Optimal Latency Guarantee for Multiple Concurrent Packets with New IP

Lijun Dong, Richard Li
Futurewei Technologies Inc.
2220 Central Expressway
Santa Clara, CA, U.S.A
Email: {lijun.dong, richard.li}@futurewei.com

***Abstract*— Precise end-to-end latency guarantee is a network service that is required by many emerging and future applications. However, today's Internet built on the best effort principle cannot provide such service, despite of the existing Quality of Service (QoS) mechanisms. Enabled by the New IP framework, the deadline for the packets could be revealed to the network nodes and leveraged to calculate the residual latency budget and average per-hop latency constraint. Correspondingly, the packet forwarding order (i.e., the placement positions of the packets) in the outgoing queue could be deliberately manipulated to satisfy the deadline constraints for as many packets as possible, while achieving the minimum average stay time in a network node. Algorithms based on backtracking, branch and bound are proposed to address the optimal scheduling problem.**

Keywords*— *in-time guarantee; multiple packets; New IP; best effort; contract; metadata; high precision communication; QoS; precise latency; backtracking; branch and bound.

I. INTRODUCTION

Today's Internet is based on the Best Effort (BE) principle. BE is a network service that attempts to deliver packets to their intended destinations, but does not provide any guarantee of the Quality of Service (QoS), e.g., whether the packet gets dropped, or the packet reaches the destination within certain deadline. The use of BE was adopted because rather than guaranteed delivery, BE can be more efficient for some earlier services, and for the network as a whole. For example, in real-time audio or video transfers, a small percentage of packets getting lost is tolerable (i.e., does not affect the sound or video conspicuously), and recovering the lost and corrupted packets results in immoderate overhead that reduces network performance. However, for the emerging Internet applications, such as remote surgery, cloud-based autonomous driving, industrial Internet, each piece of information must be delivered precisely, referred to as High Precision Communication (HPC) [1][2]. In-time guarantee regarding the latency performance [3] is one of the most important yet barely explored territory. It refers to a network service that ensures the delivery of a packet, a group of packets, or all packets in a flow within bounded time frame. Remote surgery application requires that all messages between the master console and remote robots are delivered through the networks within the specified deadlines. If all messages are specified with the same deadline, then the latency guarantee is ensured at the flow level. If each message is specified with an independent deadline, then the latency guarantee is ensured at the packet level.

Although IntServ [4] and DiffServ [5] were proposed to improve upon BE, neither of them is suitable to the above emerging applications. The IntServ QoS model works in small networks, which is hard to be implemented in a large scale or be used in the global Internet. The DiffServ QoS model differentiates the service priorities at class level, which is not satisfactory to the in-time guarantee requirement at the flow level, not to mention the packet level. In [6][7], the authors proposed a new class called Latency Guarantee Service (LGS) on top of already defined classes in DiffServ. The flows that belong to this LGS class will have the highest priority to be transmitted after being admitted. The maximum latency that may be incurred at each intermediate hop is calculated to ensure that the total end-to-end latency of an admitted LGS flow will not exceed its deadline. However, the proposal still only works at class level of granularity, and the end-to-end latency estimation is very raw at its upper bound, thus the network resource may not be efficiently used.

Some deadline-aware transport schemes have been proposed to in Data Centers, such as Deadline-Aware Data center TCP (D2TCP) [7], Deadline Driven Delivery (D3) [9], and Preemptive Distributed Quick (PDQ) [10], which perform flow scheduling to complete serving the most significant number of flows before their deadlines. D3 is a deadline-aware transport scheme, in which senders calculate the requesting rate for flows before the actual flow transmission starts, and the on-path switches towards the destination take the role in helping make decisions on the sending rate for each active flow in a First-Come-First-Served (FCFS) manner. PDQ uses two policies, Early Deadline First (EDF) and Shortest Job First (SJF), where the latter is used to break ties for scheduling. PDQ may preempt a flow that is currently being served (i.e., the active flow) if the deadline of a new arriving flow is tighter than that of the currently active flow. In a more recent work [11], the authors proposed the Preemptive Efficient Queuing (PEQ), which takes both the deadlines and sizes of the flows into account for efficient scheduling of flows in a data-center network. However, even though all those works considered deadlines, they are at the flow level and the major goal is still to optimize the flow throughput. They cannot guarantee the transmission latency of a particular packet or a group of packets in a flow to be within the bounded time frame. On the other hand, with the existing scheduling polices (e.g., FCFS, EDF, SJF), some of the concurrent flows can fail when the deadline expires.

In this paper, we propose to leverage the New Internet Protocol (New IP) framework, such that each intermediate router on the forwarding path is able to process the packet at per-hop basis and schedule all concurrent latency-sensitive packets intelligently in order to achieve the shortest total stay time for those packets in the router. We need to point out that New IP serves as one embodiment of the proposal. We do not exclude other tentative implementation possibilities, such as IPv6

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-964-5

(Internet Protocol Version 6) extension headers, or IPv4 (Internet Protocol Version 4) options. The rest of the paper is arranged as follows: Section II introduces the New IP framework; Section III describes the proposed in-time guarantee mechanisms and algorithms; Section IV gives performance comparisons; Section V concludes the paper.

II. NEW INTERNET PROTOCOL (NEW IP)

New Internet Protocol (New IP) [12] [13] has been proposed to address issues of the three major building blocks of the current Internet, i.e., statistical multiplexing, best-effort paradigm, and an IP address-based reachability. New IP is a data plane technology that defines a new network packet specification, and new service capabilities enabled in the network nodes.

A New IP packet starts with the *Header Specification*, which specifies the boundary of the following *Shipping Specification*, *Contract Specification* and *Payload Specification*.

The *Shipping Specification* intends to change the current fixed types of addressing (i.e., IPv4 or IPv6) to being able to include all types of addresses in a flexible manner and accommodate different reachability scenarios.

The *Contract Specification* provides a series of apparatuses to facilitate new network capabilities, their functionalities and regulative conditions at the finest packet-level granularity. The network and routers fulfill the contract, with the assumption that the contract has been agreed between the packet sender/receiver and the network. New IP contract could be constructed from multiple contract clauses, each of which might include Action, Event/Condition and the associated Metadata. A Contract Clause depicts the processing that network nodes (which are upgraded to support New IP) would carry out on the packet when it traverses the network according to the predefined triggering event or condition. The Metadata contains semantics about the packet, the sender/receiver context information, or the network statistics, etc.

The *Payload Specification* divides the packet payload into multiple portions, such that when network congestion happens, the network nodes could drop some portions of the payload and allow the receiver to consume the residual information. This type of communication is named as Qualitative Communication [14] [15], which helps to mitigate re-transmission overhead and delay when faced with slow or congested network conditions.

III. OPTIMAL LATENCY GURANTEE FOR MULTIPLE CONCURRENT PACKETS

A. Single Packet Scenario

We consider the simplest scenario, in which the network will need to guarantee the in-time delivery of a particular packet. In other words, this particular packet could have the highest priority when being scheduled in the outgoing queue, compared to other packets without such requirement. We consider the end-to-end in-time delivery requirement is set as: $latency \leq d$, in which $latency$ denotes the incurred end-to-end latency for the packet delivery, d denotes the deadline constraint. The first router that the packet reaches is able to apprehend the number of hops information between itself and the destination, which is denoted as *n*. In other words, there are *n* number of hops between the first router and the destination, which means *n* number of routers are involved in the packet forwarding. The exemplary topology is shown in Figure 1.

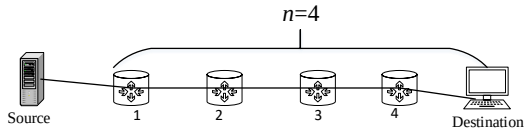

Figure 1. Example topology

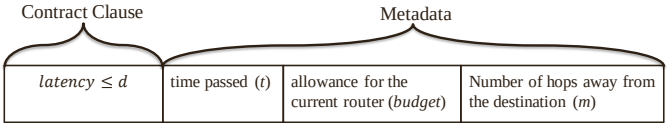

Figure 2. New IP header for packet with in-time guarantee requirement

The source can specify the in-time delivery requirement in the New IP header with the contract clause set to: $latency \leq d$. The metadata carries the following information, as shown in Figure 2:

- Time passed (t): it represents how much time has passed since the packet is sent out from the source. It is initialized to 0 by the source.
- Number of hops away from the destination (m): it represents the number of hops between the current router to the destination.
- Allowance for the current router ($budget$): it denotes the time duration that is allowed for the current router between the time when the packet arrives at the router and the time when the last bit of the packet gets transmitted to the next hop.

When the packet reaches the router 1, *t* is set to the time used to transport the packet from the source to the router 1 and *m* is set to *n*. The current router is allowed to have the time $budget$ to forward the packet to the next router, which is calculated as:

$$budget = \frac{d - t}{m} \quad (1)$$

The time $budget$ has two aspects: (1) If the router uses less time than $budget$, then it does not affect the following routers, but gives them more time budget to use. (2) If the router uses more time than the budget, it will affect the rest of the routers. However, it does not mean the packet has to be dropped if the budget cannot be met. The hybrid policy used in the router for the particular packet is that it puts the packet at the highest priority, but tries its best.

When the packet reaches the intermediate routers (e.g., the router 2, 3 and 4), t is set to be the time used to transport the packet from the source to the current router, m is deducted by 1 every time the packet is being forwarded by a router, $budget$ is calculated accordingly. When an intermediate router finds that the residual time $(d - t)$ is not enough for it to transfer the packet to the destination, the packet is dropped and the in-time guarantee fails because it is not a realistic requirement. The

intermediate router needs to reply the source with a response message, which is also designed in the embodiment of New IP. The metadata contains the following information as shown in Figure 3:

- $dsuggested$ is the suggested deadline based on the current situation, i.e., the number of hops from the current router to the destination and the average latency incurred at previous routers. $dsuggested$ is calculated based on the below equation, where σ gives a small amount of extra time added to the suggested deadline configuration.

$$dsuggested = \frac{t * n}{n - m} + \sigma \tag{2}$$

- *Unsuccessful* flag is to indicate that the packet delivery failed due to the reason that the specified deadline cannot be met.
- *Number of hops away from the source* is set to be $(n - m)$, which is used to notify the source at which intermediate router the packet gets dropped and the deadline expires.

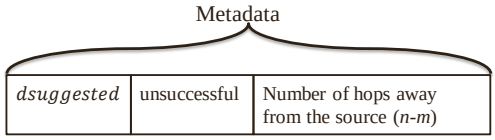

Figure 3. New IP header for reply message from an intermediate router

B. Multiple Concurrent Packets Scenario

In reality, routers might need to guarantee the in-time delivery for packets from multiple flows. We assume a router receives packets from the ingress ports with the in-time guarantee contract and latency related metadata specified as proposed in Section III.A. There is a dedicated queue for latency guaranteed packets for each outgoing port, called Latency Guarantee Queue (**LGQ**), as shown in Figure 4. All packets forwarded by the router with latency guarantee contract clause are put in the LGQ. The packets in the LGQ have the highest priority to be scheduled compared to other packets without deadline constraints.

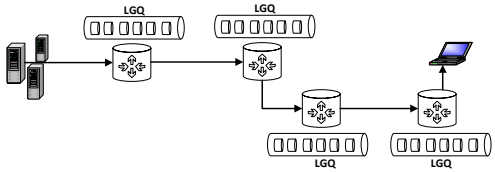

Figure 4. Latency guarantee queue (LGQ)

The time duration a packet stays in the router depends on how the other packets are scheduled, which is called ***stay time*** of the packet in the router. A packet's stay time equals to the total stay time of the packets scheduled before it and its own header processing, propagation and transmission delay in the router. The optimization problem is formulated with the objective to minimize the total stay time of the packets in the router, which have in-time guarantee contract and are going to be forwarded through the same outgoing port. Given there are total K total number of such packets:

$$\min \sum_{k=1}^{K} t_k \tag{3}$$

$$s.j.: t_k \le b_k \text{ for } k = 1 \dots K \tag{4}$$

The problem is that we want to find a permutation of $\{1, \dots K\}$ ($\sigma: \{1, \dots K\} \to \{1, \dots K\}$) which represents the scheduling order of the packets (i.e., the positions of the packets in the queue from front to rear), such that the total stay time can be minimized. For a packet at the k^{th} position, its stay time is calculated as:

$$t_{\sigma(k)} = \sum_{i=1}^{k} P_{\sigma(i)} \tag{5}$$

where $P_{\sigma(i)}$ is the stay time when the packet at the ith position is served. The optimization problem is converted to:

$$\sum_{k=1}^{K} t_{\sigma(k)} = K * P_{\sigma(1)} + (K-1) * P_{\sigma(2)} + \dots + P_{\sigma(K)} \tag{6}$$

TABLE I. PACKETS IN A ROUTER

Identifier	Budget	Stay Time
1	b_1	$P_{\sigma(1)}$
2	b_2	$P_{\sigma(2)}$
3	b_3	$P_{\sigma(3)}$

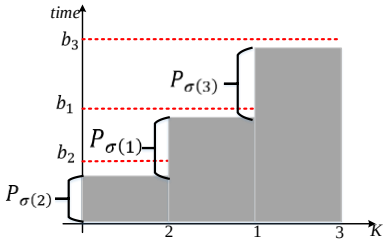

Figure 5. Simple example to illustrate the optimization problem

We use a simple example with the packets as shown in TABLE I. If the permutation is $\sigma: \{1, 2, 3\} \to \{2, 1, 3\}$), then the total stay time of the three packets is calculated as:

$$\begin{aligned} t_1 + t_2 + t_3 &= P_{\sigma(2)} + \left(P_{\sigma(2)} + P_{\sigma(1)}\right) \\ &\quad + \left(P_{\sigma(2)} + P_{\sigma(1)} + P_{\sigma(3)}\right) \\ &= 3 * P_{\sigma(2)} + 2 * P_{\sigma(1)} + P_{\sigma(3)} \end{aligned}$$

The problem is firstly to find all feasible solutions, then find the optimal one that minimizes the area in grey, as shown in Figure 5. We define a feasible solution as a schedule under which all packets' per-hop deadlines could be met.

The algorithm as shown in TABLE II. is proposed to solve the optimization problem by using backtracking method. A typical backtracking algorithm will need the procedures as follows. The constraint is shown in (4) and the objective is shown in (6).

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-964-5

- $discard(constraint, s)$: return true only if the partial scheduling s is not worth going further.
- $accept(constraint, s)$: return true if s is a solution that satisfies all constraints, and false otherwise.
- $first(constraint, s)$: generate the first extension of candidate s, which means the first packet in the queue is selected and added to s.
- $next(constraint, a)$: generate the next alternative extension of a candidate, after the extension a, which means another different packet is selected to be next in the queue.
- $record(constraint, s)$: record the solution that satisfies all constraints.
- $calculate(objective, s)$: calculate the objective result for the solution.
- $select(smallest, objective, s)$: compare the current solution with the selected solution to make sure the selected solution always has the minimal objective result.

TABLE II. BACKTRACKING ALGORITHM

```
backtracking(objective, constraint, s)
1   if discard(constraint, s), then
2      return;
3   if accept(constraint, s), then
4      record(constraint, s)
5      o = calculate(objective, s)
6      select(smallest, objective, s)
7   a = first(constraint, s)
8   while a ≠ NULL do
9      backtracking(objective, constraint, a)
10     a = next(constraint, a)
```

The enumeration procedure can find the optimal solution for any problem with constraints. But it takes too much time, even with the proposed algorithm using backtracking, if the number of packets that needs to be scheduled is large in a router.

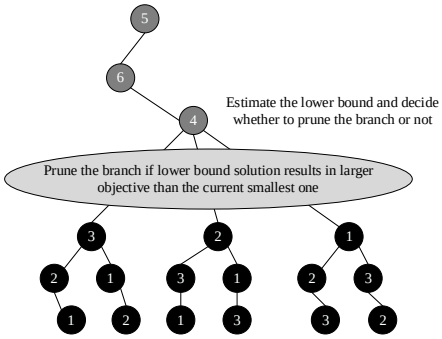

Figure 6. Example of pruning a branch

So, we improve the algorithm by leveraging Branch and Bound concept:

- For a current extension that is partial, estimate the lower bound, stop extending from the current candidate and prune the whole branch rooting from it.
- For the example as shown in Figure 6, the lower bound can easily be calculated by sorting the budgets in decreasing order. For example, for packet 1, 2, 3, the budgets b_1, b_2, b_3 in decreasing order are b_2, b_1, b_3, thus the branch of 2, 1, 3 results in the lower bound of the entire branch extended from 5, 6, 4.
- If this lower bound solution (i.e., 5, 6, 4, 2, 1, 3) cannot obtain a smaller objective than the current smallest one, then the entire branch should not be traversed and can be pruned from the recursive iteration.

In order to significantly reduce the running time of the algorithm, instead of minimizing the objective, the algorithm can be stopped when the first solution that satisfies the constraints is found. Such solution is called a feasible schedule to the packets that are being considered in the algorithm.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed backtracking algorithm. At the end of Section III.B, the proposed Backtracking Algorithm with Branch and Bound (BABB) may stop at a feasible schedule, which is a permutation of the packets that need to be scheduled and satisfies all the constraints specified in (4). It is denoted as One Possible Feasible Schedule (OPFS) in the following of the section.

TABLE III. PACKET SET EXMAPLE

Packets	Deadline	Transmission Time
P1	10	5
P2	14	2
P3	15	1
P4	6	3

TABLE IV. OPFS SCHEDULE BY EXMAPLE

OPFS	Deadline	Transmission Time	Dwell Time
P4	6	3	3
P1	10	5	8
P2	14	2	10
P3	15	1	11

Firstly, we take a look at a simple example of packet set, as shown in TABLE II. The LGQ of the router contains a set of packets, which are associated with the properties of deadline and transmission time. We assume the unit of deadline and transmission time is ms. An OPFS schedule is illustrated in TABLE IV. The average stay time is 8 ms.

TABLE V. BABB SCHEDULE BY EXAMPLE

BABB	Deadline	Transmission Time	Dwell Time
P3	15	1	1
P4	6	3	4
P1	10	5	9
P2	14	2	11

The proposed BABB algorithm is able to find the optimal schedule as shown in TABLE V. The average stay time is 25/4=6.25 ms, which decreases nearly 30% compared to the OPFS schedule.

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-964-5

Besides BABB and OPFS, the following three scheduling schemes are included in the performance evaluation and comparisons:

- First In First Out (FIFO): which is the same as FCFS in [9]. The packets are scheduled according to their arrival time at the outgoing queue of the router. The packet which arrives earliest is scheduled firstly.
- Smallest Transmission Time First (STTF): which is similar to SJF in [10]. The packets are scheduled based on the incremental order of the transmission time. The transmission time is proportional to the packet size if the outgoing link bandwidth is fixed. Thus, in STTF, the minimum-sized packet is scheduled firstly.
- Largest Transmission Time First (LTTF): the packets are scheduled based on the decremental order of the transmission time. Thus, in LTTF, the most bulky packet is scheduled firstly.

The simulator is built in C++. Two types of performances are being evaluated: (1) Packet delivery success rate, which is defined as the ratio of the packets that get scheduled appropriately and meet their corresponding deadline constraints. (2) Average stay time of the packets which could satisfy their deadline constraints under an adopted scheduling scheme.

The packets' transmission time and deadline are deliberately designed to make sure that there is at least one feasible schedule, under which all packets could be transmitted out of the router within their corresponding allowance. The feasible schedule is denoted as OPFS, as introduced above. The transmission time of the packets is randomly generated in the range [1,10] ms, then the deadline of each packet is assigned by adding some extra time compared to its stay time. **Deadline gap ratio** is defined as the ratio between the upper bound of this additional time and the transmission time upper bound. For example, if the deadline gap ratio is 3, then the deadline of a packet is given by adding a random number between [0, 10*3] = [0, 30] ms to the stay time. The packets' order is then shuffled to mimic the arrival time of those packets in the router.

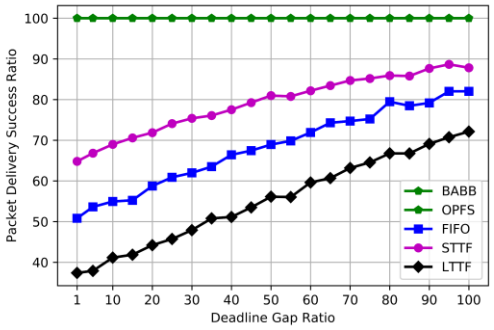

Figure 7. Packet delivery success ratio vs. deadline gap ratio

Figure 7 shows the packet delivery success ratio versus the deadline gap ratio with different packet schedule schemes. With the above simulation configurations, BABB and OPFS can always achieve the 100% packet delivery success ratio. However, if FIFO, STTF or LTTF is used, the stay time of some packets is not able to meet the deadline expectations. No matter how big the deadline gap ratio is, the positions of the packets to be scheduled in the outgoing queue are not appropriately manipulated to satisfy all packets' deadline constraints with FIFO, STTF or LTTF. The packet delivery success ratio of FIFO, STTF or LTTF increases along with the increment of the deadline gap ratio. BABB and OPFS proposed in this paper, on the other hand, guarantee all packets to be able to reach the receivers successfully without missing their deadlines, even when deadlines are very tight.

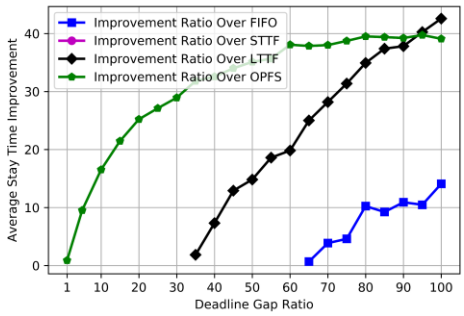

Figure 8. Average stay time improvement ratio vs. deadline gap ratio

Figure 8 shows the average stay time improvement ratio of BABB over the other scheduling schemes versus the deadline gap ratio. It is noticeable that in Figure 8, the improvement ratio of BABB over LTTF is plotted starting from the deadline gap ratio of 35, over FIFO is plotted starting from the deadline gap ratio of 65, while the line of improvement ratio of STTF is missing. The reasons are given as follows: Since we only evaluated the average stay time for those packets which could satisfy their deadline constraints, it means that the dropped packets due to missing deadline are not counted. Thus, the comparison of average stay time is not fair between BABB and FIFO/STTF/LTTF. STTF schedules the packets with the smallest transmission time firstly, those packets with larger transmission time are dropped eventually. According to (6), the total stay time for those successfully transmitted packets with STTF always has the minimal value. When the deadline gap ratio becomes large enough, BABB can improve over FIFO and LTTF. In the scenario that all packets are successfully transmitted, the fair comparison between BABB and OPFS is also shown in Figure 8. BABB achieves the minimal average stay time, while OPFS is the first solution that satisfies the constraints and the backtracking process stops at this point. As a result, BABB always accomplishes better average stay time performance than OPFS by 10% to 40% when the deadline gap ratio increases from 10 to 100.

Next, we evaluate the impact of the number of packets in the outgoing queue to the two types of considered performances. Figure 9 shows the packet delivery success ratio versus the packet number. BABB and OPFS proposed in the paper can always achieve the in-time guarantee for all packets, no matter how many packets are in the outgoing queue, which is very captivating property for latency-sensitive packet forwarding. It means that BABB or OPFS can be used in different types of network nodes, whether they have low or high volume of traffic. For other scheduling schemes, the packet success ratio declines

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-964-5

with the number of packets. When the packet number reaches 200, the packet delivery success ratio of FIFO, STTF or LTTF does not fluctuate much. The packet delivery success ratio of LTTF is the worst, with only 30% success rate.

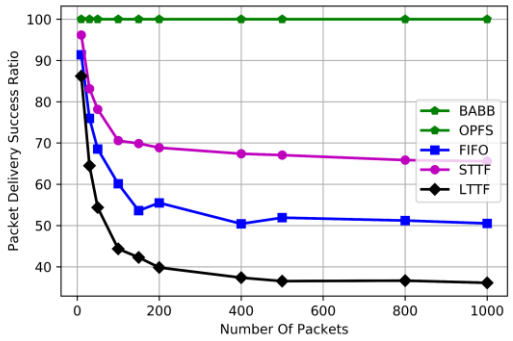

Figure 9. Packet delivery success ratio vs. packet number

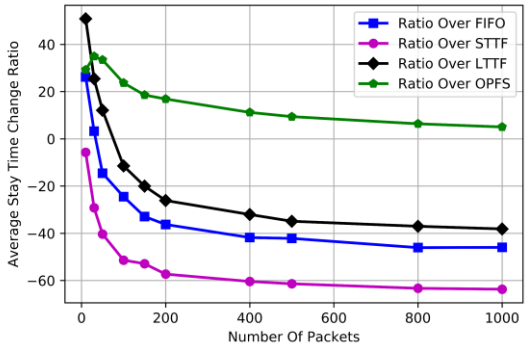

Figure 10. Average stay time change ratio vs. packet number

Figure 10 shows the average stay time change ratio of BABB over other scheduling schemes. As we explained earlier, this comparison only makes sense when the number of successfully delivered packets is the same. However, when the number of packets in the outgoing queue increases, FIFO, STTF and LTTF's packet delivery success ratio drops rapidly. The number of packets counted for the average stay time calculation becomes much less than the one counted in BABB and OPFS. We only draw those points with negative values to show that FIFO, STTF or LTTF are not a desirable scheduling scheme for packets with in-time guarantee, since they would cause too many packets dropping due to missing latency deadline. On the other hand, we can observe that the improvement ratio of BABB over OPFS decreases when there is a very large number of latency-sensitive packets in a router's outgoing queue. Thus, when the number of concurrent packets that require in-time guarantee becomes large, an OPFS scheme is good enough to be adopted to achieve the precise latency performance with reasonable low processing overhead in the router.

V. CONCLUSION

This paper leverages the New IP framework to carry an in-time guarantee contract, as well as the associated deadline constraint and other metadata in the packet, such that each intermediate router on the path from the source to the destination can execute more sophisticated scheduling on multiple packets on the same outgoing port instead of traditional statistical multiplexing. The proposed backtracking solution with bound-and-branch improvement can achieve the minimal average stay time of the packets which require in-time guarantee, and the successful delivery ratio of packets is maximized compared to any other scheduling schemes.

REFERENCES

[1] FG-NET-2030, Sub group 2, "New Services and Capabilities for Network 2030: Description, Technical Gap and Performance Target Analysis," 2019.

[2] FG-NET-2030, "Network 2030 - A Blueprint of Technology, Applications and Market Drivers Towards the Year 2030 and Beyond," May 2019.

[3] L. Dong, L. Han, and R. Li, "Support Precise Latency for Network Based AR/VR Applications with New IP," EAI MobiMedia 2020.

[4] R. Braden, D. Clark, and S. Shenker, "RFC 1663: Integrated Services in the Internet Architecutre: an Overview," IETF, Jun. 1994.

[5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "RFC 2475: An Architecture for Differentiated Services," IETF, Dec. 1998.

[6] L. Han, Y. Qu, L. Dong and R. Li, "A Framework for Bandwidth and Latency Guaranteed Service in New IP Network," IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2020, pp. 85–90.

[7] L. Dong and L. Han, "New IP Enabled In-Band Signaling for Accurate Latency Guarantee Service," 2021 IEEE WCNC, pp. 1–7.

[8] B. Vamanan, J. Hasan, and T. Vijaykumar, "Deadline-aware datacenter TCP (D2TCP)," ACM SIGCOMM Computer Communication Review, vol. 42, no. 4, pp. 115–126, 2012.

[9] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron, "Better never than late: Meeting deadlines in datacenter networks," SIGCOMM Computer Communication Review, vol. 41, no. 4, pp. 50–61, 2011.

[10] C.-Y. Hong, M. Caesar, and P. Godfrey, "Finishing flows quickly with preemptive scheduling," ACM SIGCOMM Computer Communication Review, vol. 42, no. 4, pp. 127–138, 2012.

[11] V. K. Gopalakrishna, Y. Kaymak, C. Lin and R. Rojas-Cessa, "PEQ: Scheduling Time-Sensitive Data-Center Flows using Weighted Flow Sizes and Deadlines," 2020 IEEE 21st International Conference on High Performance Switching and Routing (HPSR), 2020, pp. 1–6.

[12] R. Li, K. Makhijani and L. Dong, "New IP: A Data Packet Framework to Evolve the Internet : Invited Paper," 2020 IEEE 21st HPSR, 2020, pp. 1–8.

[13] R. Li, A. Clemm, U. Chunduri, L. Dong, and K. Makhijani, "A New Framework And Protocol For Future Networking Applications," ACM Sigcomm NEAT workshop 2018, pp. 21–26.

[14] R. Li, K. Makhijani, H. Yousefi, C. Westphal, L. Dong, T. Wauters, and F. De Turck, "A Framework For Qualitative Communications Using Big Packet Protocol," ACM Sigcomm NEAT workshop 2019, pp. 22–28.

[15] L. Dong and R. Li, "In-Packet Network Coding for Effective Packet Wash and Packet Enrichment," IEEE Globecom Workshops, 2019, pp. 1–6.

[16] L. Dong, K. Makhijani and R. Li, "Qualitative Communication Via Network Coding and New IP : Invited Paper," 2020 IEEE 21st HPSR, 2020, pp. 1–5.

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-964-5