Deficit Round-Robin: A Second Network Calculus Analysis

Seyed Mohammadhossein Tabatabaee, Member, IEEE, and Jean-Yves Le Boudec, Fellow, IEEE

Abstract—Deficit Round-Robin (DRR) is a widespread scheduling algorithm that provides fair queuing with variable-length packets. Bounds on worst-case delays for DRR were found by Boyer et al., who used a rigorous network calculus approach and characterized the service obtained by one flow of interest by means of a convex strict service curve. These bounds do not make any assumptions on the interfering traffic hence are pessimistic when the interfering traffic is constrained by some arrival curves. For such cases, two improvements were proposed. The former, by Soni et al., uses a correction term derived from a semi-rigorous heuristic; unfortunately, these bounds are incorrect, as we show by exhibiting a counter-example. The latter, by Bouillard, rigorously derive convex strict service curves for DRR that account for the arrival curve constraints of the interfering traffic. In this paper, we improve on these results in two ways. First, we derive a non-convex strict service curve for DRR that improves on Boyer et al. when there is no arrival constraint on the interfering traffic. Second, we provide an iterative method to improve any strict service curve (including Bouillard’s) when there are arrival constraints for the interfering traffic. As of today, our results provide the best-known worst-case delay bounds for DRR. They are obtained by using the method of the pseudo-inverse.

Index Terms—Deficit round-robin, delay bound, worst-case delay, network calculus, strict service curve, deterministic networking.

I. INTRODUCTION

DEFICIT Round-Robin (DRR) [1] is a scheduling algorithm that is often used for scheduling tasks, or packets, in real-time systems or communication networks. With DRR, every queue is associated with a static number, called quantum. Queues are visited round-robin (one after the other), and at every visit, receive service (measured in bits for communication networks, in seconds for task processing systems) up to the quantum value. Tasks or packets are of variable sizes and it may happen that, during one visit of the server, there remains at least one task or packet in the queue that cannot be served because the unused part of the quantum is positive but not large enough. In such a case, the unused part of the quantum (called the residual deficit) is carried over to the next round.

DRR shares resources flexibly (the amount of service reserved for one queue is proportional to its quantum) and efficiently (when a queue is idle, the server capacity is available to other queues). It is widely used as it has low complexity and very efficient implementations exist [2].

DRR can be applied to time sensitive networks, i.e., to communication networks where it is required to obtain bounds on worst-case delay (not on average). Here, flows with similar delay requirements are mapped to the same class, and every class corresponds to one DRR queue at every node. Furthermore, the traffic that every flow can send is limited at the source by an arrival curve constraint, i.e., a limit to the number of bits that can be sent over any time interval. Worst-case delay bounds for such a setting were obtained in [3]–[5] using various ad-hoc analyses. These results were improved in [6], where the authors obtain a strict service curve for DRR, i.e., a function that lower bounds the amount of service received by every DRR queue. Delay bounds are then derived by using Network Calculus formulas (see Section II-A). We call this method the strict service curve of Boyer et al.

In a DRR system, if a queue does not have enough traffic to use its quantum at every visit of the server, then the leftover capacity is automatically used to improve the service received by other queues. In a time-sensitive network, some or all interfering traffic is deterministic, and in normal operation, is limited at the source. There is interest in obtaining proven bounds for both the degraded operational mode (when some traffic classes misbehave) and for the non-degraded mode (when all time sensitive traffic satisfies its source constraints). The strict service curve of Boyer et al. does not make any assumptions on the interfering traffic. Hence, the resulting delay bounds are valid, even in degraded operational mode.

For the non-degraded operational mode, i.e., when arrival curve constraints can be assumed for interfering traffic, significantly smaller delay bounds were presented at a recent RTSS conference [7]. They use the result of [6], which is improved by what we call the correction term of Soni et al. Unfortunately, the method is semi-rigorous and cannot be fully validated. Indeed, our first contribution is to show that the correction term of Soni et al. is incorrect; we do so by exhibiting a counter-example that satisfies their assumptions and that has a larger delay (Section III). The main idea of the correction term of Soni et al. is the assumption that only packets of interfering flows arriving within a duration of the delay bound of Boyer et al. will get a chance to delay a given packet of the flow of interest; in our counter example, we observed that this assumption is incorrect, and all packets of interfering flows arriving within the global backlogged period might delay a packet of the flow of interest. Later, Bouillard, in [8], derived new strict service curves for DRR that account for the arrival curve constraints of the interfering
traffic and improve on the strict service curve of Boyer et al., hence on the delay bounds. These results are formally proven. They require that arrival curves are concave (which does not always hold, e.g., when sources are periodic).

Our next contribution is obtaining a better strict service curve for DRR when we do not take into account arrival curve constraint on interfering traffic, i.e., for degraded operational mode. To do so, we rely on the method of pseudo-inverse, as it enables us to capture all details of DRR; a similar method was used to obtain a strict service curve for Interleaved Weighted Round-Robin in [9]. We also provide simplified lower bounds that can be used when analytic, closed-form expressions are important. One such lower bound is precisely the strict service curve of Boyer et al. (Fig. 5), hence the worst-case delay bounds obtained with our strict service curve are guaranteed to be less than or equal to those of Boyer et al.

Our following contribution is a new iterative method for obtaining better strict service curves for DRR that account for the arrival curve constraints of interfering flows. Our method is rigorous and is based on pseudo-inverses and output arrival curves of interfering flows. We also provide simpler variants. Our method improves on any available strict service curves for DRR, hence, we always improve on Bouillard’s strict service curve. Furthermore, our method accepts any type of arrival curves, including non-concave ones (such as the stair function used with periodic flows), and can be applied to any type of curve. Further, our method accepts any type of arrival curves, including non-concave ones (see Section II-B).

The delay bounds obtained with our method are fully proven. Furthermore, we compute them for the same case studies as in Bouillard’s work [8] (one single server analysis) and as in Soni et al. [7] (including two illustration networks and an industrial-sized one). We find that they are smaller than Bouillard’s and the incorrect ones that use the correction term on interfered traffic). Hence as of today, it follows that our delay bounds are the best proven delay bounds for DRR, with or without constraints on interfering traffic.

The remainder of the paper is organized as follows. After giving some necessary background in Section II, we describe the counter example to Soni et al. in Section III. In Section IV, we present our new strict service curves for DRR, with no knowledge of interfering traffic. In Section V, we present our new strict service curves for DRR; they account for the interfering arrival curve constraints. In Section VI, we use numerical examples to illustrate the improvement in delay bounds obtained with our new strict service curves.

This work is the extended version of [10], which was presented at the RTAS conference, 2021. The conference version did not include a discussion of Bouillard’s service curve, which was published after the conference submission date.

## II. BACKGROUND

We consider a DRR system in the context of deterministic networking, and we are interested in the worst-case delays for flows, given arrival curve constraints on the flows. In order to do that, we use network calculus approach, where necessary background are explained in Section II-A. The DRR is explained in Section II-B. Lastly, in Sections II-C, II-D, and II-E, we explain the state-of-the-arts.

### A. Network Calculus Background

We use the framework of network calculus [11]–[13]. Let $\mathcal{F}$ denote the set of wide-sense increasing functions $f : \mathbb{R}^+ \rightarrow \mathbb{R}^+ \cup \{+\infty\}$. A flow is represented by a cumulative arrival function $\alpha \in \mathcal{F}$ and $A(t)$ is the number of bits observed on the flow between times $0$ and $t$. We say that a flow has $\alpha \in \mathcal{F}$ as arrival curve if for all $s \leq t, A(t) - A(s) \leq \alpha(t-s)$. An arrival curve $\alpha$ can always be assumed to be sub-additive, i.e., to satisfy $\alpha(s+t) \leq \alpha(s) + \alpha(t)$ for all $s, t$. A periodic flow that sends up to $\alpha$ bits every $b$ time units has, as arrival curve, the stair function defined by $\nu_{0,b}(t) = \frac{a}{b}$. Another frequently used arrival curve is the token-bucket function $\alpha = \gamma_{r,b}$, with rate $r$ and burst $b$, defined by $\gamma_{r,b}(t) = rt + b$ for $t > 0$ and $\gamma_{r,b}(t) = 0$ for $t = 0$ (see Fig. 1). All of these arrival curves are sub-additive.

Consider a system $S$ and a flow through $S$ with input and output functions $A$ and $D$; we say that $S$ offers $\beta \in \mathcal{F}$ as a strict service curve to the flow if the number of bits of the flow output by $S$ in any backlogged interval $(s,t)$ is $D(t) - D(s) \geq \beta(t-s)$. A strict service curve $\beta$ can always be assumed to be super-additive (i.e., to satisfy $\beta(s+t) \geq \beta(s) + \beta(t)$ for all $s, t$) and wide-sense increasing (otherwise, it can be replaced by its super-additive and non-decreasing closure [13]). A frequently used strict service curve is the rate-latency function $\beta_{R,T} \in \mathcal{F}$, with rate $R$ and latency $T$, defined by $\beta_{R,T}(t) = R[t-T]^+$, where we use the notation $[x]^+ = \max\{x, 0\}$. It is super-additive (see Fig. 1).

Assume that a flow, constrained by a sub-additive arrival curve $\alpha$, traverses a system that offers a strict service curve $\beta$ and that respects the ordering of the flow (per-flow FIFO). The delay of the flow is upper bounded by the horizontal deviation defined by $h(\alpha, \beta) = \sup_{t \geq 0} \{\inf\{d \geq 0 | \alpha(t) \leq \beta(t+d)\}\}$ (see Fig. 1). Also, the output flow is constrained by an arrival curve $\alpha^* = \alpha \odot \beta$ where $\odot$ is the deconvolution operation defined in the next paragraph. The computation of $h(\alpha, \beta)$ and $\alpha^*$ can be restricted to $t \in [0, t^*]$ for $t^* \geq \inf_{s \geq 0} \{s \leq \beta(s)\}$ (see Fig. 1). [13, Prop. 5.13], [14].

For $f$ and $g$ in $\mathcal{F}$, the min-plus convolution is defined by $(f \circ \otimes g)(t) = \inf_{s \leq t} \{f(t-s) + g(s)\}$ and the min-plus deconvolution by $(f \circ \div g)(t) = \sup_{s \geq t} \{f(t+s) - g(s)\}$ [11]–[13]. We will use the min-plus convolution of a stair function with a linear function, as shown in Fig. 2.

If a flow, with arrival curve $\alpha$ and a maximum packet size $l_{\text{max}}$, arrives on a link with a rate $c$, a better arrival curve for this flow at the output of the link is the min-plus convolution of $\alpha$.
and the function \( t \mapsto L^{\text{max}} + ct \); this is known as grouping (also known as line-shaping) and is also explained Section II-D (see Fig. 1).

The non-decreasing closure \( f_1 \) of a function \( f : \mathbb{R}^+ \to \mathbb{R}^+ \cup \{+\infty\} \) is the smallest function in \( \mathcal{F} \) that upper bounds \( f \) and is given by \( f_1(t) = \sup_{s \leq t} f(s) \). Also, the non-decreasing and non-negative closure \( [f]_+^\mathcal{F} \) of \( f \) is the smallest non-negative function in \( \mathcal{F} \) that upper bounds \( f \) (see Fig. 3 (a)).

The lower pseudo-inverse \( f^\dagger \) of a function \( f \in \mathcal{F} \) is defined by \( f^\dagger(y) = \inf \{ x \mid f(x) \geq y \} = \sup \{ x \mid f(x) < y \} \) (see Fig. 3 (b)).

and satisfies [15, Sec. 10.1]:

\[
\forall x, y \in \mathbb{R}^+, y \leq f(x) \Rightarrow x \geq f^\dagger(y) \tag{1}
\]

The network calculus operations can be automated in tools such as RealTime-at-Work (RTaW) [16], an interpreter that provides efficient implementations of min-plus convolution, min-plus deconvolution, non-decreasing closure, horizontal deviation, the composition of two functions, and a maximum and minimum of functions for piecewise-linear functions. All computations use infinite precision arithmetic (with rational numbers).

\section{Deficit Round-Robin}

A DRR subsystem serves \( n \) inputs, has one queue per input, and uses Algorithm 1 for serving packets. Each queue \( i \) is assigned a quantum \( Q_i \). DRR runs an infinite loop of rounds. In one round, if queue \( i \) is non-empty, a service for this queue starts and its deficit is increased by \( Q_i \). The service ends when either the deficit is smaller than the head-of-the-line packet or the queue becomes empty. In the latter case, the deficit is set back to zero. The \texttt{send} instruction is assumed to be the only one with a non-null duration. Its actual duration depends on the packet size but also on the amount of service available to the entire DRR subsystem.

\begin{algorithm}
\caption{Deficit Round-Robin}
\textbf{Input:} Integer quantum \( Q_1, Q_2, \ldots, Q_n \)
\textbf{Data:} Integer deficits: \( d_1, d_2, \ldots, d_n \)
\begin{algorithmic}
\STATE for \( i \leftarrow 1 \) to \( n \) do
\STATE \( d_i \leftarrow 0 \);
\ENDFOR
\WHILE True do
\FOR \( i \leftarrow 1 \) to \( n \) do
\IF (not empty(i)) \texttt{send(head(i)); removeHead(i);}
\ENDIF
\ENDFOR
\WHILE A service for queue \( i \) ends.
\FOR \( i \leftarrow 1 \) to \( n \) do
\IF (empty(i)) \( d_i \leftarrow 0 \);
\ENDIF
\ENDFOR
\ENDWHILE
\end{algorithmic}
\end{algorithm}

In [6] as in much of the literature on DRR, the set of packets that use a given queue is called a flow; a flow may however be an aggregate of multiple flows, called micro-flows [17] and an aggregate flow is called a class in [7]. In this paper, and in order to be consistent with the network calculus conventions, we use the former terminology and consider that a DRR input corresponds to one flow. When comparing our results to [7], the reader is invited to remember that a DRR flow in this paper corresponds to a DRR class in [7].

The DRR subsystem is itself placed in a larger system and can compete with other queuing subsystems. A common case is when the DRR subsystem is at the highest priority on a non-preemptive server with line rate \( c \). Due to non-preemption, the service offered to the DRR subsystem might not be instantly available. This can be modelled by means of a rate-latency strict service curve (see Section II-A for the definition), with rate \( c \) and latency \( L^{\text{max}}/c \) where \( L^{\text{max}} \) is the maximum packet size of lower priority. If the DRR subsystem is not at the highest priority level, this can be modelled with a more complex strict service curve [13, Section 8.3.2]. This motivates us to assume that the aggregate of all flows in the DRR subsystem receives a strict service curve \( \beta \), which we call “aggregate strict service curve”. If the DRR subsystem has exclusive access to a transmission line of rate \( c \), then \( \beta(t) = ct \) for \( t \geq 0 \). We assume that \( \beta(t) \) is finite for every (finite) \( t \). Note that the aggregate strict service curve \( \beta \), which models the service offered to the aggregate of all flows, should not be confused with the strict service curves such as \( \beta^{\text{min}} \), which
caused by an interfering flow. Here, we use the language of communication networks, but the results equally apply to real-time systems: Simply map flow to task, map packet to job, map packet size to job-execution time, and map strict service curve to “delivery curve” \cite{18, 19}.

C. Strict Service Curve of Boyer et al.

The strict service curve of Boyer et al. for DRR is given in \cite{6}, and we rewrite it using our notation. For flow $i$, let $C_{\text{Boyer-et-al}}^{\text{max}}$ be its maximum residual deficit, defined by $d_{i}^{\text{max}} = t_{i}^{\text{max}} - \varepsilon$ where $t_{i}^{\text{max}}$ is an upper bound on the packet size and $\varepsilon$ is the smallest unit of information seen by the scheduler (e.g., one bit, one byte, one 32-bit word, ...). Also, let $Q_{i}^{\text{tot}} \equiv \sum_{j=1}^{n} Q_{j}$. Then, for every flow $i$, their strict service curve is the rate-latency service curve $\beta_{i} = \min \{ t_{i}^{\text{max}}, Q_{i}^{\text{tot}} \} / Q_{i}$ and latency $T_{i} = \sum_{j \neq i} d_{j}^{\text{max}} (1 + \frac{C_{j}^{\text{Boyer-et-al}}}{Q_{j}}) \sum_{j \neq i} Q_{j}$ (see Section II-A for the definition of a rate-latency function). This strict-rate-latency is illustrated in Fig. 5 (the red curve); Note that the latency is equal to the maximum service interruption plus additional terms. As Boyer et al. finds a rate-latency with the maximum rate, they were forced to take a latency larger than the maximum service interruption; this explains why it is not optimal.

D. Correction Term of Soni et al.

When interfering flows are constrained by some arrival curves, Soni et al. give a correction term that improves the obtained delay bounds using the strict service curve of Boyer et al. in \cite{7}, which we now rewrite using our notation. Assume that every flow $i$ has an arrival curve $\alpha_{i}$, and the server is a constant-rate server with a rate equal to $c$. Let $D_{1}^{\text{Boyer-et-al}}$ be the network calculus delay bound for flow $i$ obtained by combining $\alpha_{i}$ with the strict service curve of Boyer et al., as explained in Section II-C. The delay bound proposed in \cite{7} is $D_{i}^{\text{Soni-et-al}} = D_{i}^{\text{Boyer-et-al}} - c \alpha_{i}$ with

$$C_{\text{i}}^{\text{Soni-et-al}} = \sum_{j \neq i} \frac{S_{j}(D_{j}^{\text{Boyer-et-al}}) - \alpha_{j}(D_{j}^{\text{Boyer-et-al}})}{c}$$

where $S_{j}(t) \equiv (Q_{j} + d_{j}^{\text{max}}) 1_{t \geq h_{j}} + Q_{j}(1 + \frac{c(t - H_{j})}{Q_{j}}) 1_{t \geq H_{j}}$, $h_{i} = \frac{\sum_{j \neq i} Q_{j} + d_{j}^{\text{max}}}{c}$, and $H_{j} = h_{i} + \frac{Q_{j} - Q_{j}^{\text{max}}}{c} \sum_{j \neq i} Q_{j}$. The correction term is obtained by subtracting two terms: The former, function $S_{j}$, gives the maximum possible interference caused by an interfering flow $j$ in a backlogged period of the flow of interest $i$ and is derived from a detailed analysis of DRR; and the latter gives the effective interference caused by an interfering flow $j$ in a backlogged period of the flow of interest $i$, given the knowledge of an arrival curve of that interfering flow, $\alpha_{j}$.

Two additional improvements are used in \cite{7}. The former, called grouping, uses the fact that, if a collection of flows is known to arrive on the same link, the rate limitation imposed by the link can be used to derive, for the aggregate flow, an arrival curve that is smaller than the sum of arrival curves of the constituent flows (as explained in Section II-A). This improvement is also known under the name of line shaping and is used, for example, in \cite{20}–\cite{22}. The other improvement, called offsets, uses the fact that, if several periodic flows have the same source and if their offsets are known, the temporal separation imposed by the offsets can be used to compute, for the aggregate flow, an arrival curve that is also smaller than the sum of arrival curves of the constituent flows (the latter would correspond to an adversarial choice of the offsets). Both improvements reduce the arrival curves, hence the delay bounds. Note that both improvements are independent of the correction term (and, unlike the correction term, are correct); they can be applied to any method used to compute delay bounds, as we do in Section VI.

E. Bouillard’s Strict Service Curves

A new method to compute strict service curves for DRR that account for the interfering arrival curve constraints was recently presented in \cite{8}; the method works only when arrival curves are concave and the aggregate strict service curve is convex, and it improves on the strict service curve of Boyer et al. Specifically, in \cite[Theorem 1]{8}, for the flow of interest $i$, there exists non-negative numbers $H_{j}$ for any $j \in \{1, \ldots, n\} \setminus \{i\}$ such that $\beta_{i}^{\text{Boisson}}$ is given by

$$\beta_{i}^{\text{Boisson}} = \sup_{j \subseteq \{1, \ldots, n\} \setminus \{i\}} \frac{Q_{j}}{\sum_{j \neq i} Q_{j}} \left[ \beta - \sum_{j \neq i} \alpha_{j} - H_{j} \right]^{+}$$

where an inductive procedure is presented to compute $H_{j}$. We call these Bouillard’s strict service curves.

III. COUNTER EXAMPLE TO THE CORRECTION TERM OF SONI ET AL.

In this section, we show that the delay bound of Soni et al., namely the correction term given in equation (14) in \cite{7}, rewritten using our notation in (2), is invalid. For flow 2 in a system, we denote the delay bound of Soni et al. by $D_{2}^{\text{Soni-et-al}}$, and we denote the delay experienced by a packet of flow 2 in the trajectory scenario by $D_{2}^{\text{TS}}$.

A. System Parameters

Consider a constant-rate server, with a rate equal to $c$, that uses the DRR scheduling policy. All flows have packet sizes of constant size $l$, and have quanta $Q_{1} = 10l$, $Q_{2} = 100l$, and $Q_{3} = 5l$.

Each flow is constrained by a token-bucket arrival curve:

1. $\alpha_{1}(t) = \gamma_{r_{1}, b_{1}}$ with $0 \leq r_{1} < Q_{1}/c$ and $b_{1} = 20l$.
2. $\alpha_{2}(t) = \gamma_{r_{2}, b_{2}}$ with $r_{2} = 0.86c$ and $b_{2} = l$.
3. $\alpha_{3}(t) = \gamma_{r_{3}, b_{3}}$ with $r_{3} = 0.0401c$ and $b_{3} = l$.

Assuming a token-bucket $\gamma_{r, b}$, defined in Section II-A, for a flow implies that this flow has a minimum packet-arrival time equal to $\frac{b}{c}$. Also, observe that $r_{i} < \frac{Q_{i}}{c}$ for $i = 1, 2, 3$. We compute the delay bound of Soni et al. for flow 2, as explained in Section II-D, and we obtain $D_{2}^{\text{Soni-et-al}} = 14.03383\frac{l}{c} - 1.236215\frac{l}{c}$.

B. Trajectory Scenario

We now construct a possible trajectory for our system. First, we give the inputs of our three flows. All queues are empty, and the server is idle at time $t = 0$. Then,
Then, for the output, we have the following:

1) Flow 1 arrives first and has 20 ready packets. As its deficit was zero before this service and $Q_1 = 10l$, the server serves 10 packet of this flow. The end of the service for flow 1 is $t_1 = 10l$ (the first yellow part in Fig. 4).

2) Then, there is an emission opportunity for flow 2 and $A_2(t_1) = 9.6l$, which means flow 2 has 9 ready packets at time $t_1$. The server starts serving packets of this flow. At the end of service of these first 9 packets, at $t_2 = 19.6l$, flow 2 has another 8 ready packets; hence, the server still serves packets of flow 2. This continues and 62 packets of flow 2 are served in this emission opportunity; the emission opportunity ends at $t_4 = 72l$ (the first green part in Fig. 4).

3) Then, there is an emission opportunity for flow 3 and $A_3(t_4) = 3.8872l$, which means flow 3 has 3 ready packets at time $t_4$. At the end of service of these 3 packets, at $t_2 = 19.6l$, flow 2 has another 8 ready packets; hence, the server still serves packets of flow 2. This continues and 62 packets of flow 2 are served in this emission opportunity; the emission opportunity ends at $t_4 = 72l$ (the first green part in Fig. 4).

4) A packet for flow 2 arrives at $t_2^\text{arr} = 72 + 0.11l = 72.0932l$. This packet should wait for flow 3 and flow 1 to use their emission opportunities, and then it can be served. We call this the packet of interest of flow 2, for which we capture the delay (the first blue arrow, at $t_2^\text{dep}$, on Fig. 4).

5) For flow 1, again 10 packets are served (the second yellow part in Fig. 4).

6) Finally, the packet of interest is served and its departure time is $t_2^\text{dep} = 87l$.

It follows that the delay for the packet of interest is $D_2^{\text{Boyer-et-al}} = t_2^\text{dep} - t_2^\text{arr} = 154l - 0.11l = 14.907l$. Note that $D_2^{\text{Boyer-et-al}} > D_2^{\text{Soni-et-al}}$. To fix ideas, if $l = 100$ bytes and $c = 100$ Mbps, the delay bounds are $D_2^{\text{Boyer-et-al}} = 146.228\mu s$, $D_2^{\text{Soni-et-al}} = 112.172\mu s$, and $D_2^{\text{Boyer-et-al}} = 119.256\mu s$.

C. The Contradiction With the Bound of Soni et al.

We found a trajectory scenario such that $D_2^{\text{Soni-et-al}}$ is not a valid delay bound. Let us explain why the approach of Soni et al., presented in [7], gives an invalid delay bound. In [7], it is implicitly assumed that as the delay for a packet of flow 2 is upper bounded by $D_2^{\text{Boyer-et-al}}$ (the obtained delay bound using the strict service curve of Boyer et al. for flow 2), only packets of interfering flows arriving within a duration $D_2^{\text{Boyer-et-al}}$ will get a chance to delay a given packet of flow 2. However, in the trajectory scenario given in Section III-B, all packets of flow 3 (an interfering flow for flow 2) arriving within the time interval $[0, 75l]$ with the duration $75l \gg D_2^{\text{Boyer-et-al}} = 18.3l - 2.15l$ delay the packet of interest of flow 2.

IV. NEW DRR STRICT SERVICE CURVE

Our next result is new DRR strict service curves that do not take into account arrival curves of the interfering traffic. First, a non-convex strict service curve for DRR that we show that it is the largest one and it dominates the state-of-the-art rate-latency strict service curve for DRR by Boyer et al. We also give simpler, lower approximations of it. Specifically, we also find a convex strict service curve and two rate-latency strict service curves.

Theorem 1 (Non-Convex Strict Service Curve for DRR): Let $S$ be a server shared by $n$ flows that uses DRR, as explained in Section II-B, with quantum $Q_i$ for flow $i$. Recall that the server offers a strict service curve $\beta$ to the aggregate of the $n$ flows. For any flow $i$, $d_i^{\text{max}}$ is the maximum residual deficit (defined in Section II-C).

Then, for every $i$, $S$ offers to flow $i$ a strict service curve $\beta_i^{\text{NCDM}}$ given by $\beta_i^{\text{NCDM}}(t) = \gamma_i(t)$, with

$$
\gamma_i(x) = (\lambda_i \otimes \nu_{Q_i,Q_{\text{tot}}}) \left( [x - \psi_i(Q_i - d_i^{\text{max}})]^+ \right)
+ \min\left( [x - \sum_{j \neq i} (Q_j + d_j^{\text{max}})]^+, Q_i - d_i^{\text{max}} \right)
(4)
$$

$$
Q_{\text{tot}} = \sum_{j=1}^{n} Q_j
(5)
$$

$$
\psi_i(x) = x + \sum_{j \neq i} \phi_{i,j}(x)
(6)
$$

$$
\phi_{i,j}(x) = \left[ \frac{x + d_j^{\text{max}}}{Q_i} \right] Q_j + (Q_j + d_j^{\text{max}})
(7)
$$

Here, $\nu$, $\lambda$, is the unit rate function, $\lambda_i$ is the unit rate function and $\otimes$ is the min-plus convolution, all described in Fig. 2.

Furthermore, $\beta_i^{\text{NCDM}}$ is super-additive.

The proof is in Appendix A.1. See Fig. 5 for some illustrations of $\beta_i^{\text{NCDM}}$ (NCDM: non-convex, degraded mode). Our strict service curve captures the round-robin manner of DRR.
Assume that $b_i \in \mathcal{F}$ is a strict service curve for flow $i$ in any system that satisfies the specifications above. Then $b_i \leq \beta_{\text{NCDM}}^i$ where $\beta_{\text{NCDM}}^i$ is given in Theorem 1.

The proof is in Appendix A.2. The idea of the proof is as follows: For any value of the system parameters, for any $\tau > 0$, and for any flow $i$, we create a trajectory scenario of a system such that

$$\exists s \geq 0, (s, s + \tau] \text{ is backlogged for flow } i \text{ and } D_i(s + \tau) - D_i(s) = \beta_{\text{NCDM}}^i(\tau),$$

i.e., for the flow of interest $i$, we create a backlogged period of duration $\tau$ where the service received by flow $i$ is exactly equal to $\beta_{\text{NCDM}}^i(\tau)$. Then, it follows that every other strict service curve $b_i$ is upper bounded by $\beta_{\text{NCDM}}^i$. Note that assuming the aggregate strict service curve $\beta$ is Lipschitz-continuous does not appear to be a restriction as the rate at which data is served has a physical limit. We then provide closed-form for the network calculus delay bounds when the flow of interest $i$ is constrained by frequent types of arrival curves, as defined in Section II-A.

**Theorem 3 (Closed-form Delay Bounds Obtained with the Non-convex Strict Service Curve of DRR):** Make the same assumptions as in Theorem 1, yet with one difference: Assume that the aggregate strict service curve is a rate-latency function, i.e., $\beta = \beta_{\alpha,T}$. Also, assume that flow of interest $i$ has $\alpha_i \in \mathcal{F}$ as an arrival curve. Let $\psi_i$ be defined as in (6).

Then, the closed-form of the network calculus delay bound $h(\alpha_i, \beta_{\text{NCDM}}^i)$ is given as follows:

1) if $\alpha_i$ is a token-bucket arrival curve, i.e., $\alpha_i = \gamma_{r_i,b_i}$ with $r_i \leq Q_i / b_i$,

$$T + \max \left[ \frac{\psi_i(b_i)}{c}, \frac{\psi_i(\alpha_i(\tau_i))}{c} - \tau_i \right]$$

with $\tau_i = \frac{Q_i - (b_i + d_{\max})}{r_i}$; and for $t \leq T$, namely, $\beta_{\text{NCDM}}^i$ is derived from $\gamma_i$ by a re-scaling of the $x$ axis and a right-shift.

2) if $\alpha_i$ is a token-bucket arrival curve and we take into account the effect of grouping, i.e., $\alpha_i(t) = \min(c(t) + \max(d(t), \gamma_{r_i,b_i}(t)))$ with $r_i \leq Q_i / b_i$,

$$T + \max \left[ \frac{\psi_i(\alpha_i(\tau_i))}{c}, \frac{\psi_i(\alpha_i(\tau_i))}{c} - \tau_i \right]$$

with $\tau_i = \frac{Q_i - (b_i + d_{\max})}{r_i}$ and $\bar{\tau}_i = \frac{Q_i - (b_i + d_{\max})}{r_i}$ mod $Q_i$.

3) if $\alpha_i$ is a stair arrival curve, i.e., $\alpha_i(t) = a_i \left[ \frac{t}{b_i} \right]$ with $\frac{a_i}{b_i} \leq Q_i / b_i$,

$$T + \max \left[ \frac{\psi_i(a_i)}{c}, \frac{\psi_i(\alpha_i(\tau_i))}{c} - \tau_i \right]$$

with $\tau_i = \left[ \frac{Q_i - (a_i + d_{\max})}{b_i} \right] b_i$.

The proof is in Appendix A.3. The idea of the proof is that we write $h(\alpha_i, \beta_{\text{NCDM}}^i) = \sup_{t \geq 0} \{ \beta_{\text{NCDM}}^i(\alpha_i(t)) \}$, as in [11, Prop. 3.1.1], and we plug in the $\alpha_i$ and $\beta_{\text{NCDM}}^i$. Theorem 4 enables us to compute the exact delay bounds in a very simple closed-form, independent of the complicated expression of our non-convex strict service curve. However, we provide simplified lower bounds of the non-convex strict service curve for DRR, and in this closed-form expressions are important. As explained, the function $\phi_{i,j}(x)$, defined in (7), is the maximum interference that flow $j$ can create in any backlogged period of flow $i$, such that flow $i$ receives a service $x$. Using $\phi_{i,j}$ as it is, results in the strict service delay...
Corollary 1 (Rate-Latency Strict Service Curve for DRR): With the assumption in Theorem 1 and the definitions (12)-(13), \( S \) offers to every flow \( i \) strict service curves \( \gamma_i^{\maxRate}(\beta(t)) \) and \( \gamma_i^{\minLatency}(\beta(t)) \) with

\[
\begin{align*}
\gamma_i^{\maxRate} &= \beta_i^{\maxRate} \cdot T_{\max} \\
\gamma_i^{\minLatency} &= \beta_i^{\maxRate} \cdot T_{\min}
\end{align*}
\]

The right-hand sides in (14) and (15) are the rate-latency functions defined in Section II-A.

The above result is obtained by using Theorem 4 with \( \phi_i^{\maxRate} \) and \( \phi_i^{\minLatency} \); hence, \( \gamma_i \geq \phi_i^{\maxRate} \) and \( \gamma_i \geq \phi_i^{\minLatency} \). Also, observe that the strict service curve of Boyer et al., explained in Section II-C, is equal to \( \gamma_i^{\maxRate}(\beta(t)) \). It follows that \( \beta_i^{NCDM} \) dominates it; hence, obtained delay bound using \( \beta_i^{NCDM} \) are guaranteed to be less than or equal to those of Boyer et al.

A better upper bound on \( \phi_i \) can be obtained by taking its concave closure (i.e., the smallest concave upper bound) that is equal to the minimum of \( \phi_i^{\maxRate} \) and \( \phi_i^{\minLatency} \):

\[
\phi_i^{\conca}(x) = \min\left(\phi_i^{\maxRate}(x), \phi_i^{\minLatency}(x)\right)
\]

Corollary 2 (Convex Strict Service Curve for DRR): With the assumption in Theorem 1 and the definitions (14)-(15), \( S \) offers to every flow \( i \) a strict service curve \( \gamma_i^{\convex}(\beta(t)) \) with

\[
\gamma_i^{\convex}(x) = \max\left(\phi_i^{\maxRate}(x), \phi_i^{\minLatency}(x)\right)
\]

The above result is obtained by using Theorem 4 with \( \phi_i^{\conca} \). Also, it can be shown that it is the largest convex lower bound of \( \gamma_i \). When \( \beta \) is a rate-latency function, this provides a convex piecewise-linear function, which has all the good properties mentioned earlier.

V. NEW DRR STRICT SERVICE CURVES THAT ACCOUNT FOR ARRIVAL CURVES OF INTERFERING FLOWS

The next result provides a method to improve on any strict service curve by taking into account arrival curve constraints of interfering flows. It can thus be applied to the strict service curves presented in Section IV and to Bouillard’s strict service curves.

A. A Mapping to Refine Strict Service Curves for DRR by Accounting for Arrival Curves of Interfering Flows

Theorem 3 (Non-convex, Full Mapping): Let \( S \) be a server with the assumptions in Theorem 1. Also, assume that every flow \( i \) has an arrival curve \( a_i \in \mathcal{F} \) and a strict service curve \( \beta_i^{\text{old}} \in \mathcal{F} \), and let \( N_i = \{1, 2, \ldots, n\} \setminus \{i\} \), and for any \( J \subseteq N_i \), let \( J = N_i \setminus J \).
Then, for every flow $i$, a new strict service curve $\beta_i^{\text{new}} \in \mathcal{F}$ is given by

$$\beta_i^{\text{new}} = \max \left( \beta_i^{\text{old}}, \max_{j \in N_i} \gamma_i^j \circ \left[ \beta - \sum_{j \in J} (\alpha_j \otimes \beta_j^{\text{old}}) \right] \right)^+ \tag{20}$$

with

$$\gamma_i^j(x) = \left( \lambda_i \otimes \nu_{Q_i, Q_j^i} \right) \left[ x - \psi_i^j (Q_i - d_i^{\text{max}}) \right]^+ + \min \left( x - \sum_{j \in J} (Q_j + d_j^{\text{max}}) \right)^+, \quad Q_j^i = Q_i + \sum_{j \in J} Q_j \tag{21}$$

$$\psi_i^j(x) \overset{\text{def}}{=} x + \sum_{j \in J} \phi_{i,j}(x) \tag{22}$$

$$\beta_i^{\text{old}} = \max \left( \beta_i^{\text{old}}, \max_{j \in N_i} \gamma_i^j \circ \left[ \beta - \sum_{j \in J} (\alpha_j \otimes \beta_j^{\text{old}}) \right] \right)^+ \tag{23}$$

In (20), $[ \cdot ]^+$ is the non-decreasing and non-negative closure, defined in Section II-A, and $\circ$ is the composition of functions.

The proof is in Appendix A.5. The essence of Theorem 3 is as follows. Equation (20) gives new strict service curves $\beta_i^{\text{new}}$ for every flow $i$; they are derived from already available strict service curves $\beta_i^{\text{old}}$ and from arrival curves on the input flows $\alpha_j$; thus, it enables us to improve any collection of strict service curves that are already obtained.

The key point of Theorem 3 is as follows: As explained in Section IV, in $(s, t)$, a backlogged period of the flow of interest $i$, the service received by an interfering flow $j$ (i.e., $D_j(t) - D_j(s)$) is upper bounded as a function of the service received by flow $i$ (i.e., $\phi_{i,j} (D_i(t) - D_i(s))$, where $\phi_{i,j}$ is defined in (7)). Also, the service received by an interfering flow $j$ is upper bounded by the output arrival curve of flow $j$ (i.e., $(\alpha_j \otimes \beta_j^{\text{old}}) (t - s)$); combining both upper bounds results in our Non-convex, Full mapping.

The computation of service curves in Theorem 3 and of the resulting delay bounds can be restricted to a finite horizon. Indeed, all computations in Theorem 3 are causal except for the min-plus deconvolution $\alpha_j \otimes \beta_j^{\text{old}}$. But, as explained in Section II-A, such a computation and the computation of delay bounds can be limited to $t \in [0; t^*]$ for any positive $t^*$ such that $\alpha_j (t^*) \leq \beta_j^{\text{old}} (t^*)$ for every $m \geq 1$ and $j = 1 : n$. To find such a $t^*$, we can use any lower bound on $\beta_j^{\text{old}}$.

We then compute $t_j^* = \inf_{s \geq 0} \{ \alpha_j(s) \leq \beta_j^{\text{old}}(s) \}$ and take, as sufficient horizon, $t^* = \max_j t_j^*$. The computations in Theorem 3 can then be limited to this horizon or any upper bound on it. The computations can be performed with a tool such as RealTime-at-Work (RTaW) [16], which uses an exact representation of functions with finite horizon, by means of rational numbers with exact arithmetic.

An iterative scheme can be obtained as follows: Theorem 3 can be iteratively applied, starting from any available strict service curves for all flows, and for every flow an increasing sequence of strict service curves is obtained; specifically, let $\beta_i^0$ be an initial strict service curve for every flow $i$; then, for every integer $m \geq 1$ and every flow $i$, define $\beta_i^m$ by replacing $\beta_i^j$ with $\beta_i^{m-1}$ in (20):

$$\beta_i^m = \max \left( \beta_i^{m-1}, \max_{j \in N_i} \gamma_i^j \circ \left[ \beta - \sum_{j \in J} (\alpha_j \otimes \beta_j^{m-1}) \right] \right)^+ \tag{24}$$

It follows that $\beta_i^m$ is a strict service curve for flow $i$ and $\beta_i^0 \leq \beta_i^1 \leq \beta_i^2 \leq \ldots$.

We are guaranteed simple convergence for the strict service curves of all flows when iteratively applying Theorem 3, starting from any available strict service curves for all flows. This is because, first, as explained above, computations of such strict service curves can be limited to a sufficient finite horizon; second, by iteratively applying Theorem 3, we obtain an increasing sequence of strict service curves for all flows, and every strict service curve is upper bounded by $\beta$, the aggregate strict service curve. In all cases that we tested, the iterative scheme became stationary in such a finite horizon. Note that the computed strict service curves at each iteration are valid, hence can be used to derive valid delay bounds; this means the iterative scheme can be stopped at any iteration. For example, the iterative scheme can be stopped when the delay bounds of all flows decrease insignificantly.

This iterative scheme can be initialized by strict service curves that do not make any assumptions on interfering traffic, as obtained in Section IV; specifically, recall that $\beta_i^{\text{old}}$ is defined in Theorem 1, then, for every flow $i$, let $\beta_i^0 = \beta_i^{\text{old}}$, and for every integer $m \geq 1$, $\beta_i^m$ is obtained as in (32) (see Fig. 7 (left)).

Alternatively, we can first compute Bouillard’s strict service curve $\beta_j^{\text{Bouillard}}$ for every flow $j$, as explained in Section II-E. Observe that $\beta_j^{\text{Bouillard}}$ does not usually dominate the non convex service curve $\beta_j^{\text{NCDM}}$ obtained in Theorem 1 (see Figure 8). Therefore, since the maximum of two strict service curves is a strict service curve, we can take the maximum of both. Specifically, define $\beta_i^m = \max (\beta_i^{\text{Bouillard}}, \beta_i^{\text{NCDM}})$, then, for every integer $m \geq 1$ and every flow $i$, define $\beta_i^m$ as in (23) (see Fig. 7 (right)).

In practice, in all cases that we tested, when initializing the method with either choice, we always converge to the same strict service curve for every flow (Fig. 7). Note that when initializing the method with strict service curves that are true for the degraded operational mode (i.e., ones that do not take into account the arrival curves of the interfering flows), the iterative scheme will always converge to the same strict service curves. This is because in the first iteration, we take into account our strict service curve, found in Theorem 1, which is the best possible one, shown in Theorem 2; hence, whatever the initial strict service curves will be dominated by ours, and the scheme iterates independent from the initial strict service curve.

Observe that the computation to compute strict service curve of Theorem 3, $\beta_i^{\text{new}}$ in (20), requires $2^{n-1}$ computations of $\gamma_j \circ \left[ \beta - \sum_{j \in J} (\alpha_j \otimes \beta_j^{\text{old}}) \right]^+$ for each $J$ (where $n$ is the total number of the input flows of the DRR subsystem). In some cases (class based networks) $n$ is small and this is not an issue; in other scenarios (per-flow queuing), this may cause excessive complexity. To address this, we find lower bounds on the strict service curve of Theorem 3 where only one computation at each step $m$ is needed; this is less costly when $n$ is large.

Corollary 3 (Non-convex, Simple Mapping): Make the same assumption as in Theorem 3. Then, for every flow $i$, a new strict service curve $\beta_i^{\text{new}} \in \mathcal{F}$ is given by

$$\beta_i^{\text{new}} = \max \left( \beta_i^{\text{old}}, \gamma_i \circ (\beta_i^{\text{old}}) \right) \tag{25}$$
In the examples that we tested, we observed that the iterative scheme obtained with Theorem 3, our non-convex, full mapping, converges to the same results as the iterative scheme obtained with Corollary 3, our non-convex, simple mapping; however, it requires more iterations (see Fig. 7 and Fig. 8).

**B. Convex Versions of the Mapping**

Computation of the strict service curves of Theorem 3 and Corollary 3 can be costly. We first explain some sources of complexity and how to address them. We then propose convex versions, for both the non-convex, full mapping in Theorem 3 and the non-convex, simple mapping in Corollary 3.

1) **Convex Versions of Theorem 3:** One source of complexity lies in the initial strict service curves \( \beta_0 \). For every flow \( i \), \( \beta_0 \) can be replaced by its simpler lower bounds. As presented in Section IV, \( \beta_0 \) can be replaced by its convex closure \( \gamma^\text{convex} \). For every flow \( i \), \( \gamma^\text{convex} \) can be replaced by its rate-latency functions \( \gamma^\text{minLatency} \) and \( \gamma^\text{maxRate} \).

Another source of complexity is function \( \beta^\text{new} \), as defined in (21), is non-convex and results in strict service curves that are also non-convex (Fig. 7). If there is interest in simpler expressions of Theorem 3, any lower bounding function on \( \beta^\text{new} \) results in a lower bound of \( \beta^\text{new} \), which is a valid, though less good, strict service curve for DRR.

**Corollary 4 (Convex, Full Mapping):** Make the same assumptions as in Theorem 3. Also, for a flow \( i \), let \( \tilde{\gamma}_i^J \in \mathcal{F} \) such that \( \tilde{\gamma}_i^J \leq \gamma_i^J \).

Let \( \hat{\gamma}_i^J \) be the result of Theorem 3, in (20), by replacing functions \( \gamma_i^J \) with \( \tilde{\gamma}_i^J \).

Then, \( S \) offers to flow \( i \) a strict service curve \( \hat{\gamma}_i^J \) and \( \tilde{\gamma}_i^J \leq \hat{\gamma}_i^J \).

As of today, in tools such as RTaW working with functions that are linear and convex is simpler and tractable. Hence, we apply Corollary 4 with \( \tilde{\gamma}_i^J = \tilde{\gamma}_i^\text{convex} \). Hence, we apply Corollary 4 with \( \tilde{\gamma}_i^J = \tilde{\gamma}_i^\text{convex} = \max \left( \gamma_i^\text{maxRate}^J, \gamma_i^\text{minLatency}^J \right) \) (convex closure of function \( \gamma_i^J \)) and

\[
\gamma_i^\text{maxRate}^J = \beta^\text{maxRate}_i \quad \gamma_i^\text{minLatency}^J = \beta^\text{minLatency}_i
\]
As illustrated in Fig. 9, obtained strict service curves are convex, thus computing the min-plus deconvolution with such strict service curves is much simpler than with those in Fig. 7. Also, the composition is simpler, as for \( f \in \mathcal{F} \), a function \( \gamma^{\text{conv}}(\beta(t)) \) is equal to max \( \left( R_i^{\max} \cdot t - T_i^{\max} \right)^+, T_i^{\min} \cdot \left( t - T_i^{\min} \right)^+ \), which includes only multiplication, addition, and maximum operations.

2) Convex Versions of Corollary 3: Again here a source of complexity lies in the initial strict service curves \( \beta^0_i \). For every flow \( i \), \( \beta^0_i \) can be replaced by its simpler lower bounds. As presented in Section IV, \( \beta^0_i \) can be replaced by its convex closure \( \gamma^{\text{conv}}(\beta(t)) \), or rate-latency functions \( \gamma_{\text{minLatency}}(\beta(t)) \) and \( \gamma_{\text{maxRate}}(\beta(t)) \).

Also, another source of complexity is function \( \phi_{i,j} \) (and the resulting function \( \gamma_i \)). Function \( \phi_{i,j} \), as defined in (7), is non-concave and non-linear (because it uses floor operations). This might create discontinuities that can make the computation hard, see Fig. 8. To address this problem, we derive the following convex version of Corollary 3.

**Theorem 4 (Convex, Simple Mapping):** Make the same assumptions as in Corollary 3. Also, for a flow \( i \), let \( \phi_{i,j} \) and \( \gamma_i \) be defined as in Theorem 4.

Let \( \beta^0_i \) be the result of Corollary 3 by replacing functions \( \phi_{i,j} \) and \( \gamma_i \) with \( \phi_{i,j}' \) and \( \gamma_i' \), respectively.

Then, \( S \) offers to every flow \( i \) a strict service curve \( \beta^0_i \).

The proof is not given in detail, as it is similar to the proof of Corollary 3 after replacing functions \( \phi_{i,j} \) and \( \gamma_i \) with \( \phi_{i,j}' \) and \( \gamma_i' \), respectively.

We apply Theorem 4 as follows: Apply Theorem 4 by replacing \( \phi_{i,j} \) and \( \gamma_i \) with \( \phi_{i,j}' \) and \( \gamma_i' \) defined in (12) and (14); also, apply Theorem 4 by replacing \( \phi_{i,j} \) and \( \gamma_i \) with \( \phi_{i,j,\text{latency}} \) and \( \gamma_i' \) defined in (13) and (15); then, we take the maximum of the two strict service curves obtained in each case.

This can be iteratively applied: In both cases, let the initial strict service curves \( \beta^0_i \) be defined as in Corollary 2. Specifically, the sequence of obtained strict service curves are
thus defined by either $\bar{\beta}^{\text{convex},0} = \gamma^{\text{convex}} \circ \beta = \bar{\beta}^{\text{convex}}$ or $\bar{\beta}^{\text{convex},0} = \max(\bar{\beta}^{\text{convex},0}^{\text{Bouillard}}, \bar{\beta}^{\text{convex}})$ and for $m \geq 1$, $\bar{\beta}^{\text{convex},m} = \max(\bar{\beta}^{m'}, \bar{\beta}^{m''})$ with

$$
\bar{\beta}^{m'} = \gamma^{\text{minLatency}} \circ \left( \bar{\beta}^{\text{convex},m-1} \right),
$$

$$
\bar{\beta}^{m''} = \gamma^{\text{maxRate}} \circ \left( \bar{\beta}^{\text{convex},m-1} \right).
$$

(33)

Also, $\delta^{\text{minLatency},m-1}$ and $\delta^{\text{maxRate},m-1}$ are equal to

$$
\sum_{j \neq i} \left[ \phi_{i,j}^{\text{minLatency}} \circ \bar{\beta}^{\text{convex},m-1} - \alpha_{i,j} \circ \bar{\beta}^{\text{convex},m-1} \right]^+ 
$$

and

$$
\sum_{j \neq i} \left[ \phi_{i,j}^{\text{maxRate}} \circ \bar{\beta}^{\text{convex},m-1} - \alpha_{i,j} \circ \bar{\beta}^{\text{convex},m-1} \right]^+.
$$

(34)

respectively (see Fig. 10).

Let us explain why computing the above strict service curves is simpler. The first reason is in computing the composition of $\bar{\beta}_{i,j}^{\text{maxRate}}$ (resp. $\bar{\beta}_{i,j}^{\text{minLatency}}$) with another function. Observe that for a function $f \in \mathcal{F}$, $\bar{\beta}_{i,j}^{\text{maxRate}}(f(t))$ (resp. $\bar{\beta}_{i,j}^{\text{minLatency}}(f(t))$) is equal to $\frac{Q_i}{Q_j} f(t) + \bar{\beta}_{i,j}^{\text{maxRate}}(0)$ (resp. $\frac{Q_i}{Q_j} f(t) + \bar{\beta}_{i,j}^{\text{minLatency}}(0)$), which includes only multiplication, addition, and minimum operations. The second reason is in computing the min-plus deconvolution; min-plus convolution and deconvolution of piecewise linear convex can be computed in automatic tools, such as RTaW, very efficiently [13, Section 4.2], and as illustrated in Fig. 10, obtained strict service curves are convex, thus computing the min-plus deconvolution with such strict service curves is much simpler than with those in Fig. 8. The last reason is in computing the composition of $\gamma^{\text{maxRate}}$, $\gamma^{\text{minLatency}}$, $\gamma^{\text{convex}}$ with another function. Observe that for a function $f \in \mathcal{F}$, $\gamma^{\text{maxRate}}(f(t))$ (resp. $\gamma^{\text{minLatency}}(f(t))$) is equal to $R_i^{\text{max}} [f(t) - T^{\text{min}}]^+$ (resp. $R_i^{\text{max}} \left[ f(t) - T^{\text{min}} \right]^+$), which again includes only multiplication, addition, and maximum operations.

Alternatively, one can apply Theorem 4 by replacing $\phi_{i,j}$ and $\gamma_i$ with $\phi_{i,j}^{\text{convex}}$ and $\gamma_i^{\text{convex}}$ defined in (18) and (19);

however, in this case, there is no guarantee that this version conserves convexity and we do not consider it further.

In the examples that we tested, we observed that the iterative scheme obtained with Corollary 4, our convex, full mapping, converges to the same results as the iterative scheme obtained with Theorem 4, our convex, simple mapping; however, it requires more iterations (see Fig. 9 and Fig. 10).

VI. NUMERICAL EVALUATION

In this section, we compare the obtained delay bounds by using our new strict service curves for DRR, presented in Sections IV and V, to those of Boyer et al., Bouillard, and Soni et al. We use all network configurations that were presented by Bouillard in [8] and Soni et al. in [7], specifically, one single server, two illustration networks, and an industrial-sized one. For the illustration networks, we use the exact same configuration of flows and switches that Soni et al. use.
token bucket arrival curves with bursts games, Video conference, and 4k videos constrained with four classes of traffic: Electric protection, Virtual reality server that Bouillard uses in [8]. Consider a DRR subsystem A. Single Server

we use the same network but randomly choose the missing configuration than what is already given in [7]. Consequently, we use the same network but randomly choose the missing information (explained in detail in Section VI-C).

A. Single Server

We use the exact same configuration of flows and the server that Bouillard uses in [8]. Consider a DRR subsystem with four classes of traffic: Electric protection, Virtual reality games, Video conference, and 4k videos constrained with token bucket arrival curves with bursts \( b = \{42.56, 2160, 3240, 7200\} \) kb and rates \( r = \{8.521, 180, 162, 180\} \) Mb/s, respectively; also, the packet sizes are \( l_{\text{max}} = \{3.04, 12, 12\} \) kb. The server is a constant-rate server with rate equal to \( c = 5\text{Gb/s}, \) i.e., \( \beta(t) = ct \). All classes have the same quantum equal to \( 16000 \) bits.

The delay bounds obtained with different methods are given in Table I. Our delay bounds always improve on those of Boyer et al. and Bouillard; when the delay is very small (electric protection) our non-convex service curves bring a considerable improvement. As discussed in Section V, the results are the same with the non-convex, full mapping (Theorem 3) and the non-convex simple mapping (Corollary 4). The results of the convex full and simple mappings (Corollary 3 and Theorem 4) are also identical, but less good than the former. Also, the results are the same for all choices of initial strict service curves. Finally, we used the same trajectory scenario, given in Appendix A.2, to provide a lower bound on the worst-case delay; we observe that our delay bounds are tight (for 4k videos almost tight), i.e., exactly equal to the worst-case delays, in either degraded (i.e., when some traffic classes misbehaves) and non-degraded (i.e., when arrival curve constraints can be assumed for interfering traffic) operational mode in this single server example. As we showed that the delay bounds of Soni et al. are incorrect we do not compute them for this example, but for the sake of comparison we will compute them for their case-studies.

B. Illustration Networks

Example 1 and 2 are illustrated in Fig. 12. We use the exact same network with the exact same configuration for flows and switches as used by Soni et al. in [7]. Examples 1 and 2 differ only by the configuration of the switch \( S_4 \). Flows \( \{v_1 \ldots v_5\} \), \( \{v_6 \ldots v_{12}\} \), and \( \{v_{13} \ldots v_{20}\} \) are assigned to class \( C_1, C_2, \) and \( C_3 \), respectively. There is one DRR scheduler at every switch output port; what we called “flow” earlier in the paper corresponds here to a class hence \( n = 3 \). Inside a class, arbitration is FIFO (all packets of all flows of a given class are in the same FIFO queue). Also, as in [7], we assume that queuing is on output ports only. All classes have the same quantum equal to 199 bytes. The rate of the links are equal to \( c = 100 \text{Mb/s}, \) and every switch \( S_i \) has a switching latency equal to \( 16\mu s \). Every flow \( v_i \) has a maximum packet size \( l_{\text{max}} \) and minimum packet arrival \( T_i \). Hence, flow \( v_i \) is constrained by a token-bucket arrival curve with rate equal to \( \frac{c}{T_i} \) and burst equal to \( l_{\text{max}} \); also, it is constrained by a stair arrival curve given by \( l_{\text{max}} \frac{c}{T_i} \).

For the sake of comparison, as Soni et al. do not consider grouping and offsets (explained in Section II-D) in these two examples, we also do not consider them. This means that the arrival curve we use for bounding the input of a class at a switch is simply equal to the sum of arrival curves expressed for every member flow. Arrival curves are propagated using the delay bounds computed at the upstream nodes. We illustrate the reported values in [7] for the delay bounds of Soni et al. For the other results, we use the RTaW online tool (Fig. 13). As explained in Section II-A, RTaW provides all the necessary operations to implement our new strict service curves for DRR. First, observe that delay bounds obtained with our new strict service curves for DRR, with no knowledge on the interfering traffic, are always better than those of Boyer et al. Second, delay bounds obtained with our new strict service curve for DRR that accounts for arrival curve of interfering flows are always better than the (incorrect) ones of Soni et al. and are considerably better than Bouillard’s. The obtained delay bounds using Theorem 3, our non-convex full mapping, are better than or equal to those of obtained using Corollary 4, its convex version; also, they are equal to those of Corollary 3, our non-convex simple mapping. Note that the results do not differ whatever the initial strict service curves are. When using token-bucket arrival curves, the run-times (on the RTaW online tool) of Theorem 3 and Corollary 3 are in the order of 3 minutes; for their convex versions, Corollary 4 and Theorem 4, they are in the order of 30 seconds; when using stair arrival curves, the run-times (on the RTaW online tool) of Theorem 3 and Corollary 3 are in the order of 5 minutes; for their convex versions, Corollary 4 and Theorem 4, they are in the order of 1 minute and 45 seconds, respectively.

| Class                  | Boyer et al. | Thm. 1 | Bouillard | Thm. 5 | Cor. 4 | Simulation (non-degraded) | Simulation (degraded) |
|------------------------|--------------|--------|-----------|--------|--------|---------------------------|------------------------|
| Electric protection (µs) | 52           | 44.51  | 52        | 44.51  | 52     | 44.51                     | 44.51                  |
| Virtual reality games (ms) | 1.74         | 1.74   | 1.33      | 1.32   | 1.32   | 1.32                      | 1.74                   |
| Video conference (ms)   | 2.61         | 2.61   | 1.82      | 1.81   | 1.81   | 1.81                      | 2.61                   |
| 4k videos (ms)          | 5.78         | 5.77   | 2.74      | 2.72   | 2.72   | 2.71                      | 5.77                   |

For the industrial-sized network, Soni kindly replied to our e-mail request by saying that, for confidentiality reasons, they do not have the rights to provide more details about the network configuration than what is already given in [7]. Consequently, we use the same network but randomly choose the missing information (explained in detail in Section VI-C).
Fig. 13. Delay bounds of flow $v_1, v_2, \ldots, v_{20}$ in Example 1 and Example 2 of Fig. 12. In each example, we follow [7] and assume once that flows are constrained by token-bucket arrival curves, and once that flows are constrained by stair arrival curves. The delay bounds of Soni et al. are taken from [7], and other results are computed with the RTaW online tool. First, delay bounds obtained with our new strict service curves for DRR, with no knowledge on the interfering traffic, are always better than those of Boyer et al. Second, delay bounds obtained with our new strict service curve for DRR that accounts for arrival curve of interfering flows are always better than those of Soni et al. and are considerably better than delay bounds of Bouillard. The obtained delay bound obtained with Theorem 3 and Corollary 4, our non-convex and convex full mapping, are equal to those of obtained with Corollary 3 and Theorem 4, our non-convex and convex simple mapping. In each plot, flows are ordered by values of Boyer’s bound. The state-of-the-arts, i.e., delay bounds of Boyer et al., Soni et al., and Bouillard are plotted with dashed lines.

Fig. 14. Industrial-sized network topology. The figure is taken from [23].

C. Industrial-Sized Network

We use the network of Fig. 14; it corresponds to a test configuration provided by Airbus in [21]. The industrial-sized case study that Soni et al. use in [7] is based on this network in [23]. We combine the available information in both papers to understand this network. It includes 96 end-systems, 8 switches, 984 flows, and 6412 possible paths. The rate of the links are equal to $c = 100$ Mb/s, and every switch $S_i$ has a switching latency equal to $16\mu s$. We find that each switch has 6 input and 6 output end-systems. Three classes of flows are considered: critical flows, multimedia flows, and best-effort flows. There is one DRR scheduler at every switch output port with $n = 3$ classes. At every DRR scheduler, the quanta are 3070 bytes for the critical class, 1535 bytes for the multimedia class, and 1535 bytes for the best-effort class. 128 multicast flows, with 834 destinations, are critical; they have a maximum packet-size equal to 150 bytes and their minimum packet arrival time is between 4 and 128 ms. 500 multicast flows, with 3845 destinations, are multimedia and their class has a quantum equal to 1535 bytes; they have a maximum packet-size equal to 500 bytes; and their minimum packet arrival time is between $2$ and $128$ ms. 266 multicast flows, with 1733 destinations, are best-effort; they have a maximum packet-size equal to 500 bytes; and their minimum packet arrival time is between $2$ and $128$ ms. For every flow, the path from the source to a destination can traverse at most 4 switches. Specifically, 1797, 2787, 1537, and 291 source-destination paths have 1, 2, 3, and 4 hops, respectively. We choose the paths randomly and satisfy all these constraints.
Due to the limited expressiveness of the language used by the RTaW online tool, we could not implement the industrial-size network there. Therefore, we used MATLAB, which has the required expressiveness. The obtained delay bounds are quasi identical for the full and simple versions of the mappings, therefore we illustrate results only for Theorem 3 (non-convex full mapping) and Corollary 4 (convex full mapping).

Note that the results are identical for both mentioned choices of initial strict service curves. We also computed the delay bounds obtained with the strict service curve of Boyer et al., with Bouillard’s strict service curve and with the correction term of Soni et al. In all cases, and as in [7], the arrival curve used for bounding the input of a class at a switch incorporates the effects of delay bounds computed upstream, as well as grouping (line shaping) and offset (Section II-D); furthermore, the offsets are such that they create maximum separation, as with [7]. We find that our bounds significantly improve upon the existing bounds, even the incorrect ones (Fig. 15). Moreover, we always improve on Bouillard’s delay bounds. Also, delay bounds obtained using Theorem 3 are considerably improved compared to its convex version for flows with low delay bounds.

Remark on run-times: For the industrial sized described above, run-times (on a 2.6 GHz 6-Core Intel Core i7 computer) of Theorem 3 and its convex version are 96 and 72 minutes, respectively; however, run-times of Corollary 3 and its convex version are higher and are 130 and 103 minutes, respectively. This is because the number of classes is small, i.e., 3 classes. To increase this, we divided at uniformly random flows of each class to three new classes, which results in 9 classes in total. By doing so, run-times of Theorem 3 and its convex version are 275 and 220 minutes, respectively; however, run-times of Corollary 3 and its convex version are lower and are 162 and 130 minutes, respectively. This supports the fact that computation of strict service curves of Corollary 3 is faster than those of Theorem 3, when the number of flows is large.

VII. Conclusion

The method of the pseudo-inverse enables us to perform a detailed analysis of DRR and to obtain strict service curves that significantly improve the previous results. Our results use the network calculus approach and are mathematically proven, unlike some previous delay bounds that we have proved to be incorrect. Our method assumes that the aggregate service provided to the DRR subsystem is modelled with a strict service curve. Therefore it can be recursively applied to hierarchical DRR schedulers as found, for instance, with class-based queuing. Our strict service curves in the degraded operational mode (i.e., when there is no assumption on the arrival curves of interfering flows) are the best possible ones; however, for the non-degraded operational mode (i.e., when some arrival curves can be assumed for interfering flows), our strict service curves are the best possible ones so far, but the jury is still out on them.

References

[1] M. Shreedhar and G. Varghese, “Efficient fair queuing using deficit round-Robin,” IEEE/ACM Trans. Netw., vol. 4, no. 3, pp. 375–385, Jun. 1996.
[2] L. Lenzini, E. Mingozzi, and G. Stea, “Aliquem: A novel DRR implementation to achieve better latency and fairness at O(1) complexity,” in Proc. IEEE 10th IEEE Int. Workshop Quality Service, May 2002, pp. 77–86.
[3] S. S. Kanhere and H. Sethu, “On the latency bound of deficit round Robin,” in Proc. 11th Int. Conf. Comput. Commun. Netw., Oct. 2002, pp. 548–553.
[4] D. Stiliadis, “Traffic scheduling in packet-switched networks: Analysis, design, and implementation,” Ph.D. dissertation, Univ. California, SantaCruz, Santa Cruz, CA, USA, 1996.
[5] L. Lenzini, E. Mingozzi, and G. Stea, “Full exploitation of the deficit round Robin capabilities by efficient implementation and parameter tuning,” Dipartimento di Ingegneria della Informazione, Univ. Pisa, Pisa, Italy, Tech. Rep., 2003. [Online]. Available: http://docenti.ing.unipi.it/~al080368/papers/Tecnical%20Report%20Oct%202003.pdf
[6] M. Boyer, G. Stea, and W. M. Sofack, “Deficit round Robin with network calculus,” in Proc. 6th Int. Conf. Perform. Eval. Methodol. Tools, 2012, pp. 138–147.
[7] A. Soni, X. Li, J.-L. Scharbarg, and C. Fraboul, “Optimizing network calculus for switched Ethernet network with deficit round Robin,” in Proc. IEEE Real-Time Syst. Symp. (RTSS), Dec. 2018, pp. 300–311.
[8] A. Bouillard, “Individual service curves for bandwidth-sharing policies using network calculus,” IEEE Netw. Lett., vol. 3, no. 2, pp. 80–83, Jun. 2021.
[9] S. M. Tabatabae, J.-Y. Le Boudec, and M. Boyer, “Interleaved weighted round-Robin: A network calculus analysis,” in Proc. 32nd Int. Teletraffic Congr. (ITC 32), Sep. 2020, pp. 64–72.
[10] S. M. Tabatabae and J.-Y.-L. Boudec, “Deficit round-Robin: A second network calculus analysis,” in Proc. IEEE 27th Real-Time Embedded Technol. Appl. Symp. (RTAS), May 2021, p. 13. [Online]. Available: http://infoscience.epfl.ch/record/285728
[11] J.-Y. L. Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, vol. 2050. Cham, Switzerland: Springer, 2001.

[12] C. S. Chang, *Performance Guarantees in Communication Networks*. New York, NY, USA: Springer, 2000.

[13] A. Bouillard, M. Boyer, and E. L. Corronc, *Deterministic Network Calculus: From Theory to Practical Implementation*. Hoboken, NJ, USA: Wiley, 2018.

[14] K. Lampka, S. Bondorf, and J. Schmitt, “Achieving efficiency without sacrificing model accuracy: Network calculus on compact domains,” in Proc. IEEE 24th Int. Symp. Model., Anal. Simul. Comput. Telecommun. Syst. (MASCOTS), Sep. 2016, pp. 313–318.

[15] J. Liebeherr, “Duality of the max-plus and min-plus network calculus,” Found. Trends Netw., vol. 11, nos. 3–4, pp. 139–282, 2017.

[16] RealTime-at-Work Online Min-Plus Interpreter for Network Calculus. Accessed: Aug. 1, 2021. [Online]. Available: https://www.realtimework.com/minplus-playground

[17] A. Charny and J.-Y. Le Boudec, “Delay bounds in a network with aggregate scheduling,” in *Quality of Future Internet Services*. Cham, Switzerland: Springer, 2000, pp. 1–13.

[18] D. B. Chokshi and P. Bhaduri, “Modeling fixed priority non-preemptive scheduling with real-time calculus,” in Proc. 14th IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl., Aug. 2008, pp. 387–392.

[19] L. Thiele, S. Chakraborty, and M. Naeele, “Real-time calculus for scheduling hard real-time systems,” in Proc. 21st IEEE Int. Symp. Circuits Syst. Emerg. Technol., vol. 4, May 2000, pp. 101–104.

[20] A. Mifdaoui and T. Leydier, “Beyond the accuracy-complexity trade-offs of Compositional Analyses using network calculus for complex networks,” in Proc. 10th Int. Workshop Compositional Theory Technol. Real-Time Embedded Syst. (Co-Located With RTSS), Paris, France, Dec. 2017, pp. 1–8. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01690096

[21] J. Grieu. (Sep. 2004). *Analyse et évaluation de Techniques de Commutation Ethernet Pour l’Interconnexion Des Systèmes Avioniques*. [Online]. Available: https://oatao.univ-toulouse.fr/7385/

[22] A. Bouillard, “Trade-off between accuracy and tractability of Network Calculus in FIFO networks,” Perform. Eval., vol. 153, p. 102250, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166531621000675, doi:10.1016/j.peva.2021.102250.

[23] H. Charara, J.-L. Scharbarg, J. Ermont, and C. Fraboul, “Methods for bounding end-to-end delays on an AFDX network,” in Proc. 18th Euromicro Conf. Real-Time Syst. (ECRTS), Jul. 2006, p. 10.

[24] S. M. Tabatabae, J.-Y. L. Boudec, and M. Boyer, “Interleaved weighted round-Robin: A network calculus analysis,” 2020, arXiv:2003.08372.

[25] M. Boyer and P. Roux. (May 2016). A Common Framework Embedding Network Calculus and Event Stream Theory. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01311502