will not lead to idle time at the output port h. Thus, the computation $\max_{k\in[\![1,k_h]\!]}(\min(l_k^h))$ in Formula 5.11 is reachable.

Lemma 2 *Given a continuous sequence of frames $f_1, f_2, ..., f_n$ coming from a single input link in and transmitted on a single output link out, this frame sequence is still continuous on the output link out when f_1 is the largest frame of the sequence.*

proof: A continuous frame sequence $f_1, f_2, ..., f_n$ transmitted on an input link *in* is illustrated in the upper part in Figure 5.11. Let us assume that frame f_1 is the largest frame of the sequence. Let us also assume that a frame f_i in the sequence arrives at the output link *out* as soon as its transmission is finished on the input link *in* as illustrated by the dashed lines in Figure 5.11. This arrival time is denoted $a_{f_i}^{out}$.

Without loss of generality, when the frame sequence arrives at the output link *out*, we assume that there is an idle time t_{idle} between frame f_{m-1} and frame f_m. Before the idle time

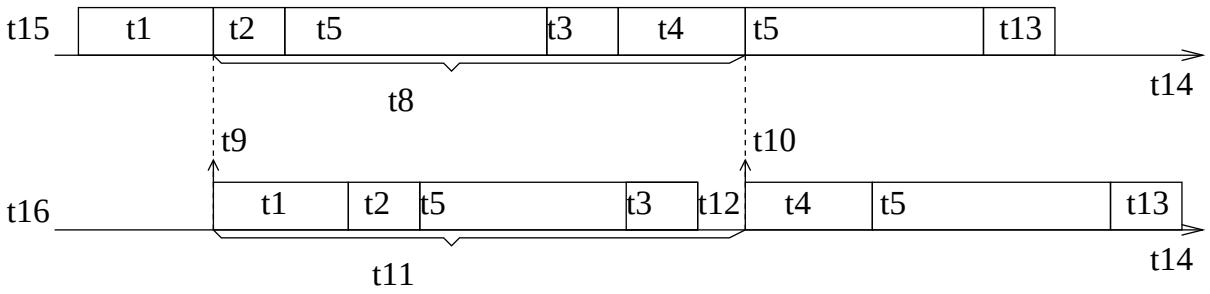


Figure 5.11: Illustration of **Lemma 2**

t_{idle}, the frame sequence from f_1 to f_{m-1} is continuous. This case is depicted in lower part in Figure 5.11.

On the input link *in*, since the sequence is continuous, we have:

$$a_{f_m}^{out} = a_{f_1}^{out} + \sum_{i \in [\![2,m]\!]} C_i$$

On the output link *out*, since the sequence is continuous before t_{idle}, we have:

$$a_{f_m}^{out} = a_{f_1}^{out} + (\sum_{i \in [\![1,m-1]\!]} C_i + t_{idle})$$

Consequently,

$$\sum_{i \in \in [\![2,m]\!]} C_i = \sum_{i \in \in [\![1,m-1]\!]} C_i + t_{idle}$$

Then we have:

$$C_m = C_1 + t_{idle}$$

Since f_1 is the largest frame in the sequence, $C_m \leq C_1$. Then we have:

$$t_{idle} \leq 0$$

Therefore, there is no idle time during the frame sequence on the output link *out*. Thus, we can conclude that the frame sequence is continuous on the output link *out* when f_1 is the largest frame of the sequence.■

A reachable continuous frame sequence seq_0^h is composed of frames transmitted from input

links of previous output port $h-1$, as illustrated in Figure 5.12. We consider a frame f_i of flow τ_i transmitted from the input link IP_0^{h-1} to the output port $h-1$ and then transmitted from the input link IP_0^h to the output port h. Figure 5.13 illustrates how to build a reachable frame sequence seq_0^h for frame f_i. $a_{f_i}^h$ is the arrival time of frame f_i at the output port h.

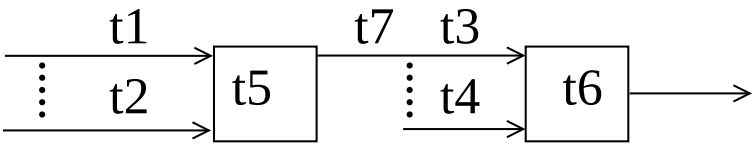


Figure 5.12: Illustration of seq_0^h

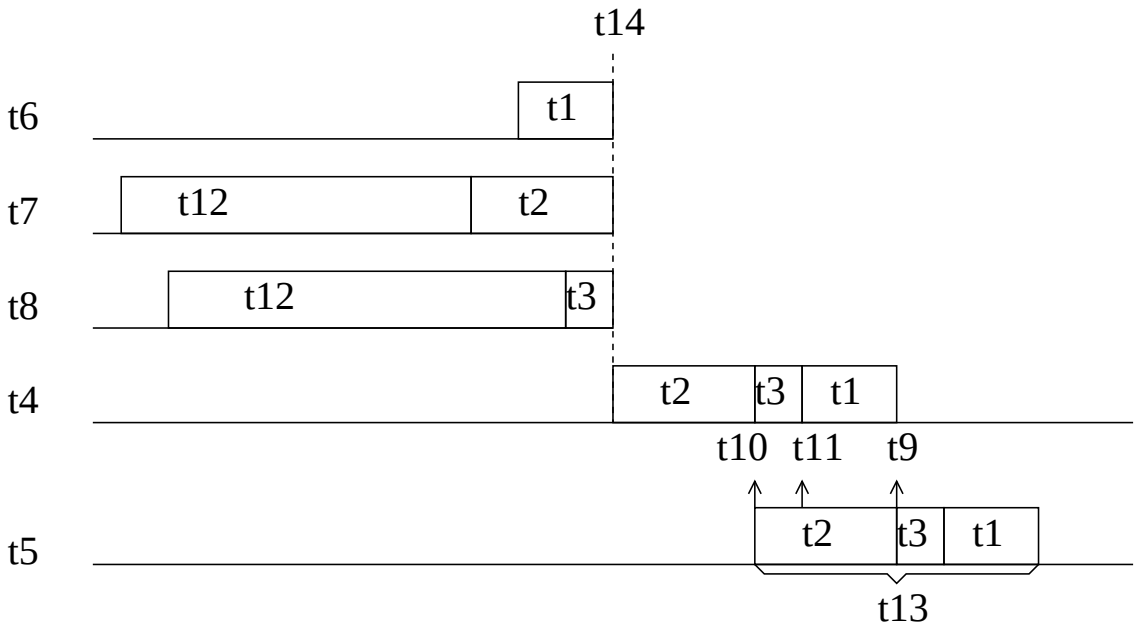


Figure 5.13: Illustration of a reachable seq_0^h: Case 1

At the output port $h-1$, there are k_{h-1} input links transmitting competing flows. For each input link IP_k^{h-1} ($k \in [\![1, k_{h-1}]\!]$), there is at least one competing frame g_k^{h-1} which can arrive at $h-1$ at the same time as f_i. We consider the smallest frame of each input link. Then at the output port $h-1$, we can build a continuous frame sequence composed of g_k^{h-1} ($k \in [\![1, k_{h-1}]\!]$) and f_i. The largest frame among g_k^{h-1} is transmitted first. In Figure 5.13, the largest frame is $f_{\mathrm{IP}_1^{h-1}}$. This frame sequence is the input frame sequence of the output port h.

Two cases are taken into account. The first one is that the first transmitted frame is larger than frame f_i, as illustrated in Figure 5.13. According to Lemma 2, when the frame sequence

arrives at the output port h, it is still continuous. Then this frame sequence is a reachable frame sequence seq_0^h. It includes the smallest frame of each competing input link of $h-1$, which is g_k^{h-1} ($k \in [\![1, k_{h-1}]\!]$), and frame f_i. The transmission time of frame g_k^{h-1} is denoted C_k^{h-1} ($k \in [\![1, k_{h-1}]\!]$). Then the length of the reachable frame sequence without its first frame (the largest one among g_k^{h-1}) is denoted rl_0^h and it is computed by:

$$\min(rl_0^h) = \sum_{k \in [\![1, k_{h-1}]\!]} (C_k^{h-1}) + C_i - \max_{k \in [\![1, k_{h-1}]\!]} (C_k^{h-1}) \tag{5.12}$$

The second case is that the first transmitted frame of the frame sequence is smaller than frame f_i. This case is illustrated in Figure 5.14. In this case, when the frame sequence arrives at the output port h, there can exist idle time, as indicated by the shadow block in Figure 5.14. Therefore, a reachable frame sequence seq_0^h includes only f_i itself, and therefore $rl_0^h = 0$.

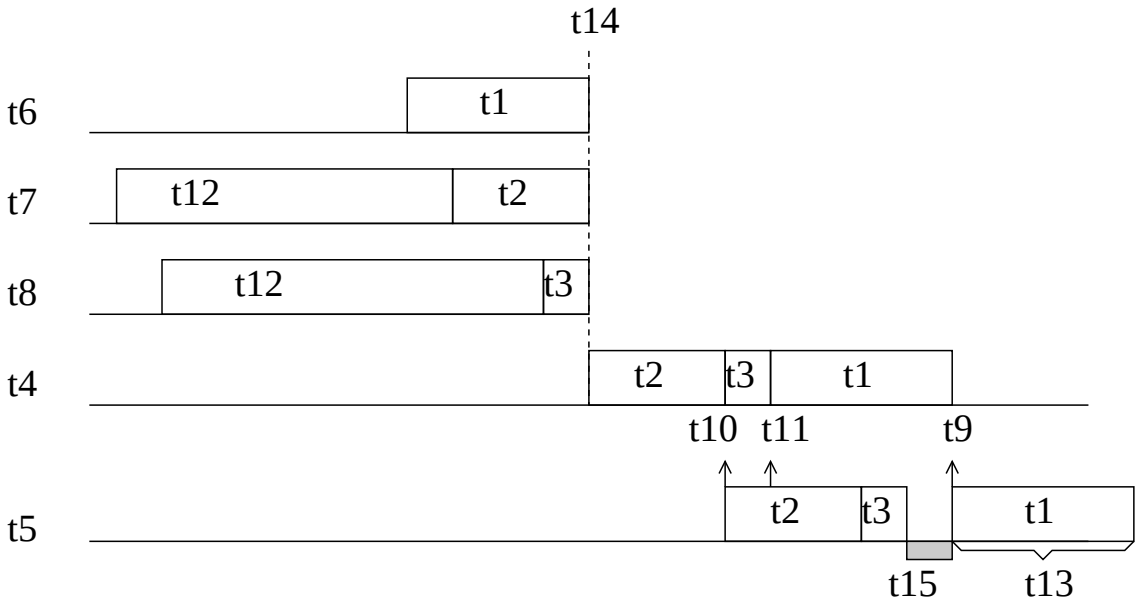


Figure 5.14: Illustration of a reachable seq_0^h: Case 2

As we have derived a reachable frame sequence length without its first frame for each input link, a reachable frame serialization factor is computed by:

$$\Delta_i^h = \max(0, \max_{k \in [\![1, k_h]\!]} (\min(l_k^h)) - \min(rl_0^h)) \tag{5.13}$$

A reachable value of serialization factor is denoted $UR_{S_i^{last_i}}$ and computed by:

$$UR_{S_i^{last_i}} = - \sum_{h \in \mathcal{P}_i / \{first_i, dest_i\}} (\Delta_i^h) \tag{5.14}$$

Therefore the pessimism introduced by the computation of frame serialization (Term 5.4) is denoted $P_{S_{i,t}^{last_i}}$ and upper bounded by:

$$P_{S_{i,t}^{last_i}} = \sum_{h \in \mathcal{P}_i / \{first_i, dest_i\}} (\Delta_i^h - \Delta_{i,t}^h)$$

Consider again the Example 7 in Figure 5.6. For $\text{IP}_0^{N_1}$, the frame f_2 is smaller than the studied frame f_1, i.e., $C_2 < C_1$. Then the frame f_2 does not contribute to the $\text{seq}_0^{S_1}$. Therefore when $t = 0$ we have:

$$\min(rl_0^{S_1}) = 0,\ \Delta_{1,t}^{N_1} = \Delta_1^{N_1} = 0,\ \Delta_{1,t}^{S_1} = 0,\ \Delta_1^{S_1} = 40\ \mu s$$

It leads to the maximum introduced pessimism of serialization factor:

$$\begin{aligned} P_{S_{1,t}^{S_1}} &= (\Delta_1^{N_1} - \Delta_{1,t}^{N_1}) + (\Delta_1^{S_1} - \Delta_{1,t}^{S_1}) \\ &= 0 - 0 + 40 - 0 \\ &= 40\ \mu s \end{aligned}$$

5.2.5 Analytical method for underestimated delays

The underestimation of the worst-case ETE delay of flow τ_i using the classical Trajectory approach is denoted UR_i, which is the sum of $UR_{CF_i^{last_i}}$, $UR_{CT_i^{last_i}}$ and $UR_{S_i^{last_i}}$:

$$UR_i = UR_{CF_i^{last_i}} + UR_{CT_i^{last_i}} + UR_{S_i^{last_i}} \tag{5.15}$$

5.2.6 Analytical method for maximum potential pessimism

The upper bound of the pessimism of the worst-case end-to-end delay computation of the flow τ_i using the Trajectory approach is denoted UP_i, which is the maximum value of the sum of

$P_{CF_{i,t}^{last_i}}$, $P_{CT_i^{last_i}}$ and $P_{S_{i,t}^{last_i}}$:

$$UP_i = \max_{-J_i+\mathcal{B}_i \geq t \geq -J_i} (P_{CF_{i,t}^{last_i}} + P_{CT_i^{last_i}} + P_{S_{i,t}^{last_i}}) \tag{5.16}$$

5.3 Integration of the minimum duration constraints

It has been shown in Section 1.6 that for two periodic flows with known offsets emitted by the same source node, there is a minimum duration between them. In Chapter 3, a modified Trajectory approach has been proposed in order to take into account these dependencies. This proposed method introduces pessimistic computation so as to guarantee the delay upper bounds (overestimated delay). One way to evaluate this pessimism is to consider the gap between an upper bounded delay (overestimated delay) and an underestimated value of worst-case delay. In previous section, a pessimism analysis has been developed based on the classical Trajectory approach. This analysis provides a computation of underestimated worst-case delays. However, it does not consider the minimum duration constraints. Thus it cannot be applied to the flows with minimum duration constraints since it might provide a worst-case delay lower bound which is not reachable.

For example, flow τ_1 and flow τ_2 are emitted by the same source node. They are periodic with periods $T_1 = T_2 = 4000\ \mu s$. The frame transmission times are $C_1 = C_2 = 40\ \mu s$ and the release jitters are null. Flow τ_1 has an offset $O_1 = 0$ and flow τ_2 has an offset $O_2 = 1000\ \mu s$. Therefore, according to Section 1.6, there is a minimum duration of 1000 μs from a frame arrival of τ_1 to a frame arrival of τ_2, and the duration is 3000 μs in the reverse order. This is illustrated in Figure 5.15. Thus a frame of τ_1 will never be delayed by a frame of τ_2 and a reachable delay of τ_1 is $C_1 = 40\ \mu s$. According to the pessimism analysis in the previous section, one frame per flow is taken into account and frames of different flows can arrive at the same time since there is no assumptions on their frame arrivals. Thus a reachable delay for τ_1 is $C_1 + C_2 = 80\ \mu s$ which corresponds to a scenario where a frame of τ_2 delays a frame of τ_1. However due to the minimum duration constraint, this scenario will never happen and this delay is not reachable for flow τ_1.

The following paragraphs present how to compute an underestimated delay considering the minimum duration constraints based on the modified Trajectory approach developed in Chapter 3.

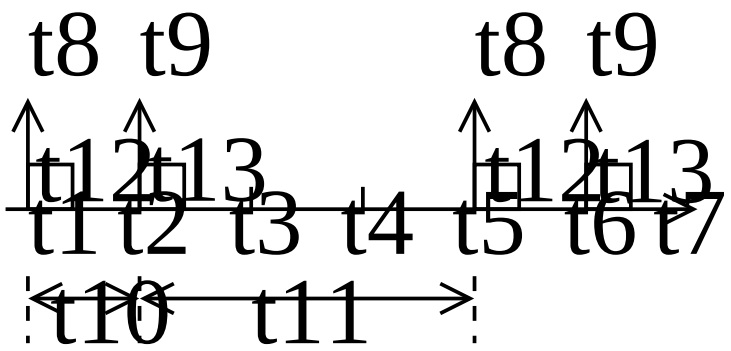


Figure 5.15: Illustration of a reachable delay of τ_1

5.3.1 Review of the modified Trajectory approach

The classical approach considers that each flow is independent. The modified approach developed in Chapter 3 classifies flows into subsets $\mathcal{G}_x$ ($x \in [\![1, n_g]\!]$). Any couple of flows in one subset has minimum duration constraints. The maximum workload generated by the subset $\mathcal{G}_x$ to delay flow τ_i is $RS_{i,t,x}$. Therefore, in order to obtain an underestimated worst-case delay, we have to compute an underestimated value of $RS_{i,t,x}$ for each subset $\mathcal{G}_x$.

In the following paragraphs, the pessimism introduced by the overestimated workload of subsets is illustrated. Then an underestimated value of the maximum workload of each subset is developed and the pessimism is computed by the gap of the underestimated value to the overestimated value.

5.3.2 Illustration on the overestimation of dependent flows

Let us first illustrate the pessimism introduced by the overestimated workload of subsets. Consider a frame f_1 of flow τ_1 in Example 8 in Figure 5.16.

Example 8 *All six flows in the example are periodic with known offsets. They have the same frame transmission time and the same period which are* $C_i = 500\ \mu s$ *and* $T_i = 8000\ \mu s$ $i \in [\![1, 6]\!]$. *Their offsets are* $O_1 = O_2 = O_3 = O_4 = 0$, $O_5 = 500\ \mu s$ *and* $O_6 = 2000\ \mu s$. *For all the flows, the release jitter is null, i.e.,* $J_i = 0$. *The network supports FIFO scheduling and works at* $R = 100\ Mbit/s$. *The switching latency is* $sl = 0$.

There are three subsets: $\mathcal{G}_1 = \{1, 2\}$, $\mathcal{G}_2 = \{3, 4\}$ and $\mathcal{G}_3 = \{5, 6\}$. Since $O_1 = O_2 = O_3 = O_4 = 0$, flows τ_1 and τ_2 as well as flows τ_3 and τ_4 have null minimum durations. Flows τ_5

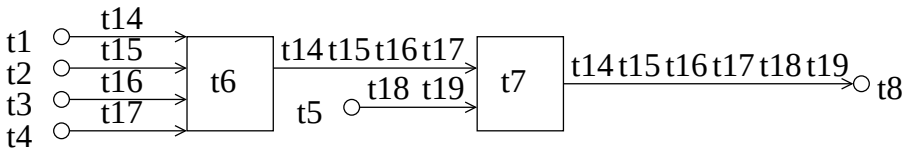


Figure 5.16: Example 8

and τ_6 are emitted at source node N_5. They delay flow τ_1 at the output port of S_2. Since $O_5 = 500\ \mu s$ and $O_6 = 2000\ \mu s$, there is a minimum duration $MD_{5,6}^{N_5} = 1500\ \mu s$ as shown in Figure 5.17.

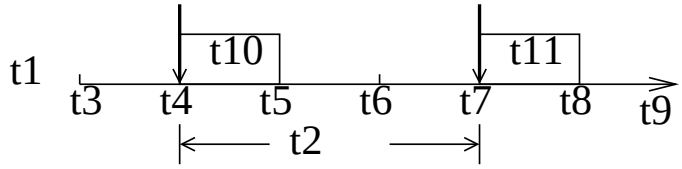


Figure 5.17: Illustration on minimum duration $MD_{5,6}^{N_5}$

Since the frames of τ_1, τ_2, τ_3 and τ_4 are released at the same time, at the output port of switch S_1, frame f_1 is delayed by frame f_2, and at the output port of switch S_1, frame f_1 is delayed by frames f_3 and f_4. At the output port of S_2, any frame of τ_5 and τ_6 can delay f_1 if it arrives later than $M_1^{S_2} = 1000\ \mu s$ and earlier than the arrival of f_1: $a_{f_1}^{S_2} = 2500\ \mu s$. The length of this workload interval at S_2 is 1500 μs which happens to be equal to the minimum duration $MD_{5,6}^{N_1}$. It means that both a frame f_5 from τ_5 and a frame f_6 from τ_6 can delay f_1, as shown in Figure 5.18. The worst-case delay computed by the modified Trajectory approach is 4000 μs which is the exact worst-case delay in this example. The worst-case scenario of f_1 is illustrated in Figure 5.18.

However, sometimes due to the overestimation of workload interval, the workload of a subset could be overestimated in the computation. Consider a frame f_1 of flow τ_1 in Example 9 in Figure 5.19, which is similar to Example 8 in Figure 5.16.

Example 9 *Compared to the Example 8 in Figure 5.16, flows τ_3 and τ_4 have left flow τ_1 at the output port of switch S_2 where the interference set of flows τ_5 and τ_6 joins flow τ_1. All the parameters of the network in the Example 9 are the same as in the Example 8 in Figure 5.16.*

Since the frame transmissions of flows τ_1, τ_2, τ_3 and τ_4 at the output ports of N_1 and S_1 remain the same as in Example 8, we have the same $M_1^{S_2} = 1000\ \mu s$ and the same $a_{f_1}^{S_2} = 2500\ \mu s$.

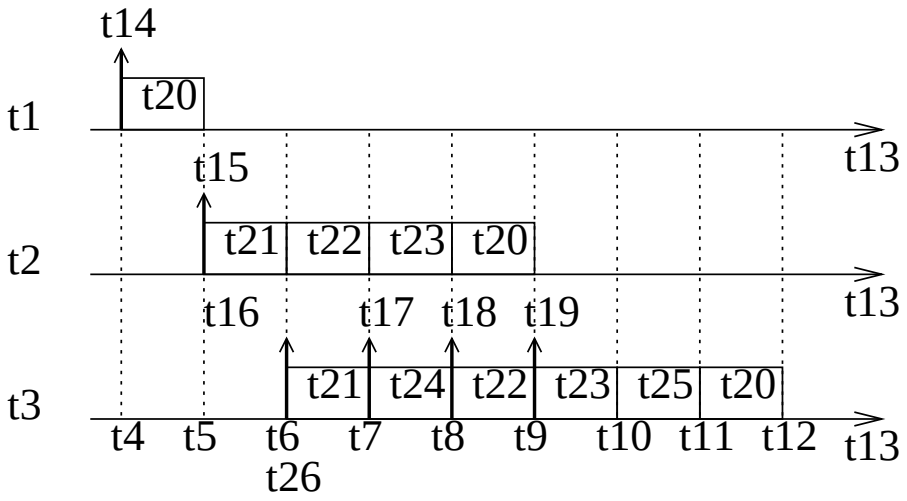


Figure 5.18: Worst case scenario for τ_1 in Example 8

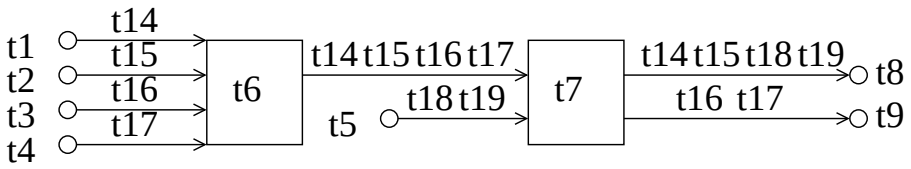


Figure 5.19: Example 9

Therefore, the computation considers the same workload interval 1500 μs at S_2, which equals to the minimum duration $MD_{5,6}^{N_1}$. Thus it leads to the same computed delay 4000 μs of flow τ_1.

The worst-case scenario for τ_1 in Figure 5.19 is illustrated in Figure 5.20 which indicates that the exact worst-case delay is 3500 μs. Compared to the computed delay 4000 μs, there is an introduced pessimism of 500 μs. Indeed, in order to delay f_1, the frame f_6 arrives at the same time as f_1 at the output port of S_2 ($a_{f_1}^{S_2} = a_{f_6}^{S_2} = 2500\ \mu s$). Due to the minimum duration $MD_{5,6}^{N_1} = 1500\ \mu s$ (shadow block in Figure 5.20), the latest possible arrival time of f_5 before f_6 is 1000 μs. As illustrated in Figure 5.20, f_5 is transmitted immediately after its arrival at S_2 and cannot delay f_1. Then only f_6 delays f_1 at S_2, and τ_5 does not delay τ_1 in this example. In this case, the workload of the subset $\mathcal{G}_3 = \{5, 6\}$ is overestimated.

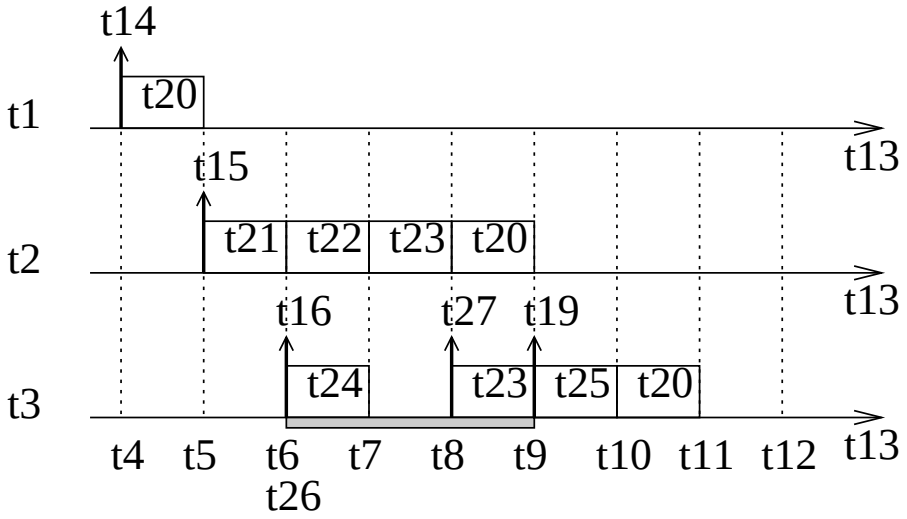


Figure 5.20: Worst case scenario for τ_1 in Example 9

In order to evaluate the introduced pessimism of the workload for each subset, in the following paragraphs we develop first a method to compute an underestimated value for each subset and then the pessimism which is the difference between the overestimated value and the underestimated value.

5.3.3 Pessimism analysis of the workload of dependent flows

It has been shown that the introduced pessimism is due to an overestimated workload of one subset. One way to obtain an underestimated worst-case delay for each subset is to consider the following assumptions:

- *Assumption 1*: only one frame per competing flow delays the considered frame;
- *Assumption 2*: all the frames generated by a subset are totally separated.

Assumption 1 is obviously optimistic by considering for each competing flow an infinite period. *Assumption 2* means that only one frame out of all the frames of one subset delays the considered frame and the largest frame is taken into account. It indicates that the minimum duration between any two flows is infinite and it is obviously optimistic. Thus, these two assumptions lead to the case where only one frame of flows in one subset can delay the considered frame, and the largest frame is considered in the computation. Back to the Example 9 in Figure 5.19, since frames f_5 and f_6 are both from one subset, then only the larger one between f_5 and f_6 is considered to delay frame f_1 at the output port of S_2.

Based on the two assumptions, an underestimated workload $UR_{CF_i^{last_i}}$ generated by n_g subsets delaying τ_i is obtained by:

$$UR_{CF_i^{last_i}} = \sum_{x \in [\![1, n_g]\!]} \max_{j \in \mathcal{G}_x}(C_j) \tag{5.17}$$

The upper bound of this pessimism is denoted $P_{CF_{i,t}^{last_i}}$. It is the difference between Term 3.10 and Term 5.17, which is:

$$P_{CF_{i,t}^{last_i}} = \sum_{x \in [\![1, n_g]\!]} (RS_{i,t,x} - \max_{j \in \mathcal{G}_x}(C_j))$$

Then the underestimation of the worst-case end-to-end delay upper bound of flow τ_i is computed by Equation 5.15 and the maximum introduced pessimism is upper bounded by Equation 5.16.

5.4 Application on the AFDX network

An application of the proposed approach on the industrial Avionic Full DupleX (AFDX) switched Ethernet network configuration (see 4.2) is presented. The modified Network Calculus approach and the modified Trajectory approach have been applied to this industrial AFDX network in Chapter 4. In this part, the introduced pessimism by both the classical and the

modified Network Calculus approach as well as the classical and the modified Trajectory approach on such a network is computed by the analytical method developed in this Chapter. The offsets are assigned according to the real-time automotive CAN network [GHN08] presented in 4.3.

5.4.1 Upper bounding the pessimism of the Network Calculus approach

Although the pessimism analysis is not directly developed from the Network Calculus approach, it does provide an analytical computation of the underestimated worst-case delays (lower bounds). The formulas are given in Section 5.2.5 and Section 5.3.3. Therefore, by measuring the difference between a worst-case delay upper bound computed by the Network Calculus approach and an underestimated value of the worst-case delay given by pessimism analysis, we can upper bound the pessimism introduced by the Network Calculus approach in the worst-case delay upper bound computation. The percentages of pessimism upper bound out of the overestimated delay (worst-case delay upper bound) of each path $\mathcal{P}_i$ of VL v_i for both classical Network Calculus approach and modified Network Calculus approach are calculated. The percentage is computed by the following term:

$$\frac{R_i - RD_i}{R_i}\%$$

where R_i is the worst-case delay upper bound of v_i computed by the classical/modified Network Calculus approach and RD_i is the underestimated value of the worst-case delay of v_i obtained by the analytical pessimism approaches.

The results are shown in Figure 5.21 where NC stands for the Network Calculus approach.

From these results, we deduce that the average percentages of pessimism for the classical Network Calculus approach and for the modified Network Calculus approach are of 18% and 17%, respectively.

5.4.2 Upper bounding the pessimism of the Trajectory approach

In order to evaluate the pessimism introduced by the Trajectory approach, a pessimism upper bound of each path is computed by the analytical method developed in this Chapter. The percentages of pessimism upper bounds out of the overestimated delays (upper bounds) of each path for both classical Trajectory approach and modified Trajectory approach are calculated.

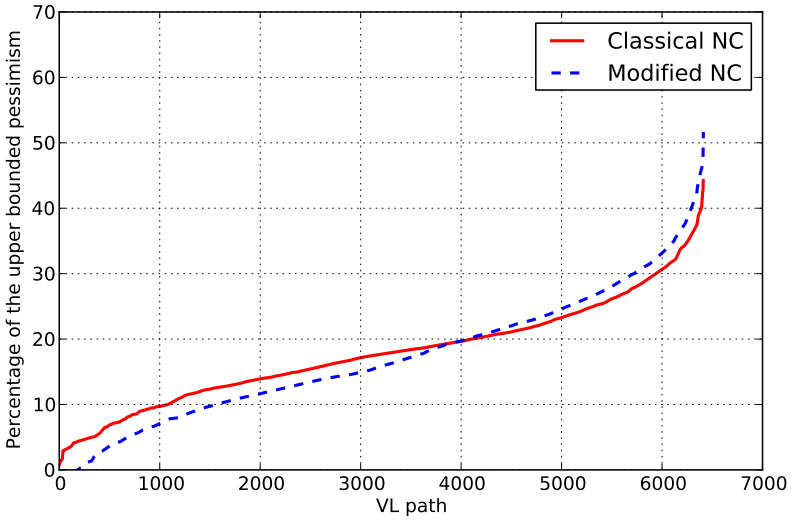


Figure 5.21: Upper bounded pessimism of the Network Calculus approach

The results are shown in Figure 5.22 where $Traj$ stands for the Trajectory approach.

The average percentages of pessimism for the classical Trajectory approach and for the modified Trajectory approach are of 12% and 14%, respectively. The Trajectory approach introduces less pessimism than the Network Calculus approach for both the classical and the modified cases. The reason is obvious since it has been shown in Chapter 4 that the Trajectory approach gives tighter computed worst-case delay upper bounds than the Network Calculus approach.

It is noticed that the classical Trajectory approach has slightly lower pessimism upper bounds than the modified Trajectory approach. This is due to *Assumption 2* in 5.3.3. This assumption considers that all the frames generated by one subset are totally separated and it takes the maximum frame out of a set of dependent flows in the computation instead of the maximum frame out of each flow. This assumption introduces more optimism compared to the classical approach. Therefore the pessimism analysis for the modified Trajectory approach leads to a lower (more optimistic) underestimated value of the worst-case delay and therefore a larger estimated maximum pessimism.

In Chapter 4, an optimized approach *Opt* is proposed which considers always the tighter computed ETE delay upper bounds between the modified Network Calculus approach and the modified Trajectory approach. The percentages of upper bounded pessimism of the optimized approach *Opt* are as well evaluated and its results are shown in Figure 5.22. The optimized

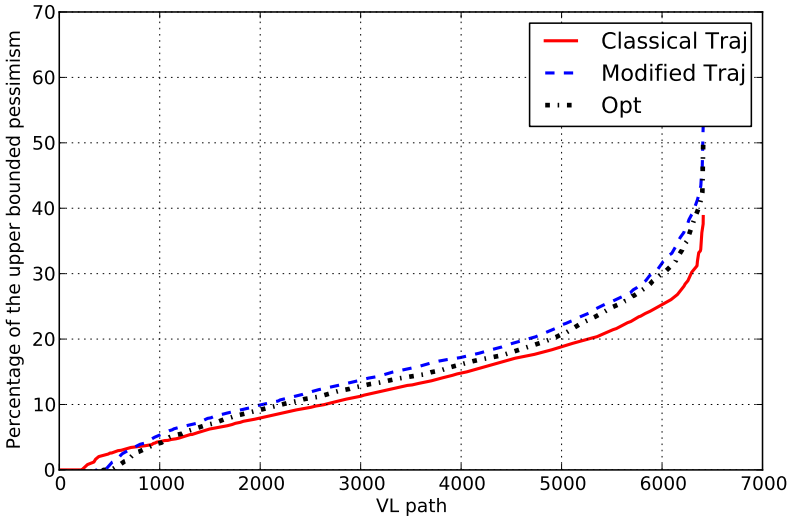


Figure 5.22: Upper bounded pessimism of the Trajectory approach

approach has a slight improvement on upper bounded pessimism (average percentage of 13%) than modified Trajectory approach because it takes tighter delay upper bounds of both the modified Network Calculus and the modified Trajectory approach, and therefore further eliminates some pessimism.

For several paths, the pessimism percentage is 0%. It means that for each of these flows, the computed delay upper bound is equal to its corresponding underestimated worst-case delay, which is the exact worst-case delay of the flow. The numbers of paths with exact worst-case delays provided by each evaluated approach and their percentages to the total number of paths (6412) are listed in Table 5.1.

	Classical NC	Modified NC	Classical Traj	Modified Traj	Opt
Nb of paths	0	194	234	459	523
% of paths	0	3.03	3.65	7.16	8.16

Table 5.1: Number and percentage of paths with exact worst-case delays

5.5 Conclusion

In the context of real-time switched Ethernet networks, the worst-case delay analysis introduces a pessimistic computation so as to guarantee an end-to-end delay upper bound. In order to evaluate the potential pessimism, a pessimism analysis is developed based on the Trajectory

approach. It first analyzes each source of possible pessimism. Then an underestimation of worst-case delay is developed. The maximum difference between the underestimation (lower bound) and the overestimation (upper bound) of the end-to-end delay gives a metric of the maximum potential pessimism. This pessimism analysis is developed for both the classical Trajectory approach and the modified Trajectory approach which takes into account the minimum duration constraints of periodic flows.

This proposed pessimism analysis is applied to the industrial AFDX network. The classical Network Calculus approach and the modified one with minimum duration constraints are taken into account in the evaluations. The difference between the computed worst-case delay upper bounds and the underestimated worst-case delays given by the proposed pessimism analysis shows the pessimism introduced by the Network Calculus approach. The percentages of pessimism out of the computed delay upper bounds for the classical method and the modified method are of 18% and 17%, respectively. The proposed pessimism analysis provides a direct analytical pessimism computation for the Trajectory approach. Both the classical Trajectory approach and the modified one with minimum duration constraints are evaluated. The results show that there is on average of about 12% introduced pessimism of the computed end-to-end delay upper bounds for the classical Trajectory approach as well as 14% introduced pessimism for the modified Trajectory approach. Clearly, the Trajectory approach introduces less pessimism than the Network Calculus approach since it gives tighter computed delay upper bounds.

Chapter 6

Worst-case delay analysis on a heterogeneous network

Contents

6.1 Introduction 127
6.2 Heterogeneous network architecture 129
6.2.1 CAN bus 129
6.2.2 Switched Ethernet backbone 129
6.2.3 Heterogeneous flows 130
6.2.4 Bridging strategy 131
6.2.5 End-to-end delay analysis 132
6.3 Component-based approach for worst-case delay analysis 133
6.3.1 Component-based architecture 133
6.3.2 Network Calculus approach for Non-preemptive FP Scheduling 138
6.4 Trajectory approach for worst-case delay analysis 139
6.4.1 Classical Trajectory approach for FP/FIFO scheduling applied to a homogeneous network 139
6.4.2 A modified Trajectory approach adapted to a heterogeneous network . 143
6.5 Case study 147
6.6 Improved Trajectory approach 150
6.7 Conclusion 153

6.1 Introduction

Controller Area Network (CAN) has been developed and successfully applied to industrial applications during the last two decades. Due to the increasing data exchange needs in the industrial environment, CAN bus no longer satisfies huge demand of data exchange. One way to build a large network is to use a switched Ethernet network that serves as a backbone network interconnecting several CAN buses via bridges [SBF05].

When using such a heterogeneous network for real-time applications, the computation of end-to-end delay upper bounds is a key issue. A CAN flow received at another CAN bus after crossing a backbone switched Ethernet network experiences bandwidth adaptations, scheduling policies and bridging strategies. The end-to-end communication delay is the sum of delays on its source and destination CAN buses with non-preemptive Fixed Priority (FP) scheduling plus delays on switched Ethernet network with First In First Out (FIFO) scheduling. For such a real-time network, the worst-case delay analysis is mandatory.

A similar heterogeneous network has been studied by Scharbarg *et al.* in [SBF05]. But the network is not deterministic due to the CSMA/CD mechanism. In [AM07], Arjmandi *et al.* proposed to use switched Ethernet to connect different types of fieldbuses, where a single switch buffers frames in order of their priorities. In [ESMGZ08], an overview of Quality of Service (QoS) paradigms for heterogeneous networks has been discussed. Recently, Rivas *et al.* [RGPH11] discussed heterogeneous Earliest Deadline First (EDF) and Fixed Priority (FP) distributed real-time systems by a compositional approach. However, previous works do not consider a heterogeneous network with both FIFO and non-preemptive FP scheduling policies.

In order to deal with heterogeneities of such a network, one possible solution is a component-based approach which has been discussed in the domain of complex systems. Especially, a hierarchical scheduling frameworks [SL04, LLB06, SL03] where components (applications) form a hierarchy has received increasing attention. Such an approach allows each component to be developed and analyzed independently and therefore guarantees temporal isolation among components. Another way is to consider the Trajectory approach, which has been successfully applied to homogeneous switched Ethernet networks [MM06a, MM06b, BSF10, BSF12]. This approach has been introduced in previous chapters. In order to apply this approach, the heterogeneity properties (e.g. heterogeneous scheduling policies) have to be integrated.

The aim of this chapter is to develop two approaches to compute the end-to-end delay upper bounds on such a heterogeneous network. The key issue is the integration of heterogeneous scheduling policies. The first approach is based on a component-based principle which allows each component to keep its own properties. It is holistic and introduces pessimism at each component level. The second one is an adapted Trajectory approach which integrates heterogeneity properties by unifying them since the computation of a given flow is processed on its whole trajectory (path) and does not allow heterogeneity properties along the trajectory. The two approaches are compared on a middle-scale network case study.

In this chapter, pessimism introduced by the adapted Trajectory approach is also identified. An improved Trajectory approach is proposed. The results show that the improved Trajectory

approach computes tighter delay upper bounds than the component-based approach in the context of a heterogeneous network.

This chapter is organized as follows. Section 6.2 introduces the heterogeneous network architecture considered in this chapter and illustrates the worst-case delay computation on an example. Section 6.3 presents the component-based architecture of the considered network and applies a component-based approach to such a network. Section 6.4 adapts the Trajectory approach by integrating heterogeneity properties of the considered network for the worst-case delay analysis. Section 6.5 applies both proposed approaches to a middle-scale real-time heterogeneous network and analyzes the obtained results of both approaches. Section 6.6 indicates the pessimism introduced by the adapted Trajectory approach and proposes an improved Trajectory approach which provides tighter computation results on case study. Section 6.7 concludes this chapter.

6.2 Heterogeneous network architecture

The network architecture considered in this chapter is composed of a set of CAN buses interconnected by a switched Ethernet network. An illustrative example of such a network is depicted in Figure 6.1. It includes three CAN buses interconnected by two switches S_1 and S_2 via bridges.

6.2.1 CAN bus

Each CAN bus is based on the CAN 2.0 A specification (see [CAN] for details) and interconnects a set of Electronic Control Units (ECUs). The scheduling on such a CAN bus is non-preemptive FP scheduling. An identifier on 11 bits is associated to each frame. It defines the priority of the frame: at each time, the pending frame with the highest priority (i.e. the smallest identifier) is transmitted. Each CAN frame includes up to 8 data bytes, leading to a maximum frame size of 135 $bits$, including bit stuffing. The bandwidth of each CAN bus is $R_{CAN} = 1\ Mbit/s$.

6.2.2 Switched Ethernet backbone

Each switch of the Ethernet backbone implements the classical IEEE 802.1d store and forward policy through static routing tables, and has an upper bounded switching latency sl to deal with frame forwarding. Even if the switching latency sl is upper bounded, various delays can

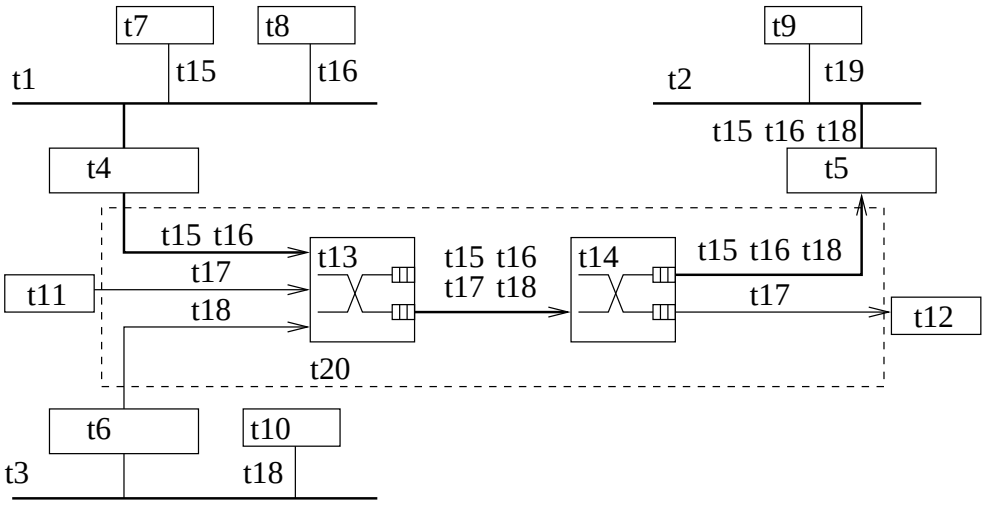


Figure 6.1: Heterogeneous network

be caused by FIFO queueing at output ports (see Chapter 1). Frame sizes are between 64 *bytes* and 1518 *bytes*. The bandwidth on each link is $R_{ETH} = 100\ Mbit/s$.

Two kinds of node are interconnected by the switched Ethernet backbone:

- Pure Ethernet nodes (N_1 and N_2 in Figure 6.1);
- Bridges connecting CAN and Ethernet ($Brdg_1$, $Brdg_2$ and $Brdg_3$ in Figure 6.1).

6.2.3 Heterogeneous flows

Three kinds of flows are transmitted over this heterogeneous architecture:

- Pure CAN flows (PC): such a flow concerns a set of ECUs interconnected by the same CAN bus;
- Remote CAN flows (RC): such a flow concerns a set of ECUs which are interconnected by two different CAN buses that must communicate through the Ethernet backbone;
- Pure Ethernet flows (PE): such a flow concerns a set of pure Ethernet stations.

As an example, the flows transmitted over the network architecture in Figure 6.1 are listed in Table 6.1. The type and path of each flow are indicated in this table.

τ_i	Type	Path $\mathcal{P}_i$
τ_1	RC	$\{CAN_1, Brdg_1, S_1, S_2, Brdg_2, CAN_2\}$
τ_2	RC	$\{CAN_1, Brdg_1, S_1, S_2, Brdg_2, CAN_2\}$
τ_3	PE	$\{N_1, S_1, S_2, N_2\}$
τ_4	RC	$\{CAN_3, Brdg_3, S_1, S_2, Brdg_2, CAN_2\}$
τ_5	PC	$\{CAN_2\}$

Table 6.1: Flows in the example in Figure 6.1

Each flow τ_i is characterized by its period T_i (minimum inter-arrival time of two consecutive frames), its frame transmission time C_i and its maximum release jitter J_i. Each CAN flow also has a priority p_i which is ignored by the switched Ethernet backbone.

6.2.4 Bridging strategy

Each bridge connects a CAN bus to the Ethernet backbone. In the CAN to Ethernet direction, the bridge has to transform the incoming CAN frames into Ethernet frames. In the Ethernet to CAN direction, the bridge releases CAN frames from the incoming Ethernet frames immediately. The frame encapsulation in the CAN to Ethernet direction and frame decapsulation in the Ethernet to CAN direction are modeled by an upper bounded delay T_B.

The transformation of CAN frames into Ethernet frames has to take into account the differences between CAN and Ethernet (frame sizes, bandwidths, scheduling policies, etc.). One important issue is the encapsulation strategy of CAN frames within Ethernet frames. Due to the very different frame sizes between CAN and Ethernet, different bridging strategies have been proposed in [SBF05]. The idea is to allow the encapsulation of more than one CAN frame in a single Ethernet frame and to send the encapsulated Ethernet frame withing a certain time. Thus, the strategy is characterized by two parameters:

- The maximum number N_B of CAN frames that can be encapsulated within a single Ethernet frame;
- The maximum waiting delay WD of a CAN frame pending in the bridge output port.

In this chapter, the following bridging strategy is used: a CAN frame arriving at a bridge output port waits at most for a time duration WD; it is transmitted on Ethernet earlier if there are N_B pending CAN frames before WD expires. Then the maximum and minimum waiting time for a remote CAN frame generated in a bridge in the CAN to Ethernet direction are $T_B + WD$ and T_B, respectively.

Remote CAN flows have a maximum frame transmission time after their frames are encapsulated in one Ethernet frame. Considering an Ethernet frame with 18 *bytes* of frame Header and CRC. encapsulating at most N_B CAN frames (at most 135 *bits* for each). The maximum size of the resulting Ethernet frame is

$$S_{ETH} = \lceil \frac{N_B \times 135}{8} \rceil + 18$$

Therefore, the maximum transmission time of such an Ethernet frame is

$$C_{ETH} = \frac{S_{ETH} \times 8}{100}$$

6.2.5 End-to-end delay analysis

The delay has to be upper bounded in the context of real-time networks. This can be achieved by a worst-case delay analysis, which has to take into account the heterogeneous properties of the network architecture considered in this chapter.

For example, consider τ_2 in Figure 6.1. It is emitted first at a CAN bus CAN_1 which supports non-preemptive FP scheduling with a bandwidth $R_{CAN} = 1\ Mbit/s$, then it crosses a bridge $Brdg_1$ where it is encapsulated in an Ethernet frame which is transmitted on the Ethernet backbone with FIFO scheduling and with a bandwidth $R_{ETH} = 100\ Mbit/s$, and finally it is decapsulated from the Ethernet flow at the bridge $Brdg_2$ and competes for the CAN bus CAN_2. Such a process is depicted in Figure 6.2.

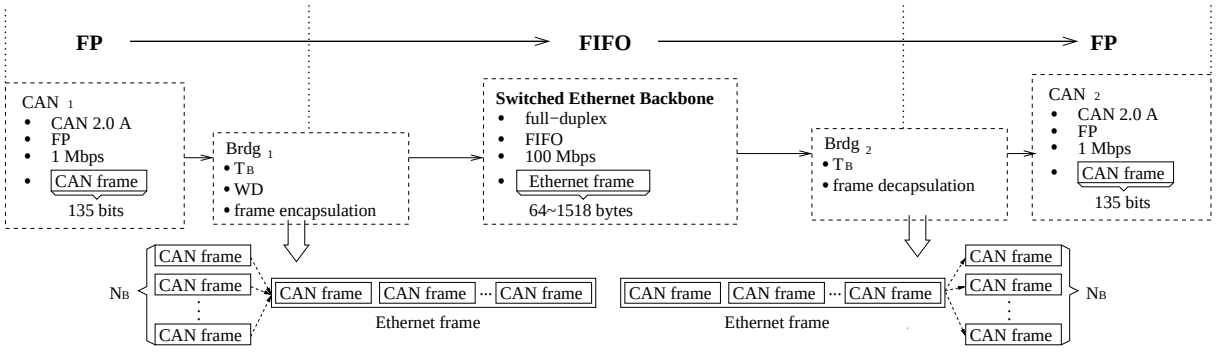


Figure 6.2: Heterogeneous path for τ_2

From the point of scheduling policy, τ_2 crosses FP, FIFO and FP schedulings, in this order. From the point of bandwidth, τ_2 is transmitted under bandwidths 1 $Mbit/s$, 100 $Mbit/s$ and

1 $Mbit/s$, in this order. Finally from the point of bridging strategy, τ_2 is encapsulated in an Ethernet flow at $Brdg_1$ and decapsulated at $Brdg_2$. Therefore, the worst-case delay analysis of such a flow has to integrate all these factors. However, existing approaches do not consider these heterogeneities.

In the following paragraphs, two approaches are proposed to deal with the worst-case end-to-end delay computation with the integration of heterogeneities of such a network.

6.3 Component-based approach for worst-case delay analysis

6.3.1 Component-based architecture

Component-based approach has been widely applied in real-time scheduling. In the context of timing property analysis, a compositional real-time scheduling framework is achieved where global timing properties are built by composing together independently analyzed local timing properties. Such an approach allows each component to be developed and analyzed independently and therefore guarantees temporal isolation among components. Especially, a hierarchical scheduling framework [SL04, LLB06, SL03] where components (applications) form a hierarchy, illustrated in Figure 6.3, has received increasing attention. Each component represents a scheduling model and a resource is allocated from a parent component to its children components. The key point of such an approach is to compose independently analyzed local timing properties, called scheduling interface, into a global timing property.

In the context of heterogeneous networks considered in this work, a component is defined as a self-contained network unit or subnetwork that can be used as a building block in the design of a larger network. Each component satisfies some functional properties and it is characterized by a scheduling policy and a set of sporadic flows. The scheduling policy is local to a component. Specifically speaking, for the considered heterogeneous network, three components are considered:

- The Ethernet backbone which exchanges flows using FIFO scheduling;
- Each CAN bus manages a set of pure and remote CAN flows using non-preemptive FP scheduling;
- Each Ethernet source node manages a set of pure Ethernet flows using FIFO scheduling.

Therefore, the component-based approach preserves FP scheduling for each CAN bus and FIFO

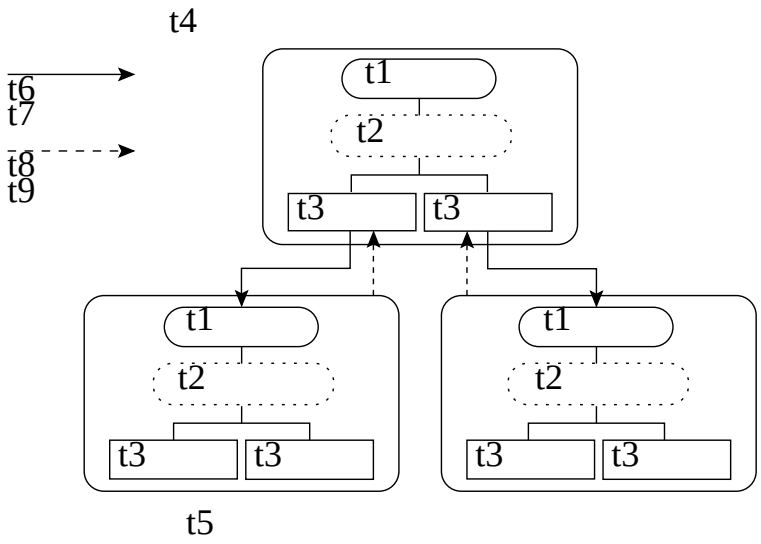


Figure 6.3: Hierarchical scheduling framework

scheduling for each Ethernet source node and the switched Ethernet backbone. However, each component has its own resource (bandwidth) and schedules its flows based on its local scheduler which is different from the hierarchical scheduling framework where the resource is allocated from the parent component to the children components. Hence, a hierarchical scheduling framework is not considered in our model.

One component can communicate with another one under two constraints:

- the communication between two CAN buses is unidirectional (a CAN bus cannot be the source of a flow and the destination of another flow),
- an Ethernet source node is not allowed to send Ethernet frames to a CAN bus.

Consider the heterogeneous network in Figure 6.1. Its compositional scheduling framework consists of six components: ETH, CAN_1, CAN_2, CAN_3, N_1 and N_2, as shown in Figure 6.4. The component CAN_1 communicates with the component CAN_2 by sending flows τ_1 and τ_2. Therefore, a remote CAN flow, like τ_2, crosses several components. At each visited component, it is considered as a local flow and scheduled by the local scheduler.

The component-based approach presented in [SL04, LLB06, SL03] cannot be directly applied to our context since there is no resource allocation issues. The component-based approach adopted for the proposed heterogeneous network architecture considers the worst-case delay upper bound of a flow τ_i as the sum of the worst-case delay upper bound of τ_i generated at

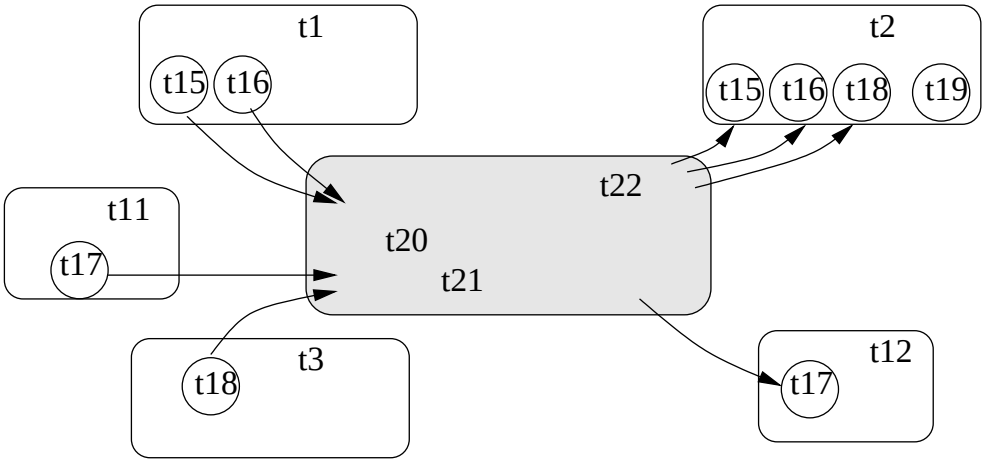


Figure 6.4: Compositional scheduling framework on an example

each visited component. The worst-case delay upper bound of τ_i generated at each visited component is obtained by a local worst-case delay analysis with a local scheduling policy at each component. This approach is illustrated in Figure 6.5 where a flow τ_i crosses m components $\{comp_1, comp_2, ..., comp_{m-1}, comp_m\}$. The worst-case delay upper bound $D_{max_i}^{comp_x}$ of τ_i in the component $comp_x$ ($x \in [\![1, m]\!]$) is obtained locally. Therefore the worst-case delay upper bound R_i of τ_i along its whole path is the sum of the delay upper bounds calculated at each crossed component:

$$R_i = \sum_{x \in [\![1,m]\!]} D_{max_i}^{comp_x}$$

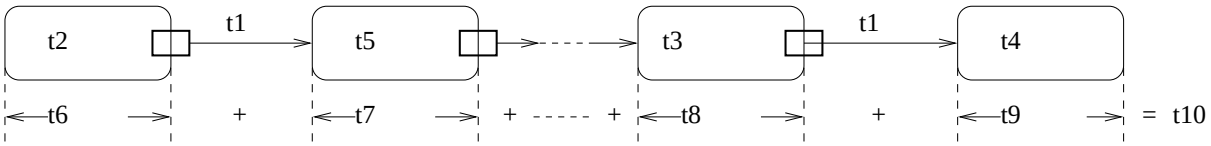


Figure 6.5: Component-based approach for heterogeneous networks

The central problem of component-based approach is to define the interface connecting the current component and the following component. At the interface of each component, the temporal characteristics of a flow are abstracted by a local analysis based on a local scheduling in order to propagate the flow model to the next component. The interface is represented by a set of outgoing flows to the following component. As shown in Figure 6.6, the input flow $\tau_i^{comp_m}$ at the component $comp_m$ is the output flow of $\tau_i^{comp_{m-1}}$ after it crossed component

$comp_{m-1}$ and its interface. The output flow gives its bandwidth requirement to the following component.

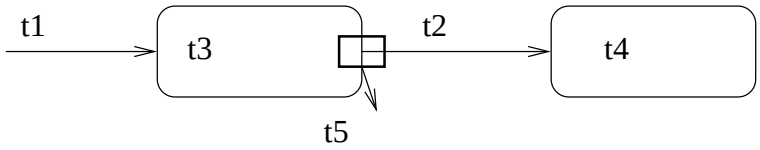


Figure 6.6: Component and interface

The bridges serve as the interfaces connecting a CAN bus and the Ethernet backbone. In the CAN to Ethernet direction, the bridge is in charge of abstracting incoming CAN flows to local Ethernet flows for the upcoming Ethernet backbone; conversely, it is in charge of abstracting incoming Ethernet flows to local CAN flows for the upcoming CAN bus. For example, the interface connecting the component CAN_1 to the component ETH is the bridge $Brdg_1$ which is characterized by a set of abstracted outgoing flows $\{\tau_1^{ETH}, \tau_2^{ETH}\}$.

An output flow $\tau_i^{comp_m}$ in Figure 6.6 is determined by the model of maximum frame density proved by Lemma 1 in [KAS08], which occurs when the first frame is delayed as much as possible and all further frames occur as early as possible. It means that $\tau_i^{comp_m}$ inherits a maximum release jitter $J_i^{comp_m}$ from $\tau_i^{comp_{m-1}}$. Suppose that the maximum and minimum delays of a frame of τ_i at $comp_m$ are $D_{max_i}^{comp_m}$ and $D_{min_i}^{comp_m}$ respectively. Then the maximum inherited jitter of $\tau_i^{comp_m}$ is computed by Equation (6.1):

$$J_i^{comp_m} = J_i^{comp_{m-1}} + D_{max_i}^{comp_{m-1}} - D_{min_i}^{comp_{m-1}} \tag{6.1}$$

Other temporal characteristics of the output flow $\tau_i^{comp_m}$ include:

- $T_i^{comp_m}$, which is equal to T_i, i.e., $T_i^{comp_m} = T_i$;
- $C_i^{comp_m}$, which depends on the maximum frame size of $\tau_i^{comp_m}$ and the transmission rate at $comp_m$;
- $p_i^{comp_m}$, which complies with the scheduling policy of $comp_m$.

The minimum delay $D_{min_i}^{comp_m}$ of $\tau_i^{comp_m}$ generated at the component $comp_m$ is the sum of the minimum frame transmission at each link plus bridge cost T_B of each visited bridge if there is any in $comp_m$ and switching latency sl of each visited switch if there is any in $comp_m$.

For the maximum delay $D_{max_i}^{comp_m}$ of $\tau_i^{comp_m}$ generated at component $comp_m$, the Network Calculus [BT01] approach is used for the worst-case delay computation. This method for FIFO scheduling has been recalled in 2.2, and the method for non-preemptive FP will be reviewed in 6.3.2.

Since the computation of the maximum delay of τ_i generated at a component is obtained by the Network Calculus approach, the worst-case delay of τ_i is the sum of the worst-case delays generated at each component visited by τ_i. This component-based approach is holistic.

We illustrate this component-based approach on τ_2 in Figure 6.4 crossing a set of components: $\{CAN_1, ETH, CAN_2\}$. This illustration is depicted in Figure 6.7.

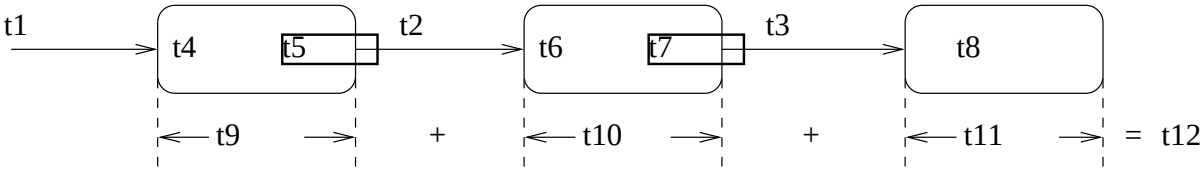


Figure 6.7: Illustration of the component-based approach for flow τ_2

From the component CAN_1 to the component ETH, the output flow τ_2^{ETH} needs to be determined at the interface ($Brdg_1$). Since the following component ETH is with FIFO scheduling, $T_2^{ETH} = T_2$, $C_2^{ETH} = C_{ETH}$ and $p_2^{ETH} = None$. Based on Equation (6.1), its inherited jitter is:

$$J_2^{ETH} = J_2 + D_{max_2}^{CAN_1} - D_{min_2}^{CAN_1}.$$

where $D_{max_2}^{CAN_1}$ is determined by the Network Calculus approach for non-preemptive FP scheduling.

Similarly, from the component ETH to the component CAN_2, the output flow $\tau_2^{CAN_2}$ is determined by $T_2^{CAN_2} = T_2$, $C_2^{CAN_2} = C_2$, $p_2^{CAN_2} = p_2$ and $J_2^{CAN_2}$ which is:

$$J_2^{CAN_2} = J_2^{ETH} + D_{max_2}^{ETH} - D_{min_2}^{ETH}.$$

where $D_{max_2}^{ETH}$ is determined by Network Calculus approach for FIFO scheduling.

The maximum delay $D_{max_2}^{CAN_2}$ of $\tau_2^{CAN_2}$ at the component CAN_2 is also calculated by Network Calculus approach for non-preemptive FP scheduling. Finally, the delay upper bound R_2 of τ_2 is computed by:

$$R_2 = D_{max_2}^{CAN_1} + D_{max_2}^{ETH} + D_{max_2}^{CAN_2}$$

6.3.2 Network Calculus approach for Non-preemptive FP Scheduling

The Network Calculus [BT01] approach is used since it is well developed for switched Ethernet networks with both non-preemptive FP ([Sch03]) and FIFO ([BSF10]) schedulings. The basis of the Network Calculus approach has been reviewed in 2.2. Here we recapitulate this classical approach for non-preemptive FP scheduling.

At each network element, the input flows are sorted into n_p classes according to priorities. The arrival traffic of a flow τ_i is constrained by an *arrival curve* $\alpha_i(t) = r_i t + b_i$, with the burst $b_i = l_{max_i}$ and the leak rate $r_i = l_{max_i}/T_i$, where l_{max_i} is the maximum frame size of τ_i. For class j, $j \in [\![1, n_p]\!]$, which has a set $\mathcal{J}$ of flows having the same priority, the aggregated arrival curve is the sum of each arrival curve of each flow in this class, i.e. $\alpha_j = \sum_{i \in \mathcal{J}} \alpha_i$.

Each network element provides a service curve $\beta(t) = R[t-T]^+$ with full capacity (rate) R and a latency T to flows. Several service curves with full capacity are considered in this paper depending on the specific network element:

- CAN bus: $\beta(t) = R_{CAN} \cdot t$;
- pure Ethernet node: $\beta(t) = R_{ETH} \cdot t$;
- switch output port: $\beta(t) = R_{ETH}(t - sl)^+$;
- bridge output port towards Ethernet backbone:
  $\beta(t) = R_{ETH}(t - T_B - WD)^+$;
- bridge output port towards a CAN bus:
  $\beta(t) = R_{CAN}(t - T_B)^+$.

The latency value of a service curve includes the switching latency sl, the encapsulation and decapsulation delay T_B and the maximum waiting delay WD in the bridge.

It is proved in [Sch03] that the service curves of lower priority classes are dependent on the arrival curves of higher priority classes. For each class j, the service curve β_j is given by:

$$\beta_j(t) = \left(Rt - \sum_{i \in [\![1, j-1]\!]} \alpha_j(t) - C_{max_j} \right)^+$$

for $j \in [\![1, n_p]\!]$, where $C_{max_j} = \max_{i \in [\![j+1, n_p]\!]}(C_i)$.

Therefore, the worst-case delay of a flow belonging to class j with aggregated arrival curve α_j in a network element which offers service curve β_j for class j is bounded by the horizontal deviation between α_j and β_j:

$$h(\alpha_j, \beta_j)$$

The Network Calculus approach, for both FIFO and non-preemptive FP schedulings, is applied to the calculation of the maximum delay $D_{max_i}^{comp_m}$ of $\tau_i^{comp_m}$ generated at component $comp_m$.

6.4 Trajectory approach for worst-case delay analysis

6.4.1 Classical Trajectory approach for FP/FIFO scheduling applied to a homogeneous network

The classical Trajectory approach for FIFO scheduling is presented in Chapter 3. It is also developed for FP/FIFO scheduling in [MM06b] and applied to a homogeneous switched Ethernet network in [BSF12]. It computes the ETE delay upper bound by considering for a frame the worst case scenario on its trajectory. Let us consider a flow τ_i following a path $\mathcal{P}_i$. The frame f_i of flow τ_i generated at time t is under study. At each output port h along its path, f_i can be delayed by other competing flows. The transmissions of these competing flows before f_i compose a busy period bp^h at the output port h. Such a busy period bp^h is a temporal interval with no idle time. In order to compute the worst-case ETE delay of frame f_i, the Trajectory approach considers the longest possible busy period ending with frame f_i at each node in its path.

For each flow τ_i, we define three sets according to priority:

- hp_i: the set of flows having a fixed priority strictly higher than that of τ_i;
- sp_i: the set of flows having a fixed priority strictly equal to that of τ_i;
- lp_i: the set of flows having a fixed priority strictly lower than that of τ_i;

The following notations are used for the computation:

- n_S: the number of switches visited by flow τ_i;

- n_B: the number of bridges visited by flow τ_i;

The worst-case delay of a frame f_i generated at time t contains several parts.

The first part is the delay generated by the flows in set sp_i. This is the same case as the Trajectory approach for FIFO scheduling introduced in Chapter 3. For a flow τ_j with the same priority as that of τ_i delaying τ_i at the node $first_{j,i}$, there is jitter interval $A_{i,j}$ which determines the maximum frame number of τ_j delaying f_i. This jitter interval is illustrated in Figure 3.4. Then for all flows in set sp_i, the competing workload is computed by:

$$D_{sp}(t, sp_i) = \sum_{\substack{j \in sp_i \\ \mathcal{P}_j \cap \mathcal{P}_i \neq \emptyset}} (1 + \lfloor \frac{t + S_{max_i}^{first_{j,i}} - S_{min_j}^{first_{j,i}} + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j \tag{6.2}$$

where $A_{i,j} = S_{max_j}^{first_{i,j}} - M_i^{first_{i,j}} + J_j$.

The second part is the delay generated by the flows in set hp_i. A flow τ_j with a higher priority than that of τ_i can delay τ_i at any shared node (from $first_{j,i}$ to $last_{i,j}$). The difference from flows with the same priority is that a frame of τ_j can delay f_i even if it arrives after f_i at a shared node, provided it arrives before the start of transmission of f_i. Hence for all flows in set hp_i, the competing workload is computed by:

$$D_{hp}(hp_i) = \sum_{\substack{j \in hp_i \\ \mathcal{P}_j \cap \mathcal{P}_i \neq \emptyset}} (1 + \lfloor \frac{W_{i,t}^{last_{i,j}} - S_{min_j}^{last_{i,j}} + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j \tag{6.3}$$

where $W_{i,t}^{last_{i,j}}$ is the latest starting time of the studied frame f_i at its last shared node $last_{i,j}$ with τ_j (Figure 6.8).

The third part is the delay caused by a lower priority flow due to non-preemptive effect. Its computation is detailed by Property 1 in [MM06b]. For any flow τ_i, the delay caused by flows in lp_i is denoted $\delta_i(lp_i)$.

The fourth part represents the transmission cost from one busy period to the next one (explained in [MM06b, BSF12]). For a flow τ_i, it considers the longest frame for each node (except $slow_i$) along path $\mathcal{P}_i$. Then for any flow τ_i, the transmission cost of busy periods along $\mathcal{P}_i$ is:

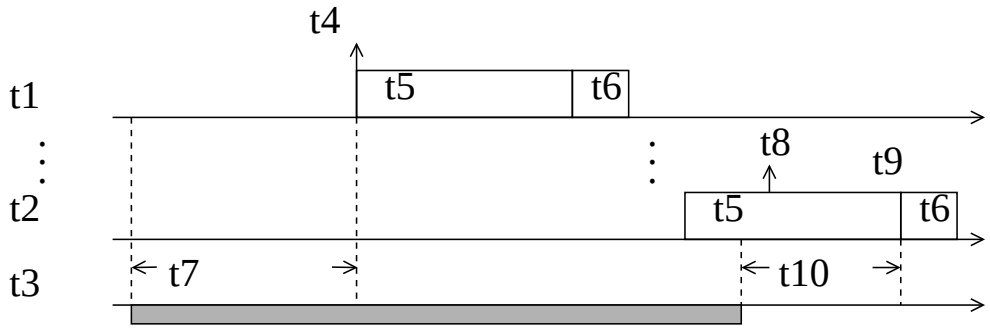


Figure 6.8: Interval of flow with higher priority

$$R_{trs}(hp_i \cup sp_i) = \sum_{h \in \mathcal{P}_i / \{slow_i\}} \left(\max_{\substack{j \in hp_i \cup sp_i \\ h \in \mathcal{P}_j}} \{C_j^h\} \right) \tag{6.4}$$

The last part is the sum of the switching latency which is an upper bounded sl per switch as well as the bridging delay which is an upper bounded T_B per bridge. For any flow τ_i, this cost is:

$$R_c = n_S \cdot sl + n_B \cdot T_B \tag{6.5}$$

Therefore, for a frame of any flow τ_i generated at time t, the latest starting time of its transmission on its last visited node $last_i$ is:

$$\begin{aligned} W_{i,t}^{last_i} &= D_{sp}(t, sp_i) + D_{hp}(hp_i) + \delta_i(lp_i) && (6.6) \\ &+ R_{trs}(hp_i \cup sp_i) + R_c && (6.7) \\ &- \sum_{h \in \mathcal{P}_i / \{first_i\}} (\Delta_{i,t}^h) && (6.8) \\ &- C_i^{last_i} && (6.9) \end{aligned}$$

Each part in Term (6.6) and Term (6.7) corresponds to one part of delay analyzed above. Term (6.8) considers frame serialization on links. More details on this optimization can be found in [BSF12]. Term (6.9) is subtracted since $W_{i,t}^{last_i}$ is not the received time of the considered frame, but its latest starting time on node $last_i$.

Then the worst-case ETE delay upper bound of the flow τ_i is obtained by:

$$R_i = \max_{-J_i+\mathcal{B}_i^{slow}>t\geq -J_i} \{W_{i,t}^{last_i} + C_i^{last_i} - t\} \tag{6.10}$$

where $\mathcal{B}_i^{slow} = \sum_{j\in hp_i\cup sp_i\cup i} \lceil \frac{\mathcal{B}_i^{slow}}{T_j} \rceil \cdot C_j^{slow_{j,i}}$. It is proved by Property 2 and Lemma 3 in [MM06b] and it limits the range of computation.

We illustrate the Trajectory approach for non-preemptive FP scheduling on the network architecture in Figure 6.1. In order to do that, a homogenized version of the network is first considered. It is assumed that all the links have the same bandwidth, and the bridging strategy is characterized by $N_B = 1$ (one CAN frame in one Ethernet frame) and $WD = 0$. It leads to a homogeneous network. In Section 6.4.2 we will show how the network in Figure 6.1 with heterogeneity properties can be analyzed by the Trajectory approach.

The computation is illustrated on flow τ_2 in the network example in Figure 6.1 which follows path $\mathcal{P}_2 = \{CAN_1, Brdg_1, S_1, S_2, Brdg_2, CAN_2\}$. Along its path, it competes with τ_1 for CAN_1, with τ_3 and τ_4 for the output port of S_1 as well as with τ_5 for CAN_2. We assume that $p_1 > p_2 = p_3 > p_4 > p_5$, i.e., for τ_2, $hp_2 = \{1\}$, $sp_2 = \{2, 3\}$ and $lp_2 = \{4, 5\}$.

According to Equation (6.2), the delay caused by flows in set $sp_2 = \{2, 3\}$ is computed by:

$$D_{sp}(t, sp_2) = \sum_{j\in\{2,3\}} (1 + \lfloor \frac{t + S_{max_2}^{first_{j,2}} - S_{min_j}^{first_{j,2}} + A_{2,j}}{T_j} \rfloor)^+ \cdot C_j$$

According to Equation (6.3), the delay caused by flows in set $hp_2 = \{1\}$ is computed by:

$$D_{hp}(hp_2) = \sum_{j\in\{1\}} (1 + \lfloor \frac{W_{2,t}^{last_{2,j}} - S_{min_j}^{last_{2,j}} + A_{2,j}}{T_j} \rfloor)^+ \cdot C_j$$

The worst-case delay incurred by flows in set $lp_2 = \{4, 5\}$ due to non-preemptive effect is $\delta_2(lp_2)$.

According to Equation (6.4), the transmission cost along path $\mathcal{P}_2$ is:

$$R_{trs}(hp_2 \cup sp_2) = \sum_{\substack{h\in\mathcal{P}_2 \\ h\neq CAN_1}} \max_{\substack{j\in\{1,2,3\} \\ h\in\mathcal{P}_j}} \{C_j^h\}$$

Since τ_2 crosses two switches ($n_S = 2$) and two bridges ($n_B = 2$), the cost of the switching latency and the bridging delay is $R_c = 2 \times sl + 2 \times T_B$.

Thus, the latest starting time of frame transmission of τ_2 on its last visited node CAN_2 is:

$$\begin{aligned} W_{2,t}^{CAN_2} &= D_{sp}(t, sp_2) + D_{hp}(hp_2) + \delta_2(lp_2) \\ &+ R_{trs}(hp_2 \cup sp_2) + R_c - C_2^{CAN_2} \end{aligned}$$

Then the worst-case ETE delay of τ_2 is upper bounded by:

$$R_2 = \max_{t \geq 0}\{W_{2,t}^{CAN_2} + C_2^{CAN_2} - t\} \tag{6.11}$$

6.4.2 A modified Trajectory approach adapted to a heterogeneous network

In order to cope with the architecture in Figure 6.1, three heterogeneity characteristics are integrated in the Trajectory approach. First, bridging strategies adopt arbitrary values of N_B and WD. Second, links have different bandwidths. Last, output ports have different scheduling policies.

Integration of bridging strategy

Let us first focus on the bridging strategy. For the direction CAN to Ethernet, there is an overhead T_B to deal with frame encapsulation which is considered in R_c. After one CAN frame arrives at the bridge output port connecting Ethernet, it waits at most for a duration WD before encapsulation. This waiting duration WD needs to be added to the computation If flow τ_i crosses $n_{B_{out}}$ bridge output ports connecting Ethernet, thus the total waiting delay is:

$$n_{B_{out}} \cdot WD$$

For the direction Ethernet to CAN, CAN frames are decapsulated from Ethernet frames immediately after a duration T_B to handle frame decapsulation.

Integration of different bandwidths

Different bit-rates in the heterogeneous network lead to different transmission durations for a remote CAN flow (e.g. τ_1 or τ_2 in Figure 6.1). For such a remote CAN flow τ_i, its transmission time at a CAN bus is C_i, and its transmission time at the Ethernet backbone is C_{ETH} as analyzed in Section 6.2.4. In order to guarantee the worst-case delay, the Trajectory approach replaces C_i in Equation (6.2), (6.3) with the largest transmission time $C_i^{slow_i}$ (see [MM06b] for details). For each remote CAN flow τ_i, we have:

$$C_i^{slow_i} = \max\{C_i, C_{ETH}\}$$

For a pure Ethernet flow and a pure CAN flow:

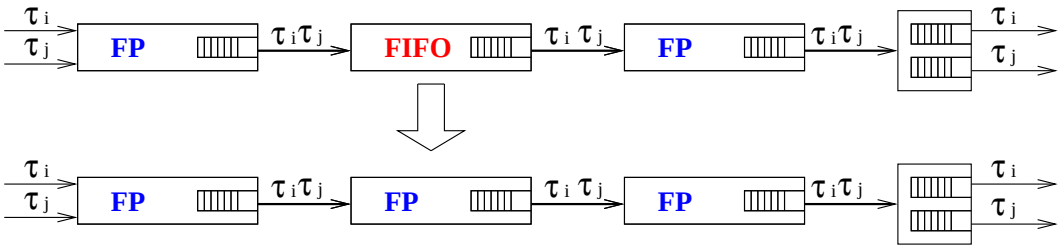
$$C_i^{slow_i} = C_i$$

Integration of different scheduling policies

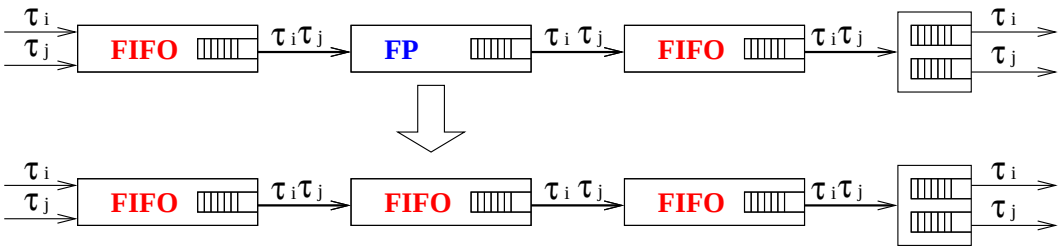
The last heterogeneity characteristic which has to be integrated concerns the schedulings of flows: Non-preemptive FP for CAN bus and FIFO for switched Ethernet. When a flow τ_j delays a flow τ_i and they share output ports with different scheduling policies, the delay upper bound computation of τ_i considers the following cases:

- Case 1: $p_j > p_i$, then FIFO is replaced by FP along their shared path since FP is less favorable for τ_i and can only increase its delay;
- Case 2: $p_j = p_i$, then FIFO and FP are equivalent, so FIFO is considered along their shared path (the same as FP/FIFO in [MM06b]);
- Case 3: $p_j < p_i$ and $first_{j,i}$ is with FP scheduling, then the computation considers FP scheduling along their shared path because τ_j cannot have another frame delaying τ_i after $first_{j,i}$ due to the frame serialization.
- Case 4: $p_j < p_i$ and $first_{j,i}$ is with FIFO scheduling along their shared path, then the computation considers FIFO scheduling because it is pessimistic for τ_i by assigning a higher priority p_i to τ_j.

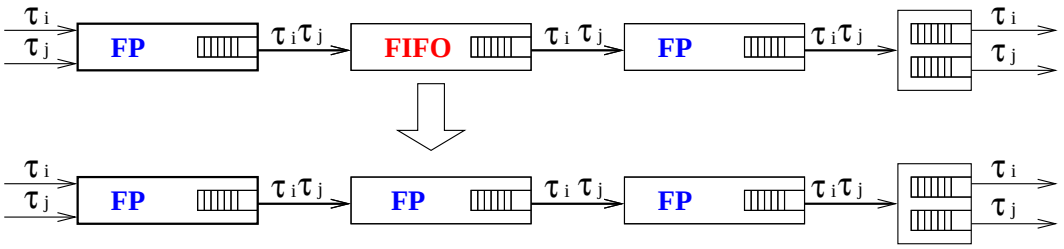
These cases are illustrated in Figure 6.9.



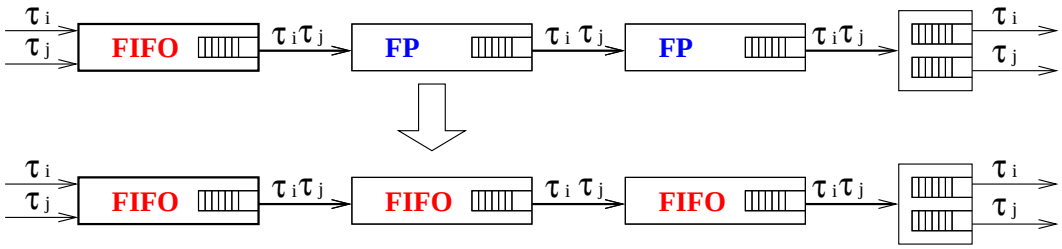
(a) Case 1



(b) Case 2



(c) Case 3



(d) Case 4

Figure 6.9: Illustration of heterogeneous scheduling policies integration

Three new sets shp_i, ssp_i and slp_i are defined according to the considered scheduling policy:

- shp_i: $j \in shp_i$ if $p_j > p_i$ and τ_j shares pure FP or both FP and FIFO scheduling policies with τ_i (Case 1);
- ssp_i: $j \in ssp_i$ if $p_j = p_i$; or if $p_j < p_i$ and $first_{j,i}$ is with FIFO scheduling (Case 2 and Case 4);
- slp_i: $j \in slp_i$ if $p_j < p_i$ and $first_{j,i}$ is with FP scheduling (Case 3).

Computation on a heterogeneous network

With the integration of the three heterogeneity properties, the latest starting time of a frame of flow τ_i generated at time t on its last visited node $last_i$ is:

$$\begin{aligned} W_{i,t}^{last_i} = & D_{sp}(t, ssp_i) + D_{hp}(shp_i) + \delta_i(slp_i) \\ & + R_{trs}(shp_i \cup ssp_i) + R_c + n_{B_{out}} \cdot WD \\ & - \sum_{\substack{h \in \mathcal{P}_i \\ h \neq first_i}} (\Delta_{i,t}^h) - C_i^{last_i} \end{aligned} \tag{6.12}$$

Then the worst-case delay of τ_i can be computed by Equation (6.10).

For τ_2 in Figure 6.1, τ_1 and τ_4 cross heterogeneous schedulings with τ_2. τ_1 has a higher priority than τ_2, then the computation of a pessimistic delay caused by τ_1 considers only FP scheduling. While τ_4 has a lower priority than τ_2 and $first_{4,2} = S_1$ is with FIFO, then the computation of a pessimistic delay caused by τ_4 considers only FIFO scheduling. τ_2 crosses one bridge output port towards Ethernet ($Brdg_1$) where the bridging strategy is with arbitrary values of N_B and WD. Therefore, for τ_2, $n_{B_{out}} = 1$ and $shp_2 = \{1\}$, $ssp_2 = \{2, 3, 4\}$ and $slp_2 = \{5\}$.

With the integrated heterogeneity characteristics, the latest starting time of frame transmission of τ_2 on CAN_2 is:

$$W_{2,t}^{CAN_2} = D_{sp}(t, ssp_2) + D_{hp}(shp_2) + \delta_2(slp_2) + R_{trs}(shp_2 \cup ssp_2) + R_c + WD - C_2^{CAN_2}$$

Then the worst-case ETE delay upper bound of τ_2 can be computed by Equation (6.11).

This adapted Trajectory approach is applied to a case study in the next section. And obtained results are compared to the ones obtained with the component-based approach.

6.5 Case study

This evaluation considers a middle-scale heterogeneous network in Figure 6.10. This network consists of four CAN buses interconnected by a switched Ethernet backbone via bridges. There are also eight pure Ethernet nodes connected to the Ethernet backbone. The bandwidth of each CAN bus is 1 $Mbit/s$ and the bandwidth of Ethernet is 100 $Mbit/s$. The non-preemptive FP scheduling policy is adopted for each CAN bus while the FIFO scheduling policy is adopted for each output port of the Ethernet backbone and each pure Ethernet node. The bridging strategy used in this evaluation is characterized by $N_B = 3$ and $WD = 500$ μs. For a bridge and a switch, the upper bounded costs are $T_B = 50$ μs and $sl = 10\mu s$, respectively.

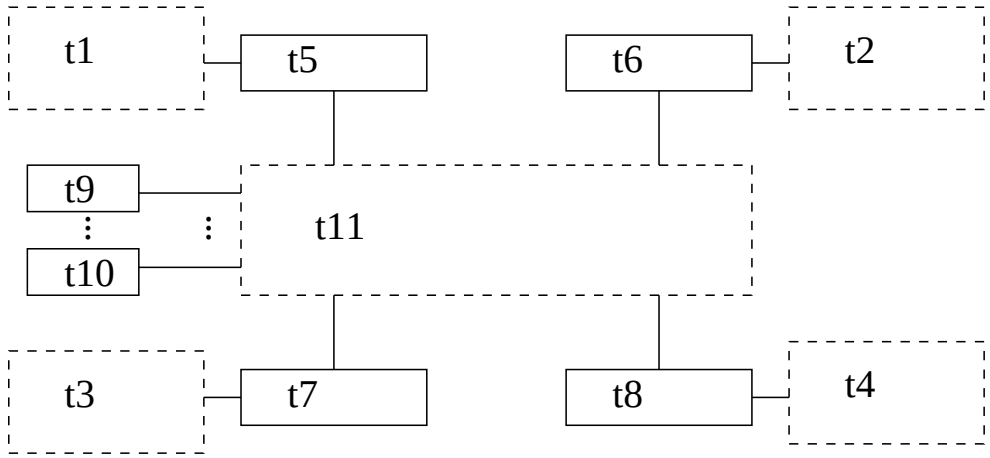


Figure 6.10: Heterogeneous network example

There are 100 flows transmitted over this network, among which there are 40 pure CAN flows, 20 remote CAN flows and 40 pure Ethernet flows. Each CAN bus transmits 10 pure

CAN flows. Moreover, for CAN_1 and CAN_2, each transmits 10 remote CAN flows to pure Ethernet nodes as well as to CAN_3 and CAN_4 through the Ethernet backbone. The temporal parameters of each pure and remote CAN flow τ_i are $T_i = 8000\ \mu s$, $C_i = 135\ \mu s$, $J_i = 0\ \mu s$. All CAN flows have different priorities. The remote CAN flows received by another CAN bus have the highest priorities, then the remote CAN flows received by Ethernet nodes follow, and the pure CAN flows have the lowest priorities. The network utilization of each CAN bus is 33.75%. The 40 pure Ethernet flows are transmitted between pure Ethernet nodes through the Ethernet backbone. Each pure Ethernet flow τ_i is characterized by: $T_i = 8000\ \mu s$, $C_i = 40\ \mu s$, $J_i = 0\ \mu s$. The average network utilization of the Ethernet backbone is 3.56%.

Figure 6.11 shows the worst-case ETE delays of all the pure Ethernet flows. They are sorted by increasing delay values. It can be seen from the figure that the adapted Trajectory approach provides tighter computed delays than the component-based approach.

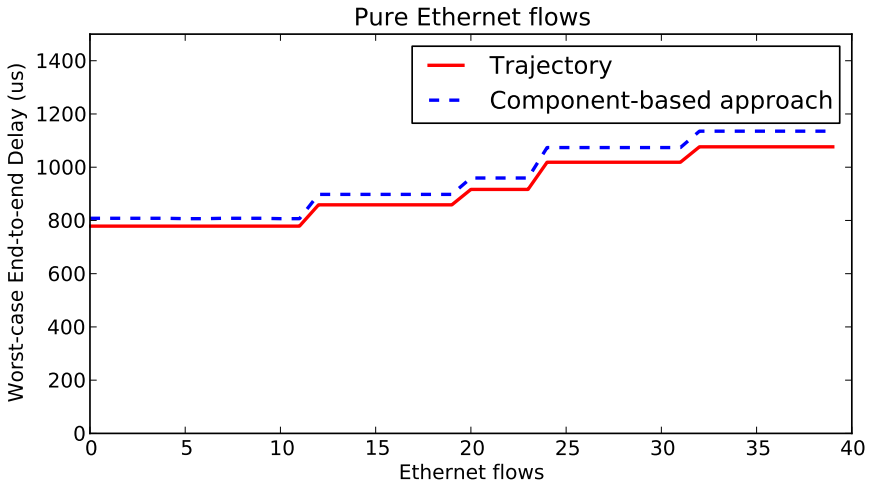


Figure 6.11: Worst-case ETE delay of pure Ethernet flows

Figure 6.12 shows the worst-case ETE delays of all pure CAN flows transmitted on CAN_4. They are sorted by increasing values of their identifiers (decreasing priorities). It can be seen from the figure that the adapted Trajectory approach provides tighter computed delays than the component-based approach. The results of pure CAN flows on the other three CAN buses are similar to the ones on CAN_4.

Figure 6.13 presents the worst-case ETE delays of all remote CAN flows sorted by increasing values of their identifiers The reason of the sawtooth shape of the curves is that these remote CAN flows cross the Ethernet backbone where they compete with different other flows along

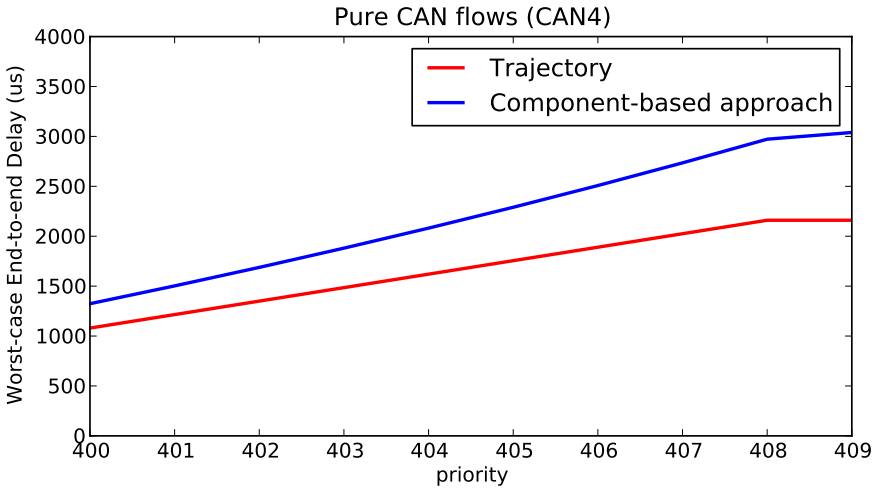


Figure 6.12: Worst-case ETE delay of pure CAN flows

different paths since the worst-case delay is directly related to the flows competing with the considered flow. For example, some remote flows sent from CAN_1 are received by CAN_3 and compete with some pure CAN flows at CAN_3, while some other remote flows sent from CAN_1 are received by CAN_4 and compete with some pure CAN flows at CAN_4. Therefore, flows received by CAN_3 compete with different sets of flows compared to the flows received by CAN_4 leading to different delays.

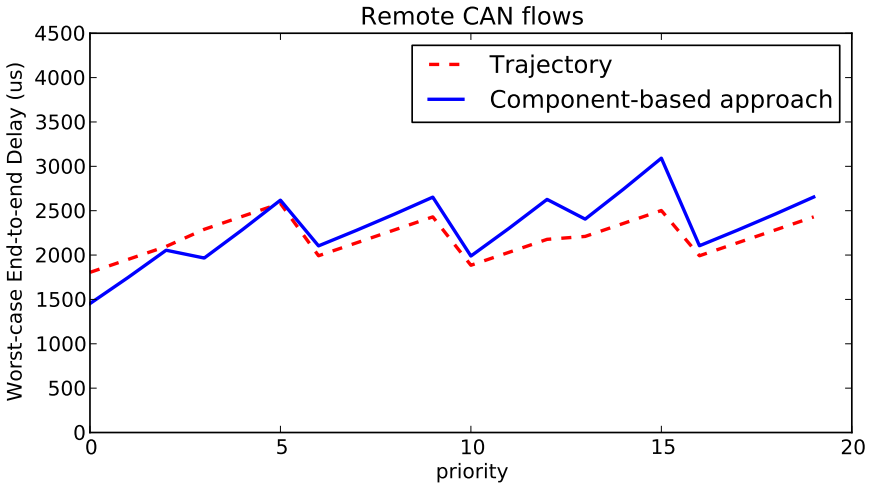


Figure 6.13: Worst-case ETE delay of remote CAN flows

In order to give a direct result for flows sharing the same path (i.e. competing with the same flows), separated results of Figure 6.13 are given in Figure 6.14 based on the path. It can

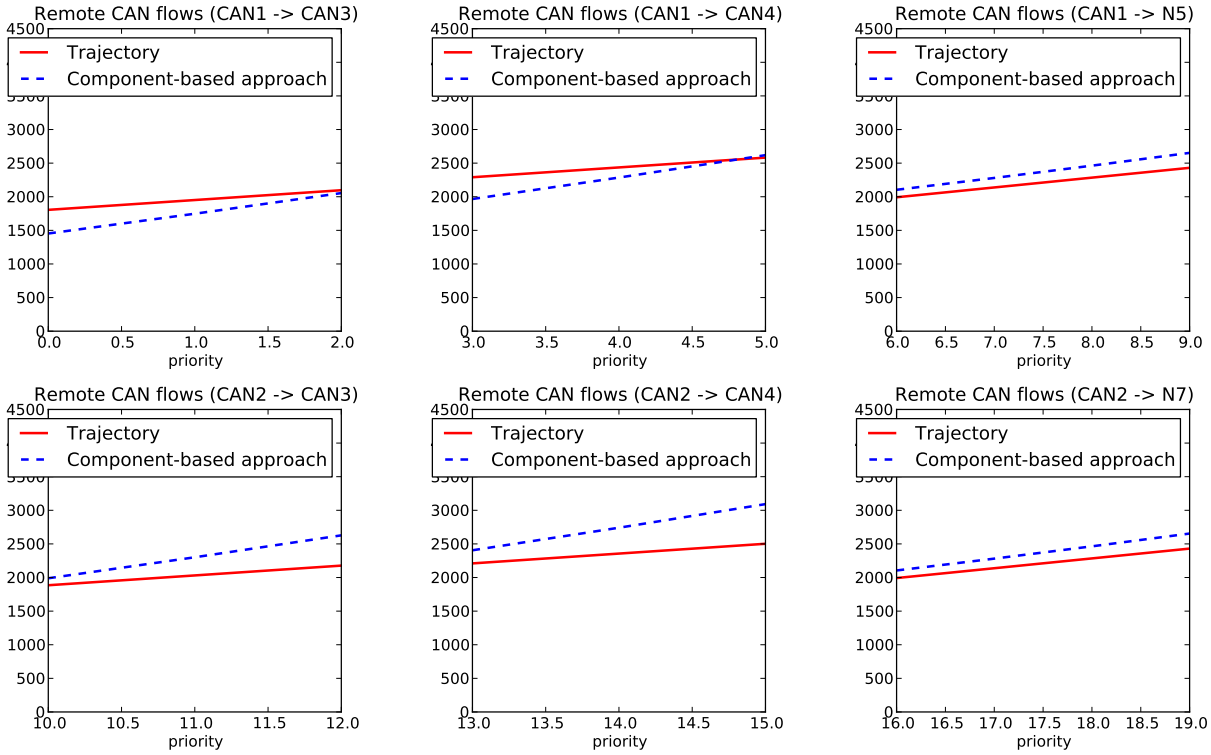


Figure 6.14: Worst-case ETE delay of remote CAN flows according to the path

be seen from the figure that the Trajectory approach provides tighter computed delays than the component-based approach for 14 flows. However, it leads to more pessimistic delays than the component-based approach for 6 flows with highest priorities (CAN_1 to CAN_3 and CAN_1 to CAN_4). The obtained pessimism for remote CAN flows with higher priorities is discussed in the next section and an improved Trajectory approach is proposed.

6.6 Improved Trajectory approach

There exists some pessimism introduced by the Trajectory approach on remote CAN flows with higher priorities, as showed in Figure 6.13. This pessimism is illustrated on τ_1 in an example network in Figure 6.15. Each CAN bus of CAN_1, CAN_2 and CAN_3 sends one remote flow to CAN_4 through an Ethernet backbone (switch S_1). For the purpose of simplicity, we consider that $N_B = 1$ and $WD = T_B = 0$ at a bridge as well as $sl = 0$ at switch S_1, i.e. $R_c = 0$.

For each flow τ_i, $T_i = 8000\ \mu s$, $C_i = 20\ \mu s$, $J_i = 0$ and $C_{ETH} = 5\ \mu s$. It is assumed that $p_1 > p_2 > p_3$, i.e. $hp_1 = \emptyset$, $sp_1 = \{1\}$ and $lp_1 = \{2, 3\}$.

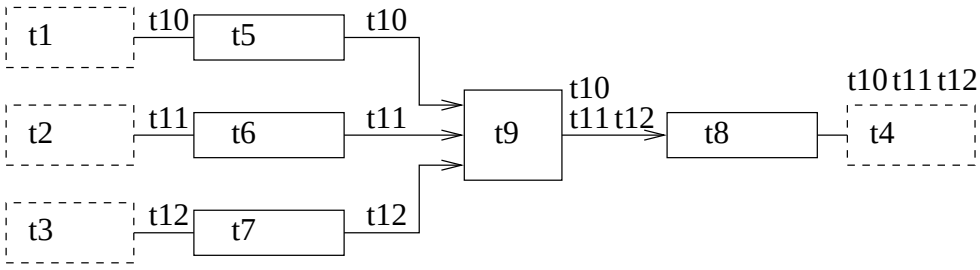


Figure 6.15: Example network

Consider a frame f_1 of τ_1 generated at time $t = 0$. Based on the Trajectory approach developed in Section 6.4.2, since τ_2 and τ_3 with lower priorities than that of τ_1 delays τ_1 first at a FIFO output port S_1, they are considered as with the same priority as τ_1, as Case 4 in Section 6.4.2. Therefore $shp_1 = \emptyset$, $ssp_1 = \{1, 2, 3\}$, $slp_1 = \emptyset$, and $C_2^{slow_{2,1}} = C_3^{slow_{3,1}} = 20\ \mu s$. Then, we obtain:

$$D_{sp}(0, ssp_1) = C_1^{slow_1} + C_2^{slow_{2,1}} + C_3^{slow_{3,1}} = 60\ \mu s$$

$$D_{hp}(shp_1) = \delta_1(slp_1) = 0$$

$$R_{trs}(shp_1 \cup ssp_1) = 5 + 5 + 20 = 30\ \mu s$$

According to Equation (6.12) and (6.10), the worst-case delay of τ_1 is computed as $R_1 = 90\ \mu s$.

The transmission of frame f_1 is illustrated in Figure 6.16, where the arrows represent frame arrivals. At switch S_1, frame f_1 is delayed by one frame f_2 of τ_2 and one frame f_3 of τ_3 due to the FIFO scheduling. At CAN bus CAN_4 with FP scheduling, frame f_1 is transmitted before frame f_3 due to the higher priority. Therefore, the worst-case delay of τ_1 is 70 μs which is 20 μs less than the computed result $R_1 = 90\ \mu s$.

The introduced pessimism of τ_1 is due to the integration of different bandwidths where $C_2^{slow_{2,1}}$ and $C_3^{slow_{3,1}}$ are considered in the computation. The integration leads to an impossible scenario where τ_2 and τ_3 delay τ_1 at S_1 with transmission times $C_2^{CAN_4} = C_3^{CAN_4} = 20\ \mu s$ since $slow_{2,1} = slow_{3,1} = CAN_4$. However, when both frames f_2 and f_3 delay frame f_1 at S_1, their frame transmission times are $C_2^{S_1} = C_3^{S_1} = 5\ \mu s$. Then considering 20 μs as their frame transmission times in the computation is pessimistic.

It happens when the considered scheduling is FIFO and the $slow_{j,i}$ is not with FIFO. In

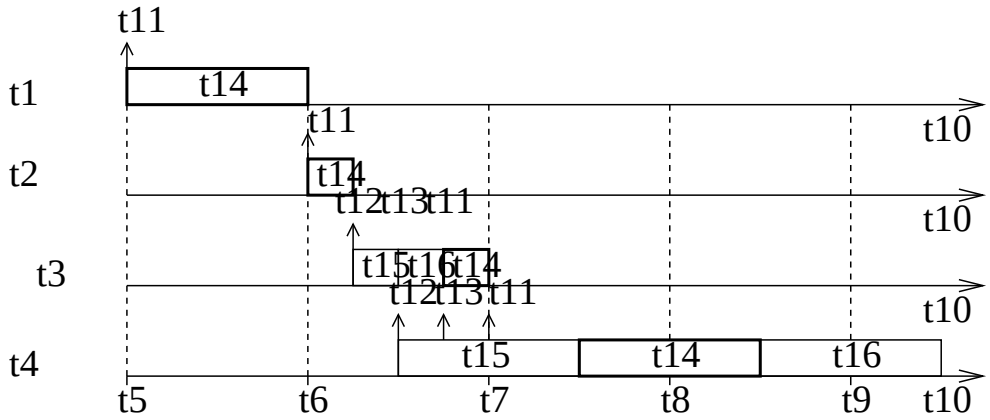


Figure 6.16: Illustration on τ_1

the context of the heterogeneous networks considered in this chapter, CAN bus has a much lower bandwidth than Ethernet backbone. Therefore this pessimism is experienced by flows of Case 4 in Section 6.4.2. In order to reduce the pessimism, for a flow τ_j corresponding to Case 4 in Section 6.4.2, its interference on τ_i is considered separately by FIFO scheduling and FP scheduling as follows:

- τ_j leaves $\mathcal{P}_i$ after their last shared FIFO output port $last_{i,j}^{FIFO}$.
- $\tau_j^{'}$ joins $\mathcal{P}_i$ at their first shared non-preemptive FP output port $first_{j,i}^{FP}$. $\tau_j^{'}$ is modeled by maximum frame density and characterized by

$$\{T_{j'} = T_j, C_{j'} = C_j, p_{j'} = p_j, J_{j'} = J_j + S_{max_j}^{first_{j,i}^{FP}} - S_{min_j}^{first_{j,i}^{FP}}\}$$

This is safe since for FIFO output ports of the shared path, flow τ_j is still assigned with a higher priority p_i, while for FP output port of the shared path, flow $\tau_j^{'}$ is with its own priority. We use lpb_i to denote the set of these new generated flows. Then $\delta_i(slp_i)$ in Equation (6.12) is replaced by $\delta_i(slp_i \cup lpb_i)$.

For τ_1 in Figure 6.15, $lpb_1 = \{2^{'}, 3^{'}\}$. Since τ_2 and τ_3 are considered to leave $\mathcal{P}_1$ after S_1, then $C_2^{slow_{2,1}} = C_3^{slow_{3,1}} = C_{ETH}$. Based on the improved approach, we obtain

$$D_{sp}(0, ssp_1) = C_1^{slow_1} + C_2^{slow_{2,1}} + C_3^{slow_{3,1}} = 30\ \mu s$$

$$\delta_1(slp_1 \cup lpb_1) = 20\ \mu s$$

$$R_{trs}(shp_1 \cup ssp_1) = 5 + 5 + 20 = 30\ \mu s$$

Therefore the improved result is $R_1 = 80\ \mu s$, which is still pessimistic but tighter than previous computed result.

This improved approach is applied to the network in Figure 6.10. The computed results for remote CAN flows are shown in Figure 6.17. Compared to results in Figure 6.13, this improved approach provides tighter delays than the component-based approach for all the remote CAN flows.

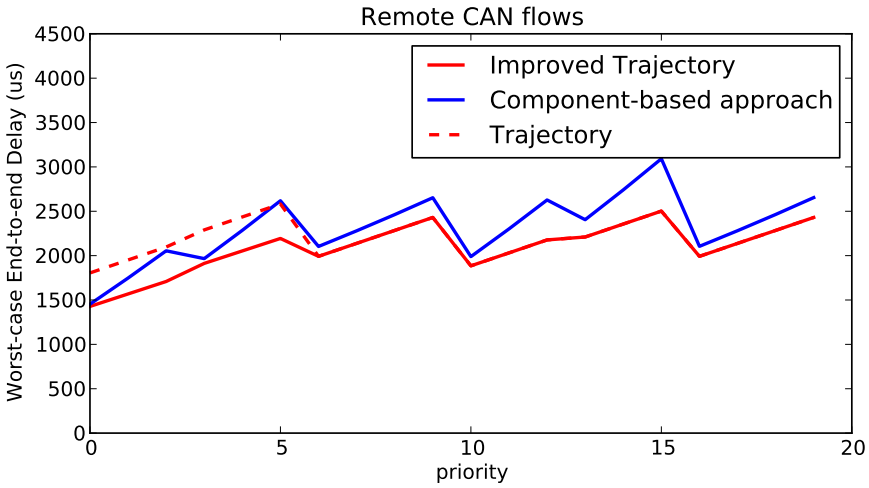


Figure 6.17: Worst-case ETE delay of remote CAN flows with the improved Trajectory approach

It can be noticed that only the results of remote CAN flows following the paths $CAN_1 -> CAN_3$ and $CAN_1 -> CAN_4$ are different. In Figure 6.18, the separated results of these two paths are given.

6.7 Conclusion

In this chapter two approaches for worst-case delay analysis are applied to a real-time heterogeneous network where a switched Ethernet backbone interconnects several CAN buses via bridges. The first presented approach is based on a component-based approach which allows each component to be analyzed independently with a local scheduling and therefore deals with heterogeneity properties at component level. The second introduced approach is an adapted Trajectory approach which integrates heterogeneity properties in the computation by unifying

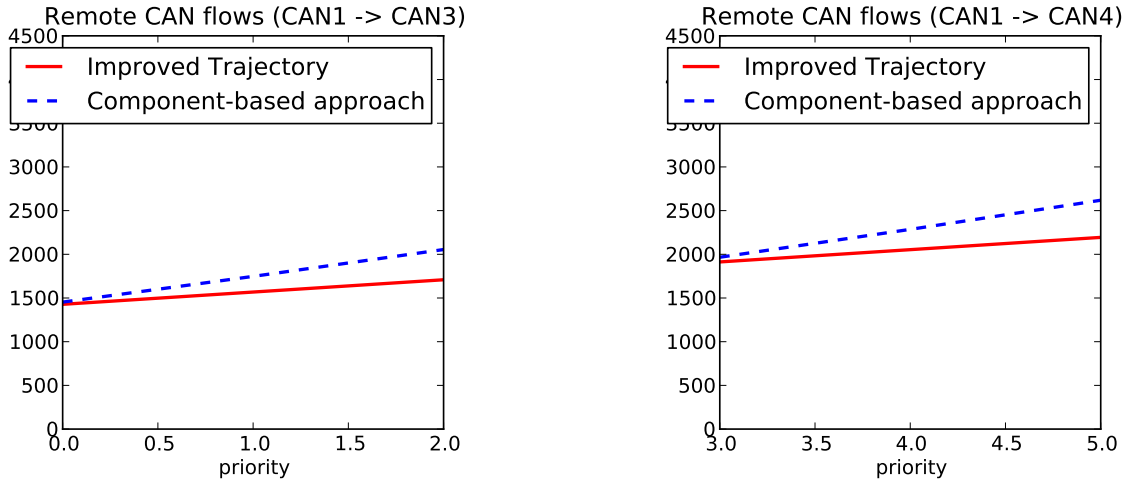


Figure 6.18: Worst-case ETE delay of remote CAN flows of CAN_1 to CAN_3 and CAN_1 to CAN_4 with the improved Trajectory approach

the properties along the considered path.

It has been shown on a heterogeneous network case study that the adapted Trajectory approach provides tighter end-to-end delay upper bounds for most flows than the component-based approach, and looser end-to-end delay upper bounds for a few remote flows. The pessimism of adapted Trajectory approach is identified and then an improved Trajectory approach is developed by removing part of the pessimism. It brings tighter end-to-end delay upper bounds for all flows than the component-based approach on the example network.

Conclusions and Perspectives

Real-time switched Ethernet network is a promising candidate for the industrial applications. It is necessary to demonstrate that worst-case end-to-end delays can be upper bounded since the delay upper bound is a crucial performance to evaluate real-time networks. Fortunately, computation of delay upper bounds can be done by different existing approaches. These existing approaches consider that the flows exchanged in the network are independent. But, periodic flows emitted by the same source node are synchronized based on a local clock. Their offsets can be known, and increase the knowledge about the frame arrival times thus improving the worst-case delay analysis.

This thesis studied the worst-case delay analysis suitable for real-time switched Ethernet networks accounting for the offset constraints.

In the first chapter, we have shown that periodic flows with known offsets emitted by the same source node are dependent since they are scheduled by a common local clock. More precisely, for two periodic flows with known offsets emitted by the same source node, it has been shown that there exist minimum durations between their frame arrivals at the output port of their source node. Therefore, the scenario that these frames arrive at the same time can be impossible. It has been further illustrated that the minimum duration constraints propagate along the shared path of these two flows. Thus, the introduction of minimum duration constraints in the worst-case delay analysis benefit not only the delay computation at the source node but also the delay computation along the whole path.

We proposed two approaches, the Network Calculus approach and the Trajectory approach, to implement the integration of minimum duration constraints in Chapter 2 and Chapter 3. The idea was to classify dependent flows in groups and compute the maximum delay generated by each group in stead of by each flow in the group. The computation of maximum delay generated by each group took into account the minimum duration constraints of flows in the group. Both two modified approaches have been illustrated on the same network example

and the results have shown great improvement on delay upper bounds (27%) compared to the results of classical approaches. As a real-time switched Ethernet application, the Avionics Full-DupleX switched Ethernet (AFDX) network was introduced in Chapter 4. An industrial AFDX configuration was presented and evaluated by the two modified approaches. The results have showed that compared to the classical approaches the modified approaches improve significantly the network performance with respect of end-to-end delay upper bounds (49% on average for the Network Calculus approach and 50% on average for the Trajectory approach). Moreover, for a scenario with only less than half of flows with offset constraints, the two modified approaches improved the delay upper bounds by about 17%. Hence, it is important to take into account the offset constraints in the worst-case delay analysis even partially. Since the importance of offset constraints was highlighted, we tried to find out the best offset assignment algorithm leading to the minimum computed end-to-end delay upper bounds for the industrial AFDX configuration (Chapter 4). Several offset assignment algorithms, including existing ones and the ones considering the avionics characteristics, have been evaluated by upper bounding the gap of each assignment and an optimal scenario. It has been demonstrated that heuristics that assign offsets to flows pair by pair are not efficient when applied to the industrial AFDX configuration due to the limited different values of period. A simple algorithm first designed for automotive networks turned out to be a near optimal algorithm. It has also been shown that although the proposed algorithms considering the avionics characteristics slightly outperform the simple algorithm, they increase the computation complexity.

The fifth chapter has investigated the pessimism analysis. The proposed worst-case delay analysis introduce pessimistic computation in order to guarantee the deterministic communication. In order to measure the maximum potential introduced pessimism, an analytical pessimism analysis was developed based on the Trajectory approach. Each possible part of the Trajectory approach computation leading to pessimistic results was analyzed. Then an underestimation of worst-case delay was derived. The maximum difference between the underestimation (lower bound) and the overestimation (upper bound) of end-to-end delay gives a metric of the maximum potential pessimism. This pessimism analysis has been further developed with the integration of offset constraints. It has been applied to the industrial AFDX configuration. The results have showed that there are on average of about 12% introduced pessimism of the computed delay upper bounds for the classical Trajectory approach and of about 14% introduced pessimism for the modified Trajectory approach. Therefore, since the exact worst-case delay upper bounds cannot be calculated for an industrial configuration, the approaches proposed in Chapter 2 and Chapter 3, although introduce some pessimism, are efficient to evaluate the network performance for an industrial configuration.

The last chapter presented a real-time heterogeneous network where a switched Ethernet backbone interconnects several CAN buses via bridges. Such a heterogeneous architecture can serve as the future real-time industrial network. For the purpose of certification, two approaches for worst-case delay analysis have been proposed. The first presented approach is based on a component-based approach which allows each component analyzed independently with a local scheduling policy and therefore deals with heterogeneity properties at component level. The second introduced approach is an adapted Trajectory approach which integrates heterogeneity properties in the computation by unifying the properties along the considered path. The results have showed on a middle scale heterogeneous network case study that the adapted Trajectory approach provides tighter end-to-end delay upper bounds than the component-based approach. Therefore, a heterogeneous network can be used for real-time industrial applications since its worst-case delay upper bounds can be determined by adapted approaches.

Some perspectives are illustrated as follows.

Fixed priority policy ensures that at any given time, the processor executes the highest priority task of all those tasks that are currently ready to execute. Since the static priority demands added cost and more complicated implementation, many standard switched Ethernet network components only support the FIFO scheduling. Nowadays some real-time applications require guaranteeing the communication of a set of critical flows. For example, an avionics network transmits both avionics flows (higher priority) and non-avionics flows (lower priority), and it needs a worst-case delay analysis for the avionics flows. Such a work has been done in [BSF12]. However, it does not consider the offset constraints. Future work can focus on the integration of fixed priority policy in the approaches proposed in Chapter 2 and Chapter 3 of this thesis in order to provide worst-case delay upper bounds needed for a real-time switched Ethernet network with offset constraints.

The maximum potential pessimism features a gap from an exact worst-case delay to an overestimated delay (delay upper bound). However, there is no method to obtain an exact worst-case delay for a large scale network. The pessimism analysis presented in Chapter 5 provides a metric to calculate the upper bound of the maximum potential pessimism by measuring the gap from an underestimated worst-case delay, which is smaller than the exact worst-case delay, to an overestimated delay. Therefore, in order to obtain an evaluation on the maximum pessimism, future work focuses on tightening the gap from the underestimation to the upper bound of worst-case delay.

As a future research direction, the heterogeneous network has been introduced and analyzed in Chapter 6. In this work, only CAN buses with non-preemptive FP scheduling are considered.

Other types of field buses, e.g., CAN buses with TDMA scheduling, can be candidates for a heterogeneous network and therefore the corresponding worst-case delay analysis needs to be developed. Moreover, only a middle scale network was evaluated as a case study in this work. It needs to evaluate more complex large scale industrial networks in future work.

This thesis therefore leaves room for a wide and promising field of research, since the real-time switched Ethernet plays an increasingly important role in the industrial domain.

Appendix

Appendix A

Network Calculus

A.1 Network Calculus

For a network element, its input data flows are described by means of the cumulative function $R(t)$, which is defined as the amount of traffic that arrives at the element over time interval $[0, t]$. The output flows of the network element is described by the output function $R^*(t)$ defined as the amount of traffic that departs from the element over time interval $[0, t]$. Both $R(t)$ and $R^*(t)$ are wide-sense increasing functions.

In order to guarantee the data flows, the arrival curve is used to limit the traffic sent by sources. An arrival curve $\alpha(t)$ is a wide-sense increasing function. A flow represented by its input function R is constrained by an arrival curve α if and only if for all $s \leq t$

$$R(t) - R(s) \leq \alpha(t - s)$$

A classical arrival curve is affine function $\gamma_{r,b}(t)$, which is defined by:

$$\gamma_{r,b}(t) = \begin{cases} rt + b & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases}$$

where parameters b and r are called the burst tolerance and the rate. An arrival curve $\gamma_{r,b}(t)$ allows a source to send b bits at once, but not more than r bit/s over the long run. Such an arrival curve is illustrated in Figure A.1.

A network element offers some guarantees to flows using the concept service curve $\beta(t)$. A service curve $\beta(t)$ is a wide-sense increasing function with $\beta(0) = 0$. Consider a flow crossing a network element with input and output functions R and R^*. It is defined that this network

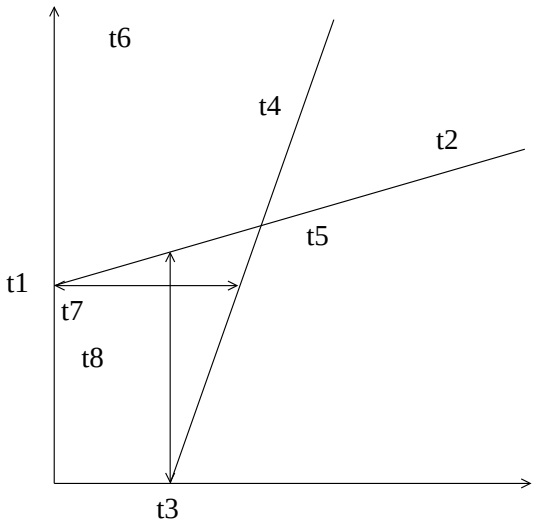


Figure A.1: Illustration of Network Calculus

element offers to the flow a service curve β if and only if for all $t \geq 0$

$$R^*(t) \geq \inf_{s \leq t}(R(s) + \beta(t-s))$$

A classical service curve is rate-latency function $\beta_{R,T}(t)$, which is defined by:

$$\beta_{R,T}(t) = R(t-T)^+ = \begin{cases} R(t-T) & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases}$$

where R and T are called the rate and the latency. A network element providing a service curve $\beta_{R,T}(t)$ delays an input flow at most by time units T, and then serves the flow by rate R. Such a service curve is illustrated in Figure A.1.

The Network Calculus brings three bounds for lossless networks with service guarantees. First, the backlog is bounded by the vertical deviation between the arrival and service curves. For a flow with constrained arrival curve α crossing a network element which provides a service curve β, the backlog $R(t) - R^*(t)$ for all t satisfies:

$$R(t) - R^*(t) \leq \sup_{s \geq 0}\{\alpha(s) - \beta(s)\}$$

Second, the delay is bounded by the horizontal deviation between the arrival and service

curves. The virtual delay at time t is

$$d(t) = \inf\{\tau \geq 0 : R(t) \leq R^*(t+\tau)\}$$

For a flow with constrained arrival curve α crossing a network element which provides a service curve β, the virtual delay $d(t)$ for all t satisfies:

$$d(t) \leq h(\alpha, \beta)$$

where

$$h(\alpha, \beta) = \sup_{s \geq 0}\{\inf\{\tau : \tau \geq 0 \; and \; \alpha(s) \leq \beta(s+\tau)\}\}$$

Third, the output flow is constrained. For a flow with constrained arrival curve α crossing a network element which provides a service curve β, the output flow is constrained by arrival curve:

$$\alpha^*(t) = \sup_{u \geq 0}\{\alpha(t+u) - \beta(u)\}$$

The arrival curve of the output flow serves as the arrival curve of the input flow of the following network element. In Figure A.2, α^* constrains the output flow of node h and is the arrival curve of input flow of node $h+1$. In this way, it allows the Network Calculus computation propagates till the destination node.

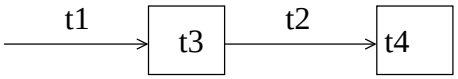


Figure A.2: Propagation of Network Calculus

A.2 Application on a switched Ethernet network

In a switched Ethernet network, a set of nodes are interconnected by switches via links. Each node of a switched Ethernet is equipped with a shaper, which guarantees that a flow τ_i emitted by its source node arrives with maximum traffic of l_{max_i} at once, and that two frames of one flow are separated by a minimum distance T_i. Then the flow is constrained by a leaky bucket model γ_{r_i,b_i} where the leak rate is $r_i = \frac{l_{max_i}}{T_i}$ and the burst is $b_i = l_{max_i}$. Such an arrival curve is represented by

$$\gamma_{r_i,b_i} = \frac{l_{max_i}}{T_i} \times t + l_{max_i}$$

Each flow transmitted over the switched Ethernet network is constrained by such an arrival curve at its source node.

A source node is a network element which emits a set of flows. It guarantees for each emitting flow an arrival curve via a shaper. Each source node has an output port which is equipped with a buffer and supports FIFO scheduling. It transmits frames at rate R. It is represented by a service curve $\beta_{R,0} = R(t)^+$ (also known as peak rate function). A switch is a device that channels incoming frame from any of multiple input ports to the specific output port that will take the frame toward its intended destination. It supports store and forward mode which means that the switch buffers and verifies each frame before forwarding it. When a frame crosses a switch, there is a switching latency which is used to verify the frame transmission and forward it to the output port toward its destination. The maximum switching latency of a switch is sl, which means that a frame is delayed in a switch at most by time of sl. Each output port of a switch is equipped with a buffer and supports FIFO scheduling. It transmits frames at rate R. Then the service curve provided by a switch is represented by a rate-latency function $\beta_{R,sl} = R(t - sl)^+$.

The worst-case delay of a frame in a network element, such as an output port of a node or of a switch, is determined by the maximum horizontal distance $h(\alpha, \beta)$ between its arrival curve and the service curve provided by this network element. It is illustrated in Figure A.3.

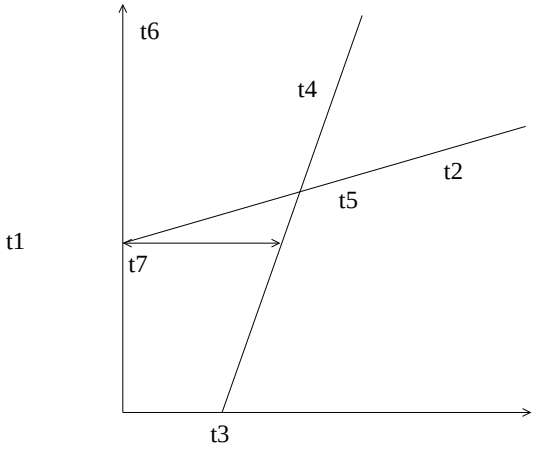


Figure A.3: Illustration of Network Calculus applied to real-time switched Ethernet

When there are more than one input flows crossing the considered network element, the arrival curve considers the sum of all input traffic. More precisely, when there are n input flows with arrival curves $\alpha_1, \alpha_2, ..., \alpha_n$ entering a network element providing a service curve β, then

the arrival curve of the aggregated traffic is

$$\alpha = \sum_{i \in [\![1,n]\!]} \alpha_i$$

In order to propagate the computation, it needs to determine arrival curve of each output flow departing from the network element. According to the dissertation of Jérôme Grieu [Gri04], the arrival curve constraining each output flow is obtained by shifting the corresponding input arrival curve toward left by maximum delay jitter of the considered flow generated in the network element. The resulting output arrival curves are not optimal, but are easy to calculate.

A path of a flow is modeled as the concatenation of a source node output port and several switch output ports. The Network Calculus computation starts from the source node output port along the path till the last visited switch output port. Therefore the end-to-end delay experienced by a frame following the path is the sum of delays encountered at each crossed output port.

A.3 Integration of frame serialization

Flows send frames to a node through different input links. Frames transmitted from different input links can arrive at the output port h at the same time, but they share the same output link and they are transmitted one by one through the output link. It means that they cannot be transmitted at the same time on this link, and it is called that they are serialized. This constraint is considered by Jérôme Grieu [Gri04].

We illustrate the frame serialization in Figure A.4. There are two input links: IP_0^h transmits a frame f_1 of flow τ_1 to the output port h; and IP_1^h transmits a frame f_2 of flow τ_2 to the output port h. At the output port h, f_1 and f_2 can arrive at the same time $a_{f_1}^h = a_{f_2}^h$. Then they are serialized at the output port h, which means that they are transmitted by the output link OP^h of h one by one. Therefore, they arrive at the following output port $h+1$ one after one, which means that they cannot arrive at $h+1$ at the same time as illustrated in Figure A.4. Then any scenario considering that f_1 and f_2 arrive at $h+1$ at the same time, i.e. $a_{f_1}^{h+1} = a_{f_2}^{h+1}$ is impossible, and introduces pessimism in the delay computation. Therefore, the frame serialization imposed by physical constraint of link should be taken into account in the worst-case delay analysis.

The frame serialization is integrated in the Network Calculus by considering that the arrival

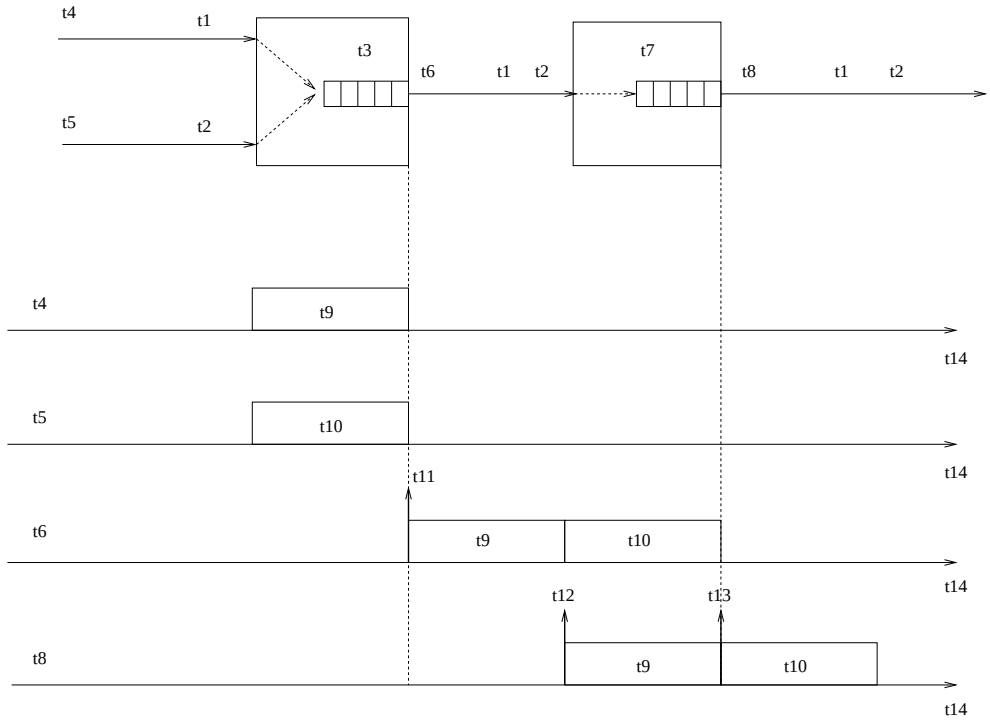


Figure A.4: Illustration on frame serialization

curve from one input port has a burst tolerance no more than the largest frame size it is carrying and a rate not higher than the transmission rate of link. More precisely, for flows competing for one output port h, they come from $k_h + 1$ input ports. For each input port IP_k^h ($k \in [\![0, k_h]\!]$), there are $n_{\mathrm{IP}_k^h}$ flows competing for the output port h. Then, for $n_{\mathrm{IP}_k^h}$ flows with input arrival curves $\alpha_i = \gamma_{r_i,b_i}$, $i \in [\![1, n_{\mathrm{IP}_k^h}]\!]$, coming from the same input link IP_k^h, the burst tolerance of their aggregated flow is $\max_{i \in [\![1, n_{\mathrm{IP}_k^h}]\!]}(b_i)$, and the aggregated flow of one input link arrives at a rate not higher than R. The aggregated flow of input port IP_k^h is then constrained by:

$$\alpha_{\mathrm{IP}_k^h} = \min_{t \in [0,\infty)} \Big(\sum_{i \in [\![1, n_{\mathrm{IP}_k^h}]\!]} (\alpha_i(t)), Rt + \max_{i \in [\![1, n_{\mathrm{IP}_k^h}]\!]} (b_i) \Big)$$

This curve is illustrated in Figure A.5.

Each input link of a network element provides such an aggregated flow of flows transmitted by it. The sum of all these aggregated flows gives the arrival curve of all the input flows in the network element, which is

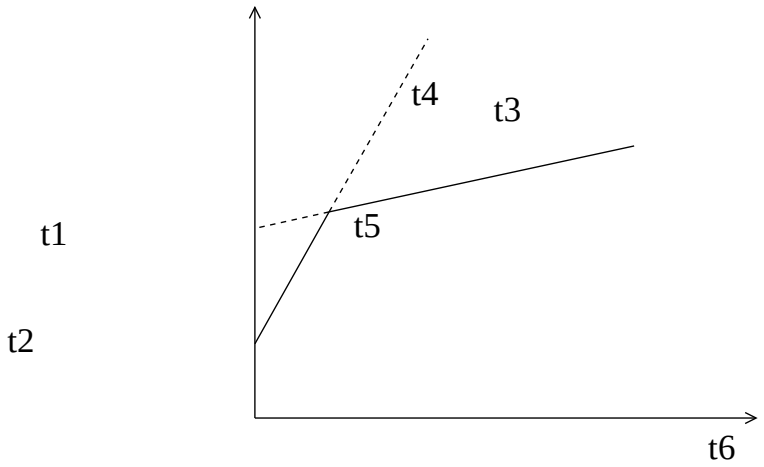


Figure A.5: The aggregated flow of one input link IP_k^h considering the frame serialization

$$\alpha^h = \sum_{k \in [\![0, k_h]\!]} (\alpha_{\mathrm{IP}_k^h}) \tag{A.1}$$

This arrival curve provides a tighter constraint on input flows, and therefore reducing pessimism of the computed results on worst-case delay.

Appendix B

Trajectory approach

B.1 Trajectory approach

The Trajectory approach is developed in [MM06a] for flows scheduled with FIFO. This approach considers a set of n sporadic flows transmitted over a network where neither failures nor frame losses happen. It computes the worst-case end-to-end delay upper bound by considering for a frame the worst case scenario on its trajectory.

A flow τ_i ($i \in [\![1, n]\!]$) following a path $\mathcal{P}_i$ is considered. The frame f_i of flow τ_i generated at time t is under study.

In order to present the Trajectory approach, we first introduce some notations:

- C_i^h: the frame transmission time of τ_i on node h;
- $first_i$ ($last_i$): the first (resp. last) node visited by τ_i on its path $\mathcal{P}_i$;
- $slow_i$: the slowest node visited by τ_i on its path $\mathcal{P}_i$;
- $slow_{j,i}$: the slowest node visited by τ_j on the path $\mathcal{P}_i$;
- bp^h: a busy period at node h without idle time during it.
- M_i^h: the starting instant of bp^h for the studied flow τ_i.
- $a_{f_i}^h$: the arrival time of frame f_i at node h.
- $first_{j,i}$: the node where a competing flow τ_j first crosses the studied flow τ_i.
- $S_{max_i}^h$ (resp. $S_{min_i}^h$): the maximum (resp. minimum) delay experienced by a frame of flow τ_i from its source node $first_i$ to node h.
- $A_{i,j}$: the workload interval of a flow τ_j to delay flow τ_i.
- $W_{i,t}^{last_i}$: the latest starting time of frame f_i at its last visited output port $last_i$.
- R_i: the worst-case end-to-end delay upper bound of flow τ_i.
- $|\mathcal{P}_i|$: the length of path $\mathcal{P}_i$.

- $h \in \mathcal{P}_i / \{first_i\}$: $h \in \mathcal{P}_i$ and $h \notin \{first_i\}$.
- L_{max} (L_{min}): the maximum (resp. minimum) processing delay when a frame crosses a link.
- $[\![1, n]\!]$: an integer set $\{1, 2, ..., n\}$.
- $(a)^+ = \max(a, 0)$.

Along the path $\mathcal{P}_i$, τ_i can cross several nodes. At each crossed node, τ_i can be delayed by different input flows which are transmitted in the busy period of f_i. For FIFO scheduling, a busy period of frame f_i is a time interval in which there is no idle time and all the frames arriving before the arrival of frame f_i are transmitted. In order to compute the worst-case end-to-end delay of frame f_i, the Trajectory approach considers the longest possible busy period bp^h ending with frame f_i at each node in its path. This is achieved by considering the maximum number of frames of each competing flow transmitted in the busy period. An illustration of busy periods along the path $\mathcal{P}_i = \{1, 2, 3, 4\}$ of τ_i is shown in Figure B.1.

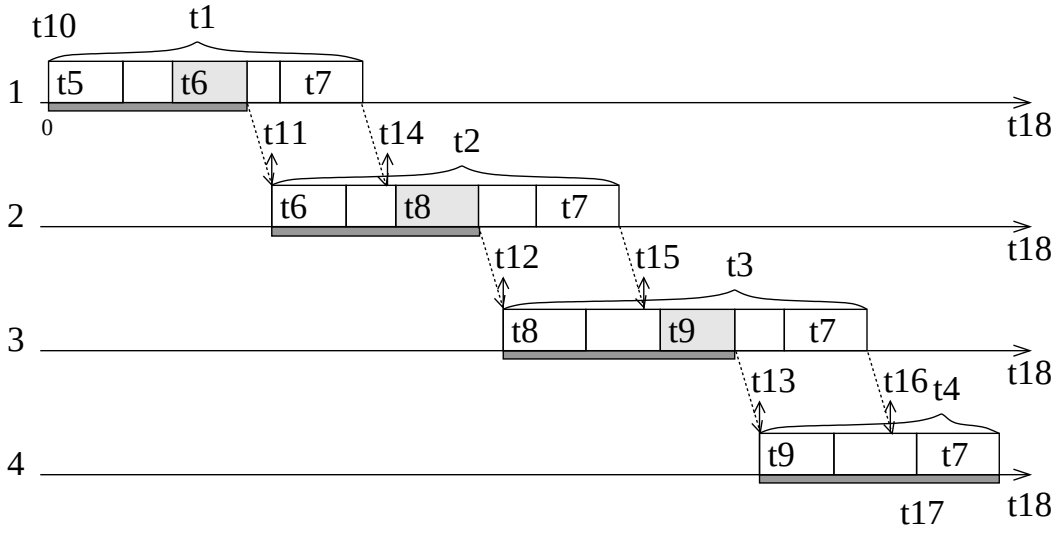


Figure B.1: Illustration of busy period

For a busy period bp^h, the first transmitted frame is denoted $f(h)$. The arrival of frame $f(h)$ is the starting instant M_i^h of the busy period.

At the source node 1, the starting time of bp^1 is taken as the time origin, i.e., $M_i^1 = 0$. At this node, the frame f_i could be delayed by frames of competing flows also emitted by the node 1 as long as these frames arrive no earlier than time 0 and no later than the arrival of frame f_i.

At the node 2, there are new incoming competing flows to cross $\mathcal{P}_i$. The starting time M_i^2 is the arrival time of the first frame coming from bp^1. The new incoming frames delay the

studied frame f_i if they arrive no earlier than M_i^2 and no later than $a_{f_i}^2$. These frames are transmitted before f_i and contribute to the busy period bp^2. This process propagates till the last visited node 4, and the end-to-end delay of f_i is marked as shadow bars in Figure B.1. The computation of this delay by the Trajectory approach is explained in the following paragraphs.

Since any frame of a competing flow τ_j arriving at the node $first_{j,i}$ no earlier than time $M_i^{first_{i,j}}$ and no later than time $a_{f_i}^{first_{j,i}}$ can delay f_i due to the FIFO scheduling, the maximum waiting delay generated by τ_j is obtained by considering maximum frames arriving during a maximized time interval $\left[M_i^{first_{j,i}}, a_{f_i}^{first_{j,i}}\right]$. In order to maximize the length of the busy period, it needs to minimize the value of M_i^h and maximized the value of $a_{f_i}^{first_{j,i}}$.

It has been shown in [MM06a] that a minimized M_i^h is achieved by the sum of the transmission time of the smallest frame at each visited node before arriving at node h plus the smallest link cost L_{min} of each crossed link. Therefore M_i^h is given by:

$$M_i^h = \sum_{h'=first_i}^{h-1} \Big(\min_{\substack{j \in [\![1,n]\!] \\ first_{j,i}=first_{i,j}}} (C_j^{h'}) + L_{min} \Big)$$

where $h-1$ means the previous node of h along path $\mathcal{P}_i$. Then $M_i^{first_{i,j}}$ is the earliest possible starting instant of the busy period $bp^{first_{j,i}}$.

Since $S_{max_i}^{first_{j,i}}$ is the maximum delay of frame f_i generated from its source node to node $first_{j,i}$, the upper bound $a_{f_i}^{first_{j,i}}$ is maximized by $t + S_{max_i}^{first_{j,i}}$, which is the latest possible arrival time of f_i at the node $first_{j,i}$.

Therefore, the length of the interval $\left[M_i^{first_{i,j}}, t + S_{max_i}^{first_{j,i}}\right]$ is maximized, which is illustrated in Figure B.2 as the shadow bar. As long as the frames of τ_j arrive at the node $first_{j,i}$ during the maximized interval, they could delay f_i.

With the maximized interval $\left[M_i^{first_{i,j}}, t + S_{max_i}^{first_{j,i}}\right]$, it needs to maximize the number of frames of flow τ_j which can arrive during this interval. Since the flow model of τ_j is known at its source node $first_j$, it needs to maximize a workload interval at $first_j$. The transmission of a frame of flow τ_j from its source node $first_j$ to the node $first_{j,i}$ requires time of at least $S_{min_j}^{first_{j,i}}$ and at most $S_{max_j}^{first_{j,i}}$, then the earliest generation time of a frame of flow τ_j at its source node is $M_i^{first_{i,j}} - S_{max_j}^{first_{j,i}} - J_j$ and the latest generation time is $t + S_{max_i}^{first_{j,i}} - S_{min_j}^{first_{j,i}}$ (Figure B.2). This workload interval from the earliest generation time to the latest generation time decides the maximized number of frames which could delay f_i at the node $first_{j,i}$, and it is represented by $t + A_{i,j}$ where:

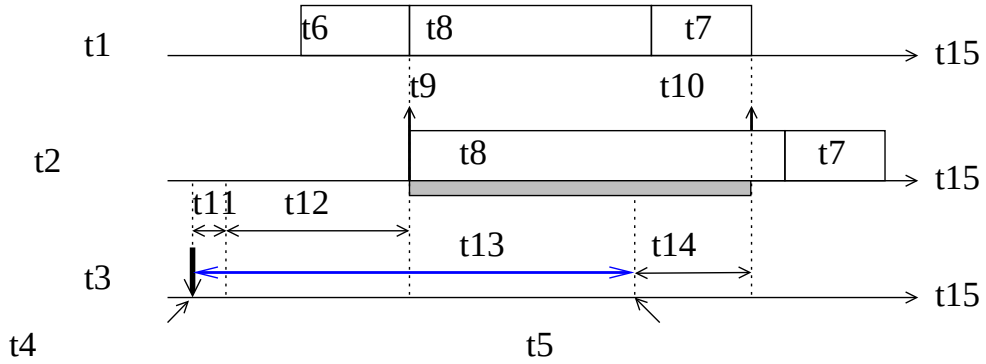


Figure B.2: Illustration of the workload interval $A_{i,j}$

$$A_{i,j} = S_{max_i}^{first_{j,i}} - S_{min_j}^{first_{j,i}} - M_i^{first_{i,j}} + S_{max_j}^{first_{i,j}} + J_j \tag{B.1}$$

With this workload interval, the maximum delay generated by a competing flow τ_j can be determined. Due to the FIFO scheduling, a frame can delay the studied frame f_i only once. Then its transmission is computed only once. Thus, the waiting delay caused by all the competing flows is maximized by:

$$\sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_j \cap \mathcal{P}_i \neq \emptyset}} (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j^{slow_{j,i}}$$

In order to ensure the maximum delay, $C_j^{slow_{j,i}}$ is considered in the computation since it is the largest transmission time of a frame of τ_j along the path $\mathcal{P}_i$.

When $j = i$, it means that frame f_i could be delayed by other frames from the same flow τ_i if these frames are pending in the output port buffer when f_i arrives at the output port. In that case, $A_{i,i} = J_i$ and $C_i^{slow_{i,i}} = C_i^{slow_i}$.

Besides the delay caused by the competing flows, there are other factors contributing to the end-to-end delay.

According to [MM06a], the frame $f(h+1)$ is counted twice as the transition cost from one busy period to the next one (frames f_2, f_3 and f_4 with shadow blocks in Figure B.1). In order to guarantee the end-to-end delay upper bound, the largest frame of each busy period bp^h (except the $slow_i$) is considered in the computation. Thus, this delay can be computed by:

$$\sum_{h \in \mathcal{P}_i / \{slow_i\}} (\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j^h))$$

Due to physical constraint, frame f_i crosses $|\mathcal{P}_i| - 1$ links. The cost to cross each link is between $[L_{min}, L_{max}]$. Then it generates an upper bounded link cost:

$$(|\mathcal{P}_i| - 1) \cdot L_{max}$$

With the FIFO scheduling, the transmission of a frame can no longer be delayed after it has started. Therefore, the latest starting time $W_{i,t}^{last_i}$ of frame f_i at its last visited output port $last_i$ is computed:

$$\begin{aligned} W_{i,t}^{last_i} = & \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j^{slow_{j,i}} \\ & + \sum_{h \in \mathcal{P}_i / \{slow_i\}} (\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j^h)) \\ & + (|\mathcal{P}_i| - 1) \cdot L_{max} \\ & - C_i^{last_i} \end{aligned}$$

The transmission time $C_i^{last_i}$ of f_i at the last visited node is subtracted because $W_{i,t}^{last_i}$ is the latest starting time.

Then the worst-case end-to-end delay upper bound of the flow τ_i is obtained by:

$$R_i = \max_{-J_i + \mathcal{B}_i^{slow} \geq t \geq -J_i} \{W_{i,t}^{last_i} + C_i^{last_i} - t\} \tag{B.2}$$

where $\mathcal{B}_i^{slow} = \sum_{j \in [\![1,n]\!]} \lceil \frac{\mathcal{B}_i^{slow}}{T_j} \rceil \cdot C_j^{slow_{j,i}}$. It is proved by Property 2 and Lemma 3 in [MM06a] and it limits the range of computation.

B.2 Application on a switched Ethernet network

The Trajectory approach has been applied to and optimized for the switched Ethernet in [BSF09, BSF10]. A real-time switched Ethernet network is composed of a set of source nodes

interconnected by switches. An illustrative example is given in Figure B.3. A source node manages a set of sporadic flows. Each source node has an input port and an output port. A switch is in charge of forwarding each incoming frame to the output port towards its destination according to a statical routing. There is a constant switching latency sl when a frame crosses a switch. Each switch has a set of input ports and a set of output port. Each input port of a source node or of a switch does not have a buffer, while each output port of a source node or of a switch has a buffer with FIFO scheduling.

An example flow τ_1 following path $\mathcal{P}_1 = \{N_1, S_1, S_2, N_4\}$ is studied. The transmission of a frame f_1 of τ_1 is shown in Figure B.4.

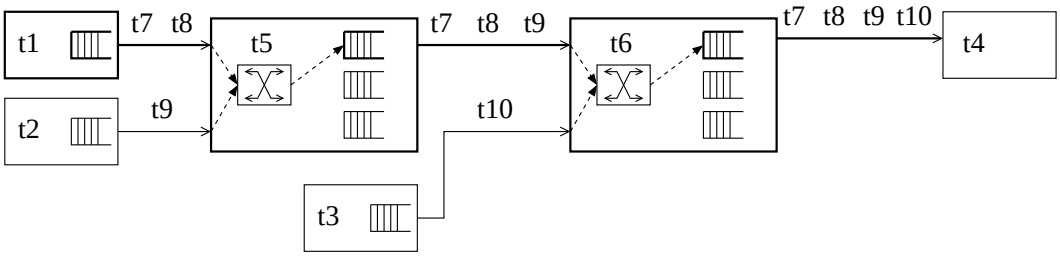


Figure B.3: An switched Ethernet network illustration

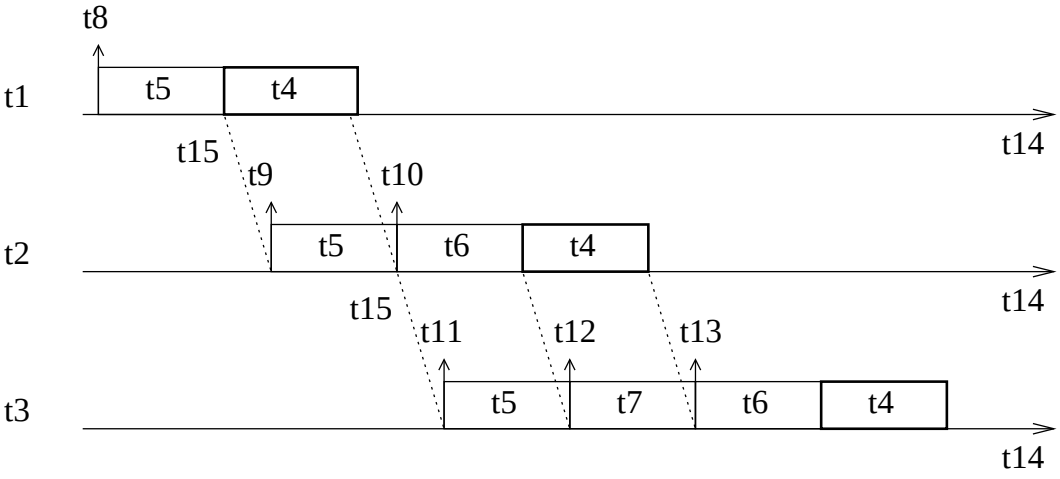


Figure B.4: The transmission process of frame f_1 of flow τ_1

Each node considered in the Trajectory approach corresponds to an output port crossed by the considered flow, since each output port is equipped with a FIFO buffer. Since the input of destination node is without FIFO buffer, it is not considered in the computation. Then $last_i$ is referred to the output pout of last visited switch. For τ_1 in Figure B.3, $first_1 = N_1$, $last_1 = S_2$

and $dest_1 = N_2$. The Trajectory approach considers only the output ports of N_1, S_1 and S_2 as shown in Figure B.4.

Each link considered in the Trajectory approach corresponds to a cross of a switch, and the link cost considered in the Trajectory approach is referred to a switching latency. Since the switching latency of the considered switch model is a constant value sl, then the computation for the switched Ethernet considers:

$$L_{max} = L_{min} = sl$$

A flow τ_i following a path $\mathcal{P}_i$ which includes $first_i$ and $dest_i$ crosses $|\mathcal{P}_i| - 2$ switches. Then the total switching latencies experienced by τ_i is:

$$(|\mathcal{P}_i| - 2) \cdot sl$$

Flow τ_1 in Figure B.3 crosses two switches, then it experiences the switching latency twice, as sl shown in Figure B.4.

A link of a switched Ethernet connects an input port and an output port and works in mode *full duplex*. Therefore, we have:

$$first_{i,j} = first_{j,i}, \quad last_{i,j} = last_{j,i}$$

All the links of the considered switched Ethernet are with the same bandwidth (transmission rate) 100 $Mbit/s$. Therefore there is no slowest node for flows, and we take:

$$slow_i = last_i, \quad C_i^h = C_i \text{ if } h \in \mathcal{P}_i$$

Therefore, for a frame f_i of a flow τ_i in a switched Ethernet network, the latest starting time $W_{i,t}^{last_i}$ of frame f_i at its last visited output port $last_i$ is computed:

$$
\begin{aligned}
W_{i,t}^{last_i} &= \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j \\
&+ \sum_{h \in \mathcal{P}_i / \{last_i, dest_i\}} (\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j)) \\
&+ (|\mathcal{P}_i| - 2) \cdot sl \\
&- C_i
\end{aligned} \tag{B.3}
$$

Then the worst-case end-to-end delay upper bound of the flow τ_i is obtained by:

$$
R_i = \max_{-J_i + \mathcal{B}_i \geq t \geq -J_i} \{W_{i,t}^{last_i} + C_i - t\} \tag{B.4}
$$

where $\mathcal{B}_i = \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} \lceil \frac{\mathcal{B}_i}{T_j} \rceil \cdot C_j$.

B.3 Integration of frame serialization

In [BSF10] an optimization of the Trajectory approach considering FIFO scheduling is proposed in order to take into account the fact that frames transmitted by the same input link are serialized and they cannot arrive at an output port at the same time.

The frame serialization exists at each switch h as illustrated in Figure B.5. Flows crossing an output port h are transmitted from $k_h + 1$ input links IP_k^h ($k \in [\![0, k_h]\!]$. For a frame f_i of flow τ_i, it crosses h in its path $\mathcal{P}_i$ from the input link IP_0^h to the output link OP^h. Then the input link IP_0^h transmits a frame sequence seq_0^h which contains f_i which will be transmitted in the busy period bp^h on the output link OP^h. There are other k_h input links IP_k^h ($k \in [\![1, k_h]\!]$) which transmits frame sequences seq_k^h ($k \in [\![1, k_h]\!]$) to the output link OP^h. These frames can delay f_i in the busy period bp^h. The duration of sequence seq_k^h without its first frame is denoted l_k^h ($k \in [\![0, k_h]\!]$). The first and last frame arrivals at each busy period bp^h are $f(h)$ and $p(h)$. The first frame arrival of each input link IP_k^h ($k \in [\![0, k_h]\!]$) is denoted p_k ($k \in [\![0, k_h]\!]$).

The Trajectory approach developed in [MM06a] maximizes the delay of τ_i generated at the output port of h by postponing the first frame arrival p_k ($k \in [\![1, k_h]\!]$) of each input link IP_k^h

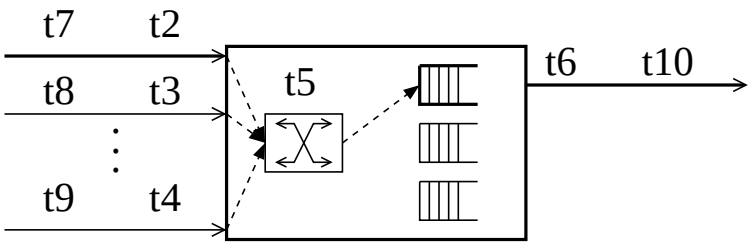


Figure B.5: Illustration on a node h with an output link OP^h and $k_h + 1$ input link IP^h_k

($k \in [\![1, k_h]\!]$) till the first frame arrival $p_0 = p(h-1)$ of IP^h_0, which is:

$$a^h_{p_k} = a^h_{p(h-1)}, \forall k \in [\![1, k_h]\!]$$

In [BSF10], an optimization on the Trajectory approach has been proposed. An illustration of this optimization is shown in Figure B.6. Due to the FIFO scheduling at the output buffer of h, any frame arriving later than the arrival of f_i (θ in Figure B.6) at h cannot delay f_i. Then the delay of τ_i is also maximized by postponing the last frame arrival of each frame sequence seq^h_k from each input link IP^h_k ($k \in [\![1, k_h]\!]$) till the arrival of f_i from IP^h_0. This time is marked as θ in Figure B.6. In this case, there can be a temporal distance from $a^h_{p_k}$ to $a^h_{p(h-1)}$ for each input link IP^h_k ($k \in [\![1, k_h]\!]$). The largest minimized distance for k_h input links is denoted by term $\Delta^h_{i,t}$, which is illustrated in Figure B.6. Any frame transmitted during this term does not delay f_i at h and their transmission time should be subtracted from the delay.

For flow τ_i, the frame serialization does not exist at the source node $first_i$ and the destination node $dest_i$ since $first_i$ does not have input links and $dest_i$ does not have output link. It does exist for the other visited output ports along the path $\mathcal{P}_i$, then each term $\Delta^h_{i,t}$ along the path $\mathcal{P}_i / \{first_i, dest_i\}$ needs to be subtracted from the delay:

$$- \sum_{h \in \mathcal{P}_i / \{first_i, dest_i\}} (\Delta^h_{i,t})$$

In order to maximize $W^{last_i}_{i,t}$, it needs to minimize term $\Delta^h_{i,t}$. According to [BSF10], this is achieved when the first frame of seq^h_0 is the smallest one of seq^h_0, which gives $\max(l^h_0)$; while the first frame of each seq^h_k ($k \in [\![1, k_h]\!]$) is the largest one of the corresponding seq^h_k, which gives $\min(l^h_k)$. In order to reduce pessimism in the computation, the largest $\min(l^h_k)$ ($k \in [\![1, k_h]\!]$) is considered. Then the minimum value of term $\Delta^h_{i,t}$ is given by the maximum value between 0

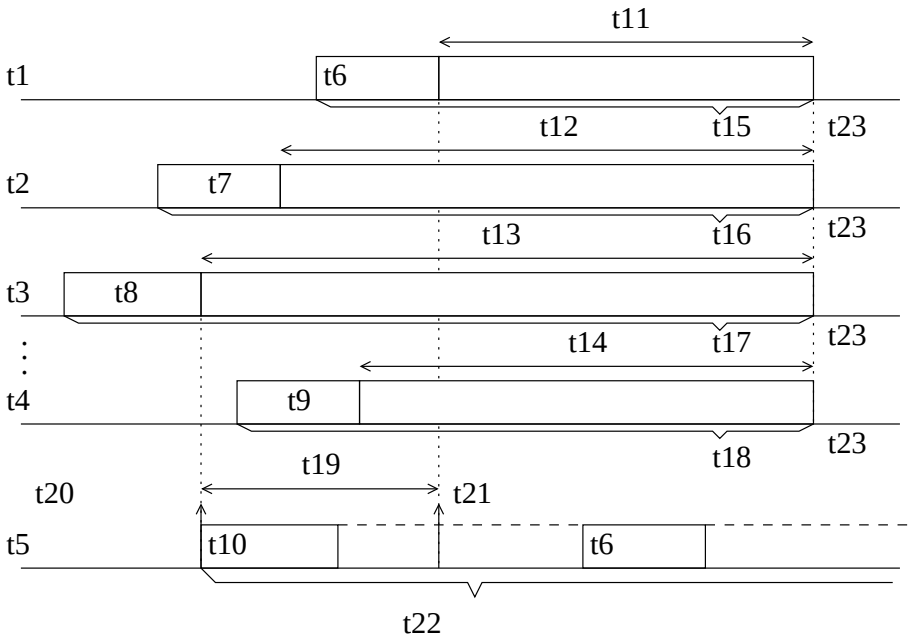


Figure B.6: Illustration on term $\Delta_{i,t}^{h}$

and

$$\max_{k\in[\![1,k_h]\!]}(\min(l_k^h)) - \max(l_0^h) \tag{B.5}$$

As illustrated in Figure B.6, suppose that the frame $p(h-1)$ is the smallest one of seq_0^h, then the largest minimized l_k^h ($k \in [\![1, k_h]\!]$) is l_2^h. Therefore $\Delta_{i,t}^h$ is given by:

$$\Delta_{i,t}^h = l_2^h - l_0^h$$

Therefore, for a frame f_i of a flow τ_i in a switched Ethernet network, the latest starting time $W_{i,t}^{last_i}$ of frame f_i at its last visited output port $last_i$ is computed by:

$$
\begin{aligned}
W_{i,t}^{last_i} \quad = \quad & \sum_{\substack{j \in [\![1,n]\!] \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} (1 + \lfloor \frac{t + A_{i,j}}{T_j} \rfloor)^+ \cdot C_j \\
& + \sum_{h \in \mathcal{P}_i / \{last_i, dest_i\}} (\max_{\substack{j \in [\![1,n]\!] \\ h \in \mathcal{P}_j}} (C_j)) \\
& + (|\mathcal{P}_i| - 2) \cdot sl \\
& - \sum_{h \in \mathcal{P}_i / \{first_i, dest_i\}} (\Delta_{i,t}^h) \\
& - C_i
\end{aligned}
$$

The worst-case delay upper bound of flow τ_i is then computed by Equation B.2.

Bibliography

[ACC04] *ARINC 429*, ACCE Std. 429, 2001-2004.

[ACC08] *ARINC 664*, ACCE Std. 664, 2002-2008.

[AD94] R. Alur and D. Dill, "A theory of timed automata," *Theoretical computer science*, vol. 126, no. 2, pp. 183–235, 1994.

[AM07] F. Arjmandi and B. Moshiri, "Fieldbus interoperability on Ethernet," in *Proc. IEEE Int. Conf. on Industrial Informatics*, Vienna, Autriche, June 2007, pp. 213–217.

[ASEF12] M. Adnan, J.-L. Scharbarg, J. Ermont, and C. Fraboul, "An improved timed automata approach for computing exact worst-case delays of AFDX sporadic flows," in *Emerging Technologies Factory Automation (ETFA), 2012 IEEE 17th Conference on*, Krakow, Poland, 2012, pp. 1–8.

[ASF11] M. Adnan, J.-L. Scharbarg, and C. Fraboul, "Minimizing the search space for computing exact worst-case delays of AFDX periodic flows," in *Proc. 6th IEEE Int. Symposium on Industrial Embedded Systems (SIES)*. IEEE, 2011, pp. 294–301.

[Aut] S. Automation, "Modbus messaging on TCP/IP implementation guide, may 2002."

[BB08] S. Baruah and A. Burns, "Quantifying the sub-optimality of uniprocessor fixed-priority scheduling," in *Proc. 16th Int. Conf. on Real Time Network and Systems (RTNS)*, Rennes, France, Oct. 2008.

[BBF+10] B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and P. Schnoebelen, *Systems and software verification: model-checking techniques and tools*. Springer Publishing Company, Incorporated, 2010.

[Bou98] J.-Y. L. Boudec, "Application of network calculus to guaranteed service networks," *IEEE Trans. Inf. Theory*, vol. 44, no. 3, pp. 1087–1096, May 1998.

[BSF09] H. Bauer, J.-L. Scharbarg, and C. Fraboul, "Applying and optimizing trajectory approach for performance evaluation of AFDX avionics network," in *Proc. IEEE Emerging Technologies and Factory Automation (ETFA)*, Marllorca, Spain, Sep. 2009, pp. 1–8.

[BSF10] ——, "Improving the worst-case delay analysis of an AFDX network using an optimized Trajectory approach," *IEEE Trans. Ind. Informat.*, vol. 6, no. 4, pp. 521–533, 2010.

[BSF12] ——, "Applying Trajectory approach with static priority queuing for improving the use of available AFDX resources," *Real-time systems*, vol. 48, no. 1, pp. 101–133, Jan. 2012.

[BT01] J.-Y. L. Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, ser. Lecture Notes in Computer Science. Springer-Verlag, 2001, vol. 2050, iSBN: 3-540-42184-X.

[C+95] I. E. Commission *et al.*, "Electrical equipment of industrial machines-serial data link for real-time communications between controls and drives (SERCOS)," 1995.

[CAN] *CAN Specification version 2.0.* Postfach 30 02 40, D-70442 Stuttgart Germany: Robert Bosch GmbH.

[CEL05] V. Cholvi, J. Echague, and J. LeBoudec, "On the feasible scenarios at the output of a FIFO server," *Communications Letters, IEEE*, vol. 9, no. 5, pp. 397–399, 2005.

[Cru91] R. Cruz, "A calculus for network delay, part 2," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 132–141, Jan. 1991.

[CSEF06] H. Charara, J.-L. Scharbarg, J. Ermont, and C. Fraboul, "Method for bounding end-to-end delays on an AFDX network," in *Proc. the 18th Euromicro Conference on Real-Time Systems (ECRTS)*, Germany, July 2006, pp. 192–202.

[DBBL07] R. Davis, A. Burns, R. Bril, and J. Lukkien, "Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised," *Real-Time Systems*, vol. 35, no. 3, pp. 239–272, 2007.

[Dec05] J.-D. Decotignie, "Ethernet-based real-time and industrial communications," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1102–1117, 2005.

[DH03] O. Dolejs and Z. Hanzalek, "Simulation of Ethernet for real-time applications," in *Industrial Technology, 2003 IEEE International Conference on*, vol. 2. IEEE, 2003, pp. 1018–1021.

[EL02] J. Eidson and K. Lee, "IEEE 1588 standard for a precision clock synchronization protocol for networked measurement and control systems," in *Sensors for Industry Conference, 2002. 2nd ISA/IEEE.* Ieee, 2002, pp. 98–105.

[EP] [Online]. Available: http://www.ethernet-powerlink.org

[ESF06] J. Ermont, J.-L. Scharbarg, and C. Fraboul, "Worst-case analysis of a mixed CAN/switched Ethernet architecture," in *Int. Conf. on Real-Time and Network Systems (RTNS)*, Poitiers, 2006, pp. 06–31.

[ESMGZ08] H. El-Sayed, A. Mellouk, L. George, and S. Zeadally, "Quality of service models for heterogeneous networks: overview and challenges," *annals of telecommunications*, vol. 63, no. 11-12, pp. 639–668, 2008.

[Eth04a] R.-T. Ethernet, "Ethernet Control Automation Technology (ETHERCAT): Proposal for a publicly available specification for real-time Ethernet," 2004.

[Eth04b] ——, "P-NET on IP: Proposal for a publicly available specification for real-time Ethernet," 2004.

[Eth04c] ——, "PROFINET IO: Proposal for a publicly available specification for real-time Ethernet," 2004.

[Eth04d] ——, "TCnet (Time-critical Control network): Proposal for a publicly available specification for real-time Ethernet," 2004.

[Fel04] J. Feld, "PROFINET-scalable factory communication for all applications," in *Factory Communication Systems, 2004. Proceedings. 2004 IEEE International Workshop on.* IEEE, 2004, pp. 33–38.

[Fel05] M. Felser, "Real-time Ethernet - industry prospective," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1118–1129, May 2005.

[FFG06] F. Frances, C. Fraboul, and J. Grieu, "Using network calculus to optimize the AFDX network," in *Proc. 3rd Embedded Real Time Software Conference (ERTS)*, Toulouse, Jan. 2006.

[FJJ09] X. Fan, M. Jonsson, and J. Jonsson, "Guaranteed real-time communication in packet-switched networks with FCFS queuing," *Computer networks*, vol. 53, no. 3, pp. 400–417, 2009.

[GGN06] M. Grenier, J. Goossens, and N. Navet, "Near-optimal fixed priority preemptive scheduling of offset free systems," in *Proc. 14th Int. Conf. on Real Time Network and Systems (RTNS)*, Poitiers, France, May 2006.

[GHN08] M. Grenier, L. Havet, and N. Navet, "Pushing the limits of CAN - scheduling frames with offsets provides a major performance boost," in *4th European Congress on Embedded Real Time Software*, Toulouse, France, Jan. 2008.

[Goo03] J. Goossens, "Scheduling of offset free systems," *Real-Time Systems*, vol. 24, no. 2, pp. 239–258, Mar. 2003.

[GRD02] J.-P. Georges, E. Rondeau, and T. Divoux, "Evaluation of switched Ethernet in an industrial context by using the network calculus," in *4th IEEE Int. Workshop on Factory Communication Systems(WFCS)*, Vősterås, Sweden, Aug. 2002, pp. 19–26.

[Gri04] J. Grieu, "Analyse et évaluation de techniques de commutation Ethernet pour interconnexion des systèmes avioniques," Ph.D. dissertation, Institut National Polytechnique de Toulouse-ENSEEIHT, 2004.

[HNNP02] H. Hansson, T. Nolte, C. Norstrom, and S. Punnekkat, "Integrating reliability and timing analysis of CAN-based systems," *Industrial Electronics, IEEE Transactions on*, vol. 49, no. 6, pp. 1240–1250, 2002.

[JB04] D. Jansen and H. Buttner, "Real-time Ethernet: the EtherCAT solution," *Computing and Control Engineering*, vol. 15, no. 1, pp. 16–21, 2004.

[JNTW02] J. Jasperneite, P. Neumann, M. Theis, and K. Watson, "Deterministic real-time communication with switched Ethernet," in *4th IEEE Int. Workshop on Factory Communication Systems(WFCS)*, Vősterås, Swede, Aug. 2002, pp. 11–18.

[KAGS05] H. Kopetz, A. Ademaj, P. Grillinger, and K. Steinhammer, "The Time-Triggered Ethernet (TTE) design," in *Object-Oriented Real-Time Distributed Computing,*

2005. ISORC 2005. Eighth IEEE International Symposium on. IEEE, 2005, pp. 22–33.

[KAS08] S. Kollmann, K. Albers, and F. Slomka, "Effects of simultaneous stimulation on the event stream densities of fixed-priority systems," in *Int. Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, Edinburgh, June 2008, pp. 353–360.

[LH04a] J. Loeser and H. Haertig, "Low-latency hard real-time communication over switched Ethernet," in *Proc. 16th Euromicro Conference on Real-Time Systems(ECRTS).* IEEE, 2004, pp. 13–22.

[LH04b] ——, "Using switched Ethernet for hard real-time communication," in *Int. Conf. on Parallel Computing in Electrical Engineering.* IEEE, 2004, pp. 349–353.

[LLB06] J. Lorente, G. Lipari, and E. Bini, "A hierarchical scheduling model for component-based real-time systems," in *Proc. 14th Int. Workshop on Parallel and Distributed Real-Time Systems*, Island of Rhodes, Greece, Apr. 2006.

[LMS05] L. Lenzini, E. Mingozzi, and G. Stea, "Delay bounds for FIFO aggregates: a case study," *Computer Communications*, vol. 28, no. 3, pp. 287–299, 2005.

[LR93] G. L. Lann and N. Rivierre, "Real-time communications over broadcast networks: the CSMA-DCR and the DOD-CSMA-CD protocols," 1993.

[MFF07] A. Mifdaoui, F. Frances, and C. Fraboul, "Real-time characteristics of switched Ethernet for "1553b"-embedded applications: Simulation and analysis," in *Industrial Embedded Systems, 2007. SIES'07. International Symposium on.* IEEE, 2007, pp. 33–40.

[MM06a] S. Martin and P. Minet, "Schedulability analysis of flows scheduled with FIFO: application to the expedited forwarding class," in *Proc. IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Rhodes Island, Greece, Apr. 2006, pp. 8–pp.

[MM06b] ——, "Worst case end-to-end response times of flows scheduled with FP/FIFO," in *Proc. IEEE Int. Conf. on Networking*, Mauritius, Apr. 2006, pp. 54–60.

[MMG05] S. Martin, P. Minet, and L. George, "End-to-end response time with fixed priority scheduling: Trajectory approach versus holistic approach," *International Journal of Communication Systems*, vol. 18, no. 1, pp. 37–56, 2005.

[MMG06] ——, "The Trajectory approach for the end-to-end response times with non-preemptive FP/EDF*," in *Software Engineering Research and Applications*. Springer, 2006, pp. 229–247.

[NS98] N. Navet and Y. Song, "Design of reliable real-time applications distributed over CAN (Controller Area Network)," in *INCOM98, IFAC 9th Symposium on Information Control in Manufacturing*, 1998.

[PAN04] *The Evolution of Copper Cabling Systems from Cat5 to Cat5e to Cat6*, PANDUIT Std., 2004.

[RGPH11] J. Rivas, J. Gutierrez, J. Palencia, and M. Harbour, "Schedulability analysis and optimization of heterogeneous EDF and FP distributed real-time systems," in *Proc. 23rd Euromicro Conference on Real-Time Systems (ECRTS)*, July 2011, pp. 195–204.

[SBF05] J.-L. Scharbarg, M. Boyer, and C. Fraboul, "Interconnecting CAN busses via an Ethernet backbone," in *Proc. 6th Int. Conf. on Fieldbus Systems and their Applications (IFAC)*, Puebla, Mexico, Nov. 2005.

[Sch01] V. Schiffer, "The CIP family of fieldbus protocols and its newest member-Ethernet/IP," in *Emerging Technologies and Factory Automation, 2001. Proceedings. 2001 8th IEEE International Conference on.* IEEE, 2001, pp. 377–384.

[Sch03] J. Schmitt, "On average and worst case behaviour in non-preemptive priority queueing," *SIMULATION SERIES*, vol. 35, no. 4, pp. 197–204, 2003.

[SF07] J.-L. Scharbarg and C. Fraboul, "Simulation for end-to-end delays distribution on a switched Ethernet," in *Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference on.* IEEE, 2007, pp. 1092–1099.

[SKS02] Y. Song, A. Koubaa, and F. Simonot, "Switched Ethernet for real-time industrial communication: Modelling and message buffering delay evaluation," in *Factory Communication Systems, 2002. 4th IEEE International Workshop on.* IEEE, 2002, pp. 27–35.

[SL03] I. Shin and I. Lee, "Periodic resource model for compositional real-time guarantees," in *Real-Time Systems Symposium RTSS.* IEEE, 2003, pp. 2–13.

[SL04] ——, "Compositional real-time scheduling framework," in *Proc. 25th IEEE Int. Conf; on Real-Time Systems Symposium (RTSS'04)*, Dec. 2004, pp. 57–67.

[Son01] Y. Song, "Time constrained communication over switched Ethernet," in *IFAC international conference on fieldbus systems and their applications*, 2001, pp. 152–159.

[SRF09] J.-L. Scharbarg, F. Ridouard, and C. Fraboul, "A probabilistic analysis of end-to-end delays on an AFDX avionic network," *IEEE Trans. Ind. Informat.*, vol. 5, no. 1, pp. 38–49, Feb. 2009.

[TB94] K. Tindell and A. Burns, "Guaranteeing message latencies on Controller Area Network (CAN)," in *Proceedings of the 1st International CAN Conference*. Citeseer, 1994.

[TBW95a] K. Tindell, A. Burns, and A. Wellings, "Analysis of hard real-time communications," *Real-Time Systems*, vol. 9, no. 2, pp. 147–171, 1995.

[TBW95b] ——, "Calculating Controller Area Network (CAN) message response times," *Control Engineering Practice*, vol. 3, no. 8, pp. 1163–1169, 1995.

[TC94] K. Tindell and J. Clark, "Holistic schedulability analysis for distributed hard real-time systems," *Microprocessing and microprogramming*, vol. 40, no. 2, pp. 117–134, 1994.

[THW94] K. Tindell, H. Hansson, and A. Wellings, "Analysing real-time communications: Controller Area Network (CAN)," in *Real-Time Systems Symposium, 1994., Proceedings.* IEEE, 1994, pp. 259–263.

[TV99] E. Tovar and F. Vasques, "Real-time fieldbus communications using Profibus networks," *Industrial Electronics, IEEE Transactions on*, vol. 46, no. 6, pp. 1241–1251, 1999.

[Wor] [Online]. Available: http://www.worldfip.org