SDATP: An SDN-Based Adaptive Transmission Protocol for Time-Critical Services

Jiayin Chen, Qiang Ye, *Member, IEEE*, Wei Quan, Si Yan, Phu Thinh Do, Weihua Zhuang, *Fellow, IEEE*, Xuemin (Sherman) Shen, *Fellow, IEEE*, Xu Li, and Jaya Rao

***Abstract*—In this article, a comprehensive software-defined networking (SDN) based adaptive transmission protocol (SDATP) is proposed for the fifth generation (5G) communication networks. In SDATP, a slice-level customized protocol is developed for supporting time-critical services that require high reliability and low-latency, such as machine-type communication (MTC) services for industrial automation. To satisfy service quality requirement for an MTC service, we introduce in-network intelligence in the proposed protocol, by enabling the functionalities of in-path caching, in-path retransmission, in-network congestion detection and congestion control. To minimize the end-to-end (E2E) delay, we optimize the configuration of caching functionalities, including the number of enabled caching nodes, caching node placement, and probabilistic packet caching policy. Since the optimization problem is NP-hard, we simplify the problem by reducing the number of decision variables and propose a low-complexity algorithm to solve the simplified problem. Extensive simulation results are presented to validate the effectiveness of the proposed algorithm in terms of retransmission hops and its adaptiveness to network dynamics.**

***Index Terms*—5G, SDN, adaptive transmission protocol, in-path packet caching/retransmission, in-network congestion control.**

I. Introduction

Recent advancement of the fifth generation (5G) communication networking technologies has led to the development of applications with stringent bandwidth and quality-of-service (QoS) requirements [1], [2]. Due to the limited resources on transmission links and network servers/switches in current core networks, it is difficult to meet diversified end-to-end (E2E) QoS requirements [3]. To mitigate network congestion and improve QoS satisfaction, an efficient transport-layer protocol is required. However, under the existing distributed and ossified core network architecture, only best-effort E2E performance is achieved by current transport-layer protocols, e.g., the transmission control protocol (TCP) [4], due to slow reaction to the increased level of network congestion. Software-defined networking (SDN) is a promising networking technology to enhance traffic load balancing and reduce network congestion [5].

Jiayin Chen, Qiang Ye, Si Yan, Weihua Zhuang, and Xuemin (Sherman) Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, N2L 3G1 (email: {j648chen, q6ye, s52yan, wzhuang, sshen}@uwaterloo.ca)

Wei Quan is with School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing, P.R. China, 100044 (email: dr.wei.quan@ieee.org)

Phu Thinh Do is with Sendo Technology Joint Stock Company, Ho Chi Minh City, Vietnam (email: dopthinh@gmail.com)

Xu Li, and Jaya Rao are with Huawei Technologies Canada Inc., Ottawa, ON, Canada, K2K 3J1 (email: {xu.lica, jaya.rao}@huawei.com)

Within the SDN architecture, more fine-grained in-network control can be realized to improve the E2E performance. In [6], to realize early detection and fast response of congestion, the SDN control module is exploited to collect in-network statistics (e.g., buffer occupancy information) [6]. However, the congestion control is still performed at end hosts. The SDN architecture can reduce the E2E delay by enabling in-path packet caching and retransmission function. In [7], an SDN-based user datagram protocol (UDP) framework is proposed, where retransmission engine is activated in-path to detect packet loss and request for retransmission. To effectively utilize caching resources, McGarry *et al.* propose to enable caching functions on some of the in-network switches in wireless sensor networks [8]. However, those caching switches are to cache all received packets, which incurs redundant packet caching along the routing path with wasted storage resources and additional processing delay. In [9], instead of caching all the received packets, it is proposed to cache a packet with certain probability so as to eliminate repetitive caching. However, how to place the caching functions on network switches to further improve the utilization of caching resources is yet to be considered.

To support time-critical services (e.g., machine-type communication (MTC) for industrial automation) with stringent delay and packet loss requirements, a proper caching scheme, including caching node placement and packet caching probability at each caching node, is required to maximize the packet caching efficiency. Given a packet caching probability, activating more caching nodes leads to a lower caching buffer overflow probability but less efficient caching resource usage. Moreover, the location of caching node placement affects QoS provisioning. If a caching node is placed at the beginning of a network path, the cached packets are more likely to be requested for retransmission, indicating a better caching effectiveness. However, the performance gain (e.g., reduced retransmission hops for each packet) can be low. Therefore, the caching node placement should be optimized via balancing the trade-off between effective caching and performance gain. The correlation between caching node placement and packet caching probability poses technical challenges on the caching scheme design.

In this article, we propose a comprehensive SDN-based adaptive transmission protocol (SDATP) with in-path packet caching, including caching node placement and probabilistic packet caching on each caching node, to reduce E2E packet transmission delay for supporting time-critical services (e.g., MTC for industrial automation). To optimize the caching

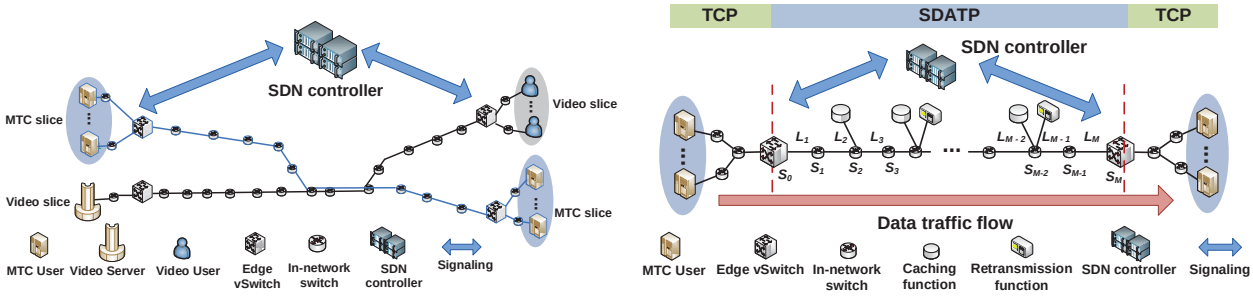


Fig. 1: Network slices for supporting differentiated services: a) a coexistence of network slices for MTC and video services; b) network slice for MTC service with customized protocol functionalities.

policy, a joint caching node placement and probabilistic packet caching problem is formulated as a mixed-integer non-linear program (MINLP) to minimize the number of average packet retransmission hops. However, the correlation of packet caching probabilities among nodes and the interaction between probabilistic caching and caching node placement make it challenging to solve the problem. For a tractable solution, we reduce the number of decision variables by taking into account the feature of in-path caching, and then develop a low complexity heuristic algorithm to solve the simplified problem. It is demonstrated that the optimized caching policy outperforms several other benchmark policies, in terms of the number of retransmission hops.

The remainder of this article is organized as follows: The system model is described in Section II. In Sections III, we present the proposed SDATP to support time-critical services, where we take MTC for industrial automation as an example. Simulation results are given in Section IV to demonstrate the performance of the proposed protocol, in comparison to an enhanced TCP. Finally, conclusions are drawn in Section V.

II. System Model

In this section, the system model under consideration is presented, including a network model, a description of protocol function modules, and traffic model, to support time-critical services such as in MTC for industrial automation.

A. Network Model

In 5G core networks, with SDN and network function virtualization (NFV) [10], [11], each service flow [1] aggregated from the wireless domain needs to go through a sequence of service functions, such as firewall and network address translation, to fulfill a flexible, cost-effective, and QoS guaranteed E2E service delivery. Therefore, a designed virtual network to support those service functions over the physical substrate network is called an embedded virtual network topology, with associated network servers, interconnected links, and provisioned resources, as shown in Fig. 1(a). To be compatible with the current transport layer protocol on end hosts, we assume the connections between end hosts and virtual switches[2] (i.e., vSwitches) at the edge of the core network are running a state-of-the-art TCP protocol. For accommodating data transmission from a set of E2E connections of a service, an edge vSwitch is deployed for traffic aggregation and other higher-layer network functionalities (e.g., transport-layer header processing). Our proposed customized protocol for a service operates over an embedded virtual network topology between a pair of sending and receiving edge switches, and is developed at "slice-level" for enhanced data transmission. We define one network *slice* to support an aggregated service flow as a combination of the embedded virtual network topology, with the set of allocated resources, and the customized protocol elements. Each network slice is differentiated and isolated from the slices for other service flows, to guarantee different levels of QoS. We focus on slice-level protocol customization, because traffic flows aggregated from a set of end connections belonging to the same service type should follow the same set of protocol operation rules along the configured routing path between a pair of edge vSwitches. For the flows belonging to different services, the protocol is customized to satisfy their differentiated QoS requirements, by adjusting protocol operation rules. Taking one MTC service for industrial automation as an example in Fig. 1(b), the aggregated MTC traffic flow traverses from edge vSwitch S_0 to edge vSwitch S_M. The virtual network topology established for the service has proper resource allocation and a configured routing path between edge switches. As shown in Fig. 1(b), the path between edge switches of an MTC service is a linear topology, which consists of M transmission links, denoted by $\{L_1, L_2, ..., L_M\}$, and $(M-1)$ in-network vSwitches, denoted by $\{S_1, S_2, ..., S_{M-1}\}$, and edge vSwitches S_0 and S_M. All the switches under the SDN architecture are OpenFlow vSwitches that are managed by the SDN controller. The edge-to-edge routing path is equipped with proper resources supporting higher-layer network function programmability. Conventional switches with only second-layer packet forwarding capability also exist, which are not depicted in Fig. 1(b) for clarity. In the following, a switch refers to an OpenFlow vSwitch.

For services that require caching functions, we denote the

[1] A service (traffic) flow refers to an aggregation of packets of the same service type passing through a pair of edge switches in the core network.

[2] A virtual switch refers to a softwarized network switch with virtualized resources to program higher-layer network functions, apart from the functionalities of traffic forwarding or traffic aggregation.

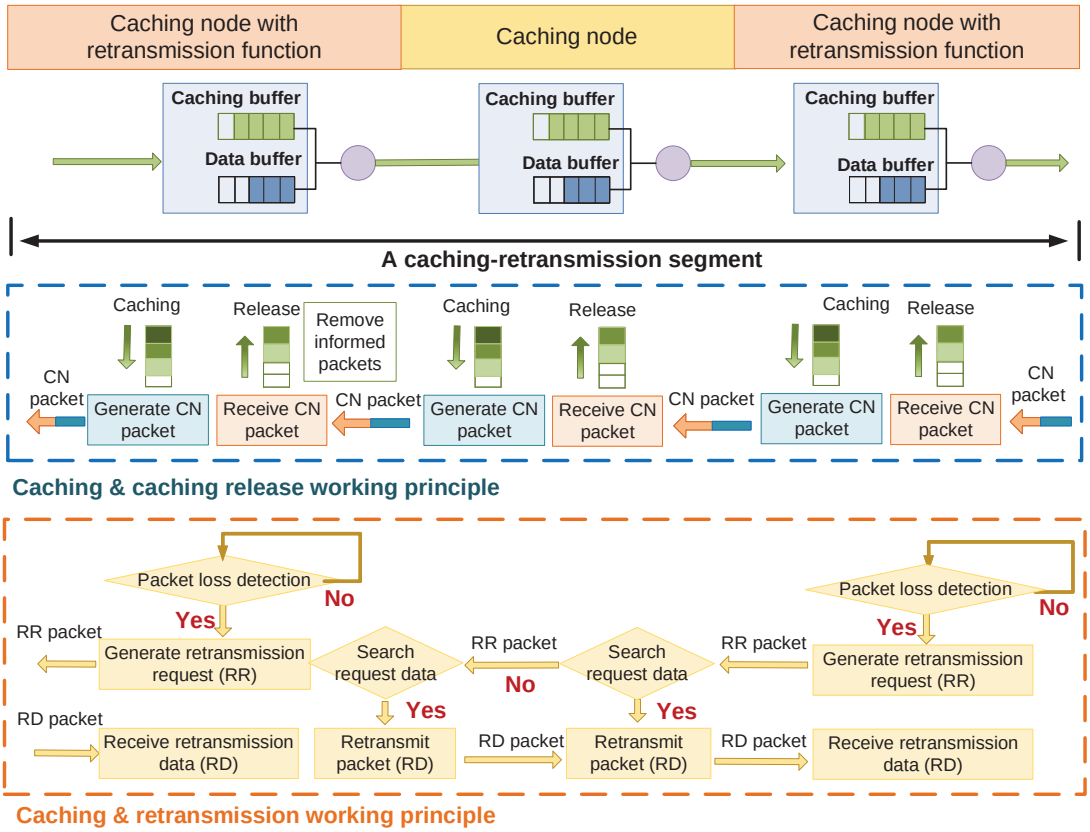


Fig. 2: An illustration of functionalities in a CR segment.

caching buffer size in packets for the $(M-1)$ in-network switches as $\{B_1, B_2, ..., B_{M-1}\}$. Packet loss at the m-th ($m = 1, 2, ..., M$) transmission hop is caused by congestion at switch S_{m-1} and transmission errors of link L_m.

B. Network Functionalities

In order to meet the high reliability and low latency requirements of MTC services, SDATP incorporates a set of in-network functionalities, including in-path caching and retransmission, which can be deployed and activated on demand.

1) Caching and Retransmission Functions: For a service that requires high reliability and low latency, caching and retransmission functions are activated at in-network switches to realize in-path packet loss detection and packet retransmission.

A network switch with activated caching function is defined as a caching node, which consists of a unique pair of caching and data transmission buffers, as shown in Fig. 2. Data packets are queued in the transmission buffer. During packet processing at one node, it duplicates each packet to its caching buffer with certain probability, referred to as probabilistic caching. Since all caching buffers have a limited size, buffer release is needed at the caching node to avoid buffer overflow. The procedures are shown as the caching and caching release working principle in Fig. 2. Time is partitioned into intervals of constant duration T_{CN} (in the unit of second). In every T_{CN} time interval, a caching notification (CN) packet is transmitted upstream to release cached packets in preceding caching nodes. The CN packet provides information of successfully cached packets at a caching node during the last T_{CN} interval. Cached packets are used for retransmission once packet loss happens, and a retransmission request is initiated by a retransmission node (which is an in-network switch with retransmission functionalities) including in-path packet loss detection and retransmission triggering. Between consecutive retransmission nodes, all the caching nodes and network switches/transmission links are referred to *one caching-retransmission (CR) segment*.

2) Caching and Retransmission Function Placement: The decision on which network switches should be activated with caching or retransmission functionality is a part of virtual network establishment procedure. For an MTC service with requirements of high reliability and low latency, caching function element is activated at both the sending and receiving edge switch, while the receiving edge switch also enables retransmission function elements, in order to minimize the maximum packet retransmission delay[3]. Specifically, the policy of placing caching and retransmission function depends on packet loss probability over each transmission hop, as discussed in Subsection III-D.

C. Traffic Model

Considering the temporal independence of access requests from different MTC devices, the MTC traffic arrivals are mod-

[3]Packet retransmission delay is the time duration from the instant that a retransmission request packet is sent (after detecting packet loss) to the instant that a retransmitted packet is received by the request node.

eled as a Poisson process [12], [13]. Thus, packet arrivals from an aggregated MTC traffic flow, consisting of the superposition of a number of flows from individual devices, follows a Poisson process, with the average arrival rate ranging from tens to a few hundreds of packets per second. For an aggregated MTC traffic flow, let Y denote the number of packet arrivals at edge switch S_0 during each caching release period (with duration T_{CN}), where Y follows a Poisson distribution with mean packet arrivals (denoted by λ) over T_{CN}.

III. Customized Protocol for Time-Critical Services

We propose a customized protocol for time-critical MTC services to achieve in-path packet loss detection and lost packet retransmission, and in-network congestion detection and congestion control, in order to minimize the E2E transmission delay for packet. The protocol function elements include connection establishment, data transmission, caching-based in-path packet retransmission, and caching-based congestion control. We also develop an optimization framework to determine the optimal number of enabled caching nodes, optimal caching node placement, and optimal caching probabilities for the enabled caching nodes.

A. Connection Establishment

Conventional TCP requires a *three-way* handshake to establish a connection before data transmissions. This connection establishment process is required for a distributed network paradigm, for the two end hosts to ensure the reachability of an E2E path. With SDN, the controller assists the connection negotiation and path configuration. Therefore, the three-way handshake process for connection establishment can be simplified, due to the following reasons:

(1) One purpose of the three-way handshake is to check whether there remains an available path between a client and a server. With a global view of the network, the SDN controller is able to check the path availability, which is more efficient than the hop-by-hop exploration;
(2) The three-way handshake facilitates the negotiation between two end hosts for the initial sequence number and acknowledgement number. We can utilize the SDN controller to assign or notify the initial sequence number to end hosts;
(3) To satisfy certain service requirements, the SDN controller can configure and update a routing path (a transmission pipeline) during the connection establishment.

Motivated by the preceding considerations, an SDN-based connection establishment mechanism is introduced as part of our proposed SDATP, to accelerate the connection establishment and reduce the signaling overhead. In SDATP, we exploit two-way handshake for connection establishment, including the initial sequence number exchanges between the end hosts. The controller is in charge of checking the path availability for the two-way connections. The first handshake establishes the connection for the forward direction, and the second handshake establishes the connection for the reverse direction. This SDN-based two-way connection establishment for supporting MTC services is presented in detail in our previous work [14].

B. Data Transmission

1) SDATP Data Packet Formats: We develop a set of new packet formats for efficient slice-level data transmission between a pair of edge switches. On one hand, with the SDN controller and in-network transmission control, the conventional TCP/IP header format is simplified to make the data transmission more efficient. For example, since packet loss detection in SDATP is a receiver-triggered method, the Acknowledgment field designed to acknowledge each received packet is not needed in the SDATP data packet header. The header formats should be supported by the OpenFlow-switches which extract the required matching fields (i.e., slice ID) in each received SDATP data packet to match the cached flow entries, to forward packet along the virtual network path. On the other hand, the SDATP introduces new functionalities to support enhanced data transmissions, such as in-path caching, caching-based retransmission, and caching-based congestion control. Therefore, two new fields in the SDATP packet header, Flag and Optional, are re-designed to generate several important types of packets, including data packet, retransmission data (RD) packet, retransmission request (RR) packet, and CN packet, to support the aforementioned enhanced protocol functionalities for data transmission.

2) Header Conversion/Reversion: For compatibility, between an end host and its corresponding edge switch, the packet transmission is based on the conventional TCP, whereas it follows the SDATP protocol between edge switches. Therefore, header conversion and reversion should be executed by the sending and receiving edge switches. When a sender sends a data packet based on TCP to the sending edge switch, the packet is converted to an SDATP data packet. When the packet arrives at the receiving edge switch, the SDATP packet is reverted to a TCP packet. To realize header conversion/reversion for data packets, we adopt one technology called *tunneling* [15]. That is, a TCP packet can be encapsulated as an SDATP packet by adding a new SDATP header over the TCP header, and can be decapsulated by removing the SDATP header, at the price of increasing the packet length.

C. Packet Retransmission

We briefly introduce the proposed packet retransmission scheme, including in-path receiver-triggered packet loss detection and caching-based packet retransmission. The detailed description of scheme is given in our previous work in [14].

1) Terminology for Packet Loss Detection: For each data traffic flow, at each retransmission node, a *content window list* is initiated and maintained, and an *expected packet list* is established according to the content window list.

Content Window List – Each retransmission node establishes the content window list to record the received packets, in which one content window describes a number of packets received in sequence. The list is updated once a new packet is received.

Expected Packet List – If the packet is expected by a retransmission node, its information is stored in an expected packet list at the node. When a retransmission node detects packet loss, it can select the lost packets from the list and trigger a retransmission request.

2) Thresholds for Packet Loss Detection: The original packet loss is detected based on the measurement of 1) the time interval of consecutive packet receptions, which is defined as *InterTime* at a retransmission node, or 2) the number of disordered packets that received by a retransmission node, called *Interarrival Counter*. The retransmission packet loss detection is based on the measurement of retransmission delay.

Interarrival Timeout – When InterTime exceeds an interarrival timeout which is called *expected interarrival time*, a retransmission request is triggered by this loss detection. The packet with the highest priority is selected from the expected packet list for retransmission. The threshold is determined by estimating the expected interarrival time.

Interarrival Counter Threshold – For the linear network topology of a single MTC service slice, disordered packet reception at one switch indicates packet loss, and packet retransmission can also be triggered according to the degree of packet disorder (i.e., packet disorder length). For example, if packet loss happens in the first CR segment, the first retransmission node sends an RR packet, and then the requested RD packet will be transmitted from a caching node. After the RD packet is received by following segments, it leads to disordered packet reception. To avoid false packet loss detection, the following retransmission nodes need to update the estimated packet disorder length. The updated disorder length is set as an interarrival counter threshold for the packet in the expected packet lists of corresponding retransmission nodes.

Retransmission Timeout – After retransmission is triggered, packet loss can happen during the transmission of RR and RD packets. Thus, a retransmission node should have the capability of detecting this retransmitted packet loss and resending the RR packet. Based on the measurement of each sampled packet retransmission delay, its expected value is estimated and used as the retransmission timeout threshold.

3) Retransmission for RD packet: If a retransmission node detects packet loss, it will generate and send an RR packet to its preceding caching node(s) in its CR segment through an upstream data link, and the expected packet list is updated accordingly. An illustration of how caching and retransmission functions cooperate within one CR segment to realize packet loss recovery is given in Fig. 2. When an upstream caching node receives the RR packet, it identifies and searches the requested data in caching buffer. If the data are found, the caching node will send out the RD packet to the retransmission node. If the data are not found, the current caching node will forward the RR packet upstream. The RR packet will be forwarded consecutively by the caching nodes until the requested data are found.

D. Probabilistic Caching Optimization

We propose an optimized packet caching policy under caching resource constraints at each caching node, to determine the optimal number of enabled caching nodes, caching node placement, and packet caching probabilities, which minimizes the average number of packet retransmission hops.

With an embedded network topology supporting one service as in Fig. 1(b), packet loss can happen between edge switches with packet loss probability (p_m) over link $m\,(m = 1, 2, ..., M)$, and an RD packet is retransmitted for loss recovery. In TCP, lost packets are retransmitted from the sender, and the required retransmission hops (RRHs) for each lost packet between edge switches is M. For SDATP, suppose that a sequence of $N\,(\leq M)$ network switches are activated as caching nodes from the sending edge switch to the receiving edge switch, and the set of indexes for the activated caching nodes are the indexes of the corresponding network switches, denoted by $\mathbb{C}(N) = \{C_1, C_2, ..., C_N\}$. With caching buffer capacity B_{C_n} (i.e., maximum number of cached packets) at switch S_{C_n}, the number of packets that can be cached at S_{C_n} is limited to B_{C_n}. Let $P_c(C_n)$ denote the caching probability of a packet passing through caching node S_{C_n}. A lost packet can be retransmitted by an in-path caching node, say S_{C_n}, instead of the sender. If a packet cached at S_{C_n} is requested for retransmission, at least C_n transmission hops can be avoided for packet retransmission. As compared with TCP, the reduction in the average number of RRHs for packets cached at S_{C_n} during T_{CN} is denoted by $R_S(C_n)$.

1) Objective: Since RRHs are proportional to time and transmission resources consumed for packet retransmission, the performance gain of in-path caching-based retransmission is represented by the ratio of total number of eliminated RRHs for packets cached at all caching nodes over the total number of RRHs for all lost packets during T_{CN} required by TCP. Our objective is to maximize the performance gain $G(N)$ by which the optimal number of enabled caching nodes, N^*, optimal caching node placement $\mathbb{C}(N^*)$, and optimal packet caching probabilities $\mathbb{P}(N^*) = \{P_c(C_1), P_c(C_2), ..., P_c(C_N)\}$ are obtained.

2) Optimized Probabilistic Caching: If a caching node is placed at the beginning of a network path, only a small number of hops (C_n) is avoided for each retransmitted packet, but the cached packets have a high probability to be requested for retransmission (i.e., effective caching proportion). Therefore, in terms of caching node placement $\mathbb{C}(N)$, there is a trade-off between effective caching proportion and reduced retransmission hops for each packet. To evaluate the gain achieved by activating one switch (S_i) as a caching node, we define caching weight $W_i\,(i = 1, 2, ..., M-1)$ as the achieved performance gain at the cost of one unit of caching resources. The number of caching nodes, N, imposes an upper limit on the available caching resources. A larger N value means more caching resources, which can be beneficial to guarantee a smaller caching buffer overflow probability. On the other hand, if more caching nodes are activated, the packets will be distributively cached at more swithches, which leads to low utilization of switches with a large caching weight W_i and a decrease in performance gain. Therefore, in terms of optimal N value, there is a trade-off between low buffer overflow probability and high utilization of resources at switches with large caching weight W_i. A maximum performance gain can be achieved through balancing the aforementioned two pairs of trade-off,

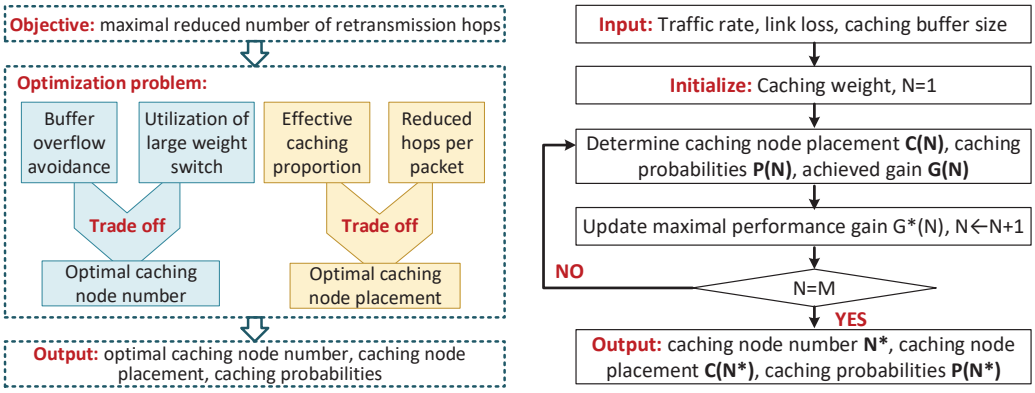


Fig. 3: An illustration of probabilistic caching: a) problem description; b) heuristic algorithm flowchart.

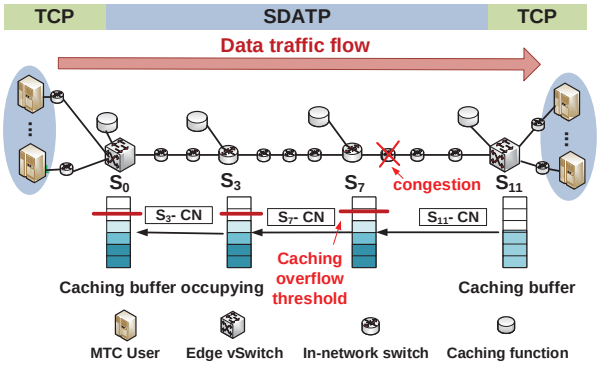


Fig. 4: Caching-to-caching congestion control in a virtual network with N=4.

as described in Fig. 3(a). Specifically, an optimization problem for probabilistic caching can be formulated as a MINLP with decision variables $\mathbb{P}(N)$, N and $\mathbb{C}(N)$, which is a proved as an NP-hard problem.

For a tractable solution, we reduce the number of decision variables by deriving $\mathbb{P}(N)$ in terms of $\mathbb{C}(N)$, traffic arrival rate and caching buffer size. We develop a low-complexity heuristic algorithm named probabilistic caching (*PC*) scheme to find the solution of the problem, as highlighted in Fig. 3(b). In order to determine caching node placement for a given N, caching nodes are selected based on their W_i values. Then, $\mathbb{C}(N)$ is determined, $\mathbb{P}(N)$ is derived, and $G(N)$ is calculated. In comparison with $G(N)$, the maximal performance gain $G^*(N)$ is updated after current iteration. After iterating N from 1 to $(M-1)$, the optimal number of caching nodes N^* and the corresponding $\mathbb{C}(N^*)$ and $\mathbb{P}(N^*)$ are determined to achieve maximal $G^*(N)$.

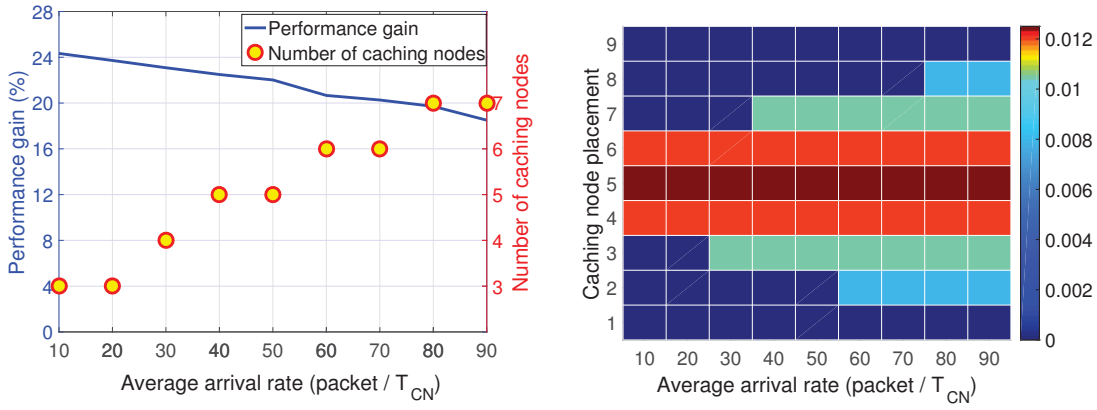
E. Congestion Control

For each virtual network, data transmissions between an end host and an edge switch are based on TCP, while communications between two edge switches use the proposed SDATP. Since multiple virtual networks share resources on some transmission links, congestion happens if a link utilization is close to 100%, leading to packet loss.

To overcome the link congestion problem, caching-to-caching congestion control is applied between two edge switches, while the TCP congestion control is employed between an end host and an edge switch. The detection of congestion is realized by analyzing CN packets sent from downstream caching nodes to upstream caching nodes. Each CN packet carries the information of how many packets are received within last T_{CN} in *received packet number* field, and the ratio of the available caching space over the caching buffer size in *caching state information* field. After receiving a CN packet sent by S_{C_n} $(n = 2, ..., N)$, the upstream caching node $S_{C_{n-1}}$ calculates the number of lost packets over the link between $S_{C_{n-1}}$ and S_{C_n}, and a large number of lost packets indicates high level congestion over the link. In addition, a smaller available buffer space ratio corresponds to a higher risk of caching buffer overflow at S_{C_n}.

Fig. 4 illustrates caching-to-caching congestion control in a virtual network with 4 caching nodes (i.e., S_0, S_3, S_7, and S_{11}), and congestion happens between S_7 and S_{11}. Node S_7 detects the congestion from a received CN packet, indicating packets loss happens between S_7 and S_{11}, and starts to lower its sending rate to alleviate the congestion. The remaining caching buffer space of S_7 decreases because of its lowered sending rate and reduced number of released cached packets. Node S_3 receives the CN packet including the remaining caching buffer space of S_7, and compares it with a threshold. If the remaining space is smaller than the threshold, node S_3 estimates that S_7 is at the risk of caching buffer overflow, and decreases its sending rate to reduce the risk. Based on the same procedure, congestion is detected and spread out to each upstream caching node, until the sending edge switch (S_0) is reached. After the edge switch estimates the potential caching buffer overflow at S_3, it decreases the sending rate and, at the same time, adjusts the rate of replying ACKs to the sender according to its remaining caching buffer space.

When congestion happens, the reaction time for a sender to reduce its sending rate depends on the setting of caching overflow threshold. The threshold guides in-network caching nodes to detect congestion and spread out the information upstream to the sender. Since a smaller caching overflow threshold achieves a faster overflow estimation for each caching node,



(a) (b)

Fig. 5: Scheme adaptiveness to the traffic arrival rate: a) performance gain ($G(N)$); b) caching node placement.

the sender experiences a faster reaction for the congestion resolution. However, due to a larger caching overflow threshold, more packets will be sent out from the sender at the premise of not aggravating the congestion. Therefore, a lower E2E transmission delay is achieved with reduced waiting time at the sender. The caching overflow threshold is an important design parameter, with the consideration of its impact on both congestion reaction time and E2E transmission delay.

IV. A Case Study

In this section, numerical results are presented to demonstrate the effectiveness of the proposed SDATP protocol supporting MTC service, in terms of reduced retransmission delay through in-path caching. To evaluate the performance, both analytical results and Monte Carlo simulation results are provided, for the network scenario shown in Fig. 1(b). Considering one service flow aggregated from a set of MTC connections, the routing path between the sending edge switch and the receiving edge switch for this MTC flow is linear, which consists of 10 transmission hops. When packets are transmitted between two edge switches, packet loss may happen due to network congestion and link errors. We assume packet loss probability is identical over each hop, and the loss probability of the entire path is 0.5%. We consider limited caching buffer capacity in the caching scheme design, which affects the total reduced retransmission delay. For each switch, caching buffer size B_m for the aggregated service flow is an integer randomly distributed between 5 and 20 (packets).

A. Protocol Adaptiveness

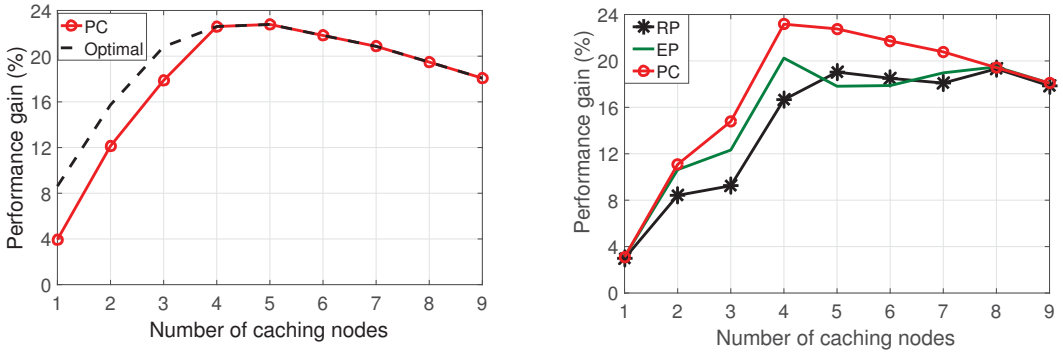
First, we evaluate the adaptiveness of the proposed SDATP protocol to varying network conditions. When the traffic load changes, caching strategies, including caching node placement and packet caching probabilities, should be updated to achieve consistently maximal performance gain.

In some MTC use cases (e.g., smart alarming system), the number of active MTC devices varies with time, which leads to different traffic volumes during peak and off-peak hours. We change average arrival rate (λ) of a single MTC flow from 10 packets per T_{CN} to 90 packets per T_{CN}. Recall that the performance gain with packet caching indicates the improvement of using in-path caching on the number of retransmission hops, and that caching node placement and caching probabilities are determined by the proposed probabilistic caching scheme. Fig. 5(a) shows how the number of caching nodes and performance gain change with the traffic arrival rate. More caching nodes are activated upon a traffic load increase to guarantee caching buffer reliability. However, the caching gain decreases because some nodes with a small weight are selected to accommodate the increasing traffic. Fig. 5(b) shows the optimal caching node placement versus traffic load, where node weights W_i are differentiated with colours, with the dark blue block representing unactivated nodes.

B. Performance Evaluation

To evaluate the performance of the proposed probabilistic caching scheme, both analytical and simulation results of the performance gain are shown in Fig. 6. The analytical results demonstrate the maximal gain, $G^*(N)$, achieved through the probabilistic caching scheme. In the Monte Carlo simulation, caching nodes are placed following the designed probabilistic caching scheme. We simulate the transmission of one million packets from the sending to receiving edge switches. During the transmission, packet loss may happen based on the link loss probabilities, which is approximatively 0.05% for each hop. The lost packets are retransmitted from the caching nodes, and the delay of retransmitting the packets are measured and compared with that of retransmitting from the sending edge switch to obtain the performance gain.

To demonstrate that the proposed heuristic algorithm achieves a near-optimal solution, we compare the analytical performance gain achieved by the proposed heuristic algorithm and the optimal gain achieved by brute-force algorithm in Fig. 6(a), where the traffic arrival rate is 50 packets per T_{CN}. It is shown that the optimal number N^* of activated caching nodes is 4 and the gap of optimal performance gains at N^* between the proposed heuristic algorithm and the brute-force method is small.



(a) (b)

Fig. 6: The performance gain ($G(N)$) obtained by different caching schemes: a) analytical results; b) simulation results.

The proposed caching scheme is also compared with two other in-path caching schemes, namely *RP*, and *EP* schemes. The RP scheme randomly selects in-path caching nodes; the EP scheme chooses in-path caching nodes, so that the packet loss probabilities between consecutive caching nodes are approximately equalized. For comparison, we set traffic arrival rate at 50 packets per T_{CN}. Fig. 6(b) plots the simulation results of the performance gain. It is seen that the optimal number of activated caching nodes of the proposed PC scheme is 4, which is consistent with the analytical results in Fig. 6(a). At the point of activating 4 caching nodes, nearly 24% of retransmission hops can be avoided by the proposed caching scheme, which outperforms the other two methods. Moreover, from Fig. 6(a) and Fig. 6(b), the tendency of performance gain with the varying number of caching nodes is consistent, and there exists an optimal performance gain by balancing the trade-off between low buffer overflow probability and high utilization of resources at switches with large caching weight W_i.

V. Conclusion

In this article, the customized SDATP is proposed to support services with stringent delay and high reliability requirements, such as MTC services for industrial automation. In SDATP, in-network intelligence is introduced, including in-path caching and retransmission functionalities and in-network congestion control. Furthermore, the SDATP is enhanced by jointly optimizing the caching node placement and probabilistic packet caching, which achieves minimum E2E packet delay with reduced retransmission overhead. We simplify the original optimization problem and propose a low-complexity heuristic algorithm. Through computer simulation, we evaluate the adaptiveness of the proposed protocol under a varying traffic load, and show that the proposed algorithm achieves consistently high performance gain. The proposed in-path caching scheme outperforms two other schemes in terms of the effectiveness of retransmission hops reduction.

References

[1] Y. Meng, W. Zhang, H. Zhu, and X. Shen, "Securing consumer IoT in the smart home: architecture, challenges, and countermeasures," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 53–59, 2018.

[2] H. Li, H. Zhu, S. Du, X. Liang, and X. Shen, "Privacy leakage of location sharing in mobile social networks: Attacks and defense," *IEEE Trans. Depend. Sec. Comput.*, vol. 15, no. 4, pp. 646–660, 2018.

[3] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5G: RAN, core network and caching solutions," *IEEE Commun. Surv. & Tutor.*, vol. 20, no. 4, pp. 3098–3130, 4th Quarter 2018.

[4] J. Postel, *Transmission Control Protocol,* RFC 793, 1981.

[5] Q. Ye, J. Li, K. Qu, W. Zhuang, X. Shen, and X. Li, "End-to-end quality of service in 5G networks: Examining the effectiveness of a network slicing framework," *IEEE Veh. Technol. Mag.*, vol. 13, no. 2, pp. 65–74, 2018.

[6] Y. Lu, Z. Ling, S. Zhu, and L. Tang, "SDTCP: Towards datacenter TCP congestion control with SDN for IOT applications," *Sensors*, vol. 17, no. 1, p. 109, Jan. 2017.

[7] M. H. Wang, L. W. Chen, P. W. Chi, and C. L. Lei, "SDUDP: A reliable UDP-Based transmission protocol over SDN," *IEEE Access*, vol. 5, pp. 5904–5916, Apr. 2017.

[8] M. P. McGarry, R. Shakya, M. I. Ohannessian, and R. Ferzli, "Optimal caching router placement for reduction in retransmission delay," in *Proc. ICCCN'11*, Aug. 2011, pp. 1–8.

[9] H. Chen, D. Fang, X. Chen, F. Chen, X. Gong, B. Zhou, and L. Qin, "A reliable transmission protocol based on dynamic link cache," in *Proc. iThings/CPSCom'11*, Oct. 2011, pp. 752–755.

[10] O. Alhussein, P. T. Do, J. Li, Q. Ye, W. Shi, W. Zhuang, X. Shen, X. Li, and J. Rao, "Joint VNF placement and multicast traffic routing in 5G core networks," in *Proc. IEEE GLOBECOM*, Dec. 2018, pp. 1–6.

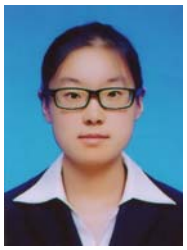
[11] J. Li, W. Shi, Q. Ye, W. Zhuang, X. Shen, and X. Li, "Online joint VNF chain composition and embedding for 5G networks," in *Proc. IEEE GLOBECOM*, Dec. 2018, pp. 1–6.

[12] Y. Liu, C. Yuen, X. Cao, N. U. Hassan, and J. Chen, "Design of a scalable hybrid MAC protocol for heterogeneous M2M networks," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 99–111, 2014.

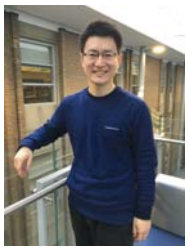
[13] E. Soltanmohammadi, K. Ghavami, and M. Naraghi Pour, "A survey of traffic issues in machine-to-machine communications over LTE," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 865–884, 2016.

[14] J. Chen, S. Yan, Q. Ye, W. Quan, P. T. Do, W. Zhuang, X. Shen, X. Li, and J. Rao, "An SDN-based transmission protocol with in-path packet caching and retransmission," in *Proc. IEEE ICC*, May 2019, pp. 1–6.

[15] B. Davie and J. Gross, "A stateless transport tunneling protocol for network virtualization (STT)," *Internet Engineering Task Force*, April 2016, [Online]. Available: https://tools.ietf.org/html/draft-davie-stt-08.



Jiayin Chen received the B.E. degree and the M.S. degree in the School of Electronics and Information Engineering from Harbin Institute of Technology, Harbin, China, in 2014 and 2016, respectively. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Her research interests are in the area of vehicular networks and machine learning, with current focus on Intelligent Transport System and big data.



Qiang Ye (S'16-M'17) received his Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2016. He is currently a Research Associate with the Department of Electrical and Computer Engineering, University of Waterloo, where he had been a Post-Doctoral Fellow from Dec. 2016 to Nov. 2018. His current research interests include AI and machine learning for future networking, IoT, SDN and NFV, network slicing for 5G networks, VNF chain embedding and end-to-end performance analysis.

Wei Quan received the Ph.D. degree in communication and information system from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2014. He is currently an Associate Professor with the School of Electronic and Information Engineering, Beijing Jiaotong University (BJTU). He has published more than 20 papers in prestigious international journals and conferences including IEEE Communications Magazine, IEEEWIRELESS COMMUNICATIONS, IEEE NETWORK, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE COMMUNICATIONS LETTERS, IFIP Networking, IEEE ICC, and IEEE GLOBECOM. His research interests include key technologies for network analytics, future Internet, 5G networks, and vehicular networks. He serves as an Associate Editor for the Journal of Internet Technology (JIT), Peer-to-Peer Networking and Applications (PPNA), and IET Networks, and as a technical reviewer for many important international journals. He is also a Member of ACM and a Senior Member of the Chinese Association of Artificial Intelligence (CAAI).

Si Yan received his B.S. degree in Electronic Information Engineering and M.E. degree in Electrical and Computer Engineering from Tianjin University and University of Calgary, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Waterloo. His current research interests include fifth-generation networks, software-defined networking, network function virtualization, and network protocol design. He is a Student Member of the IEEE.

Phu Thinh Do received the B.S. degree in electrical engineering from the Ho Chi Minh City University of Technology in 2010 and the M.S.E and Ph.D. degrees from the Department of Electronics and Radio Engineering, Kyung Hee University, Korea, in 2016. From 2016 to 2019, he was first a Postdoctoral Fellow at Digital Communication Lab, Kyung Hee University then at Broadband Communications Research Lab, University of Waterloo, Canada. He's now a Data Scientist at Sendo.vn. His current research interests include 5G wireless communications, search engines and recommendation systems for e-commerce platforms.

Weihua Zhuang (M'93–SM'01–F'08) has been with the Department of Electrical and Computer Engineering, University of Waterloo, Canada, since 1993, where she is currently a Professor and a Tier I Canada Research Chair of wireless communication networks. She was a recipient of the 2017 Technical Recognition Award from the IEEE Communications Society Ad Hoc and Sensor Networks Technical Committee, one of 2017 ten N2Women (Stars in Computer Networking and Communications), and a co-recipient of several best paper awards from IEEE conferences. She was the Technical Program Symposia Chair of the IEEE GLOBECOM 2011 and the Technical Program Chair/Co-Chair of the IEEE VTC in 2016 and 2017. She is a Fellow of the Royal Society of Canada, the Canadian Academy of Engineering, and the Engineering Institute of Canada. She is an Elected Member of the Board of Governors and VP Publications of the IEEE Vehicular Technology Society. She was an IEEE Communications Society Distinguished Lecturer from 2008 to 2011. She was the Editor-in-Chief of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2007 to 2013.

Xuemin (Sherman) Shen (M'97-SM'02-F'09) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on resource management in interconnected wireless/wired networks, wireless network security, social networks, smart grid, and vehicular ad hoc and sensor networks. He is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.

Dr. Shen received the R.A. Fessenden Award in 2019 from IEEE, Canada, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015 and the Education Award in 2017 from the IEEE Communications Society. He has also received the Excellent Graduate Supervision Award in 2006 and the Outstanding Performance Award 5 times from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, the IEEE Globecom'07, the Symposia Chair for the IEEE ICC'10, the Tutorial Chair for the IEEE VTC'11 Spring, the Chair for the IEEE Communications Society Technical Committee on Wireless Communications, and P2P Communications and Networking. He is the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL and the Vice President on Publications of the IEEE Communications Society.

Xu Li is a staff researcher at Huawei Technologies Inc., Canada. He received a Ph.D. (2008) degree from Carleton University, an M.Sc. (2005) degree from the University of Ottawa, and a B.Sc. (1998) degree from Jilin University, China, all in computer science. Prior to joining Huawei, he worked as a research scientist (with tenure) at Inria, France. His current research interests are focused in 5G system design and standardization, along with 90+ refereed scientific publications, 40+ 3GPP standard proposals and 50+ patents and patent filings. He is/was on the editorial boards of the IEEE Communications Magazine, the IEEE Transactions on Parallel and Distributed Systems, among others. He was a TPC co-chair of IEEE VTC 2017 (fall) – LTE, 5G and Wireless Networks Track, IEEE Globecom 2013 – Ad Hoc and Sensor Networking Symposium. He was a recipient of NSERC PDF awards, IEEE ICNC 2015 best paper award, and a number of other awards.

Jaya Rao received his B.S. and M.S. degrees in Electrical Engineering from the University of Buffalo, New York, in 2001 and 2004, respectively, and his Ph.D. degree from the University of Calgary, Canada, in 2014. He is currently a Senior Research Engineer at Huawei Technologies Canada, Ottawa. Since joining Huawei in 2014, he has worked on research and design of CIoT, URLLC and V2X based solutions in 5G New Radio. He has contributed for Huawei at 3GPP RAN WG2, RAN WG3, and SA2 meetings on topics related to URLLC, network slicing, mobility management, and session management. From 2004 to 2010, he was a Research Engineer at Motorola Inc. He was a recipient of the Best Paper Award at IEEE WCNC 2014.