RPC: An Approach for Reducing Compulsory Misses in Packet Processing Cache

Hayato YAMAKI†, Hiroaki NISHI†, Shinobu MIWA†, and Hiroki HONDA†, Members

SUMMARY We propose a technique to reduce compulsory misses of packet processing cache (PPC), which largely affects both throughput and energy of core routers. Rather than prefetching data, our technique called response prediction cache (RPC) speculatively stores predicted data in PPC without additional access to the low-throughput and power-consuming memory (i.e., TCAM). RPC predicts the data related to a response flow at the arrival of the corresponding request flow, based on the request-response model of internet communications. Our experimental results with 11 real-network traces show that RPC can reduce the PPC miss rate by 13.4% in upstream and 47.6% in downstream on average when we suppose three-layer PPC. Moreover, we extend RPC to adaptive RPC (A-RPC) that selects the use of RPC in each direction within a core router for further improvement in PPC misses. Finally, we show that A-RPC can achieve 1.38x table-lookup throughput with 74% energy consumption per packet, when compared to conventional PPC.

key words: internet router, packet processing cache, data prediction

1. Introduction

The demand for high-throughput and low-power core routers has increased with the rapidly growing internet traffic. The Ministry of Internal Affairs and Communications, Japan, reports that Japanese network equipment will face 190x internet traffic of 2006 and result in 10x power consumption in 2025 [1], [2]. This trend in internet traffic growth is not limited to Japan [3]–[5], and it drives network engineers to develop new techniques of high-throughput and low-power packet processing in core routers.

Intelligent table lookup is a key to efficient packet processing in core routers. Packet processing requires search operations on several tables such as a routing table, which is usually configured with a ternary content addressable memory (TCAM). TCAM has the capability to search stored data quickly; however, it does not achieve the throughput required for next-generation networks such as a 400-Gbps network. To make matters worse, TCAM consumes a remarkable amount of energy (16x SRAM energy [6]) that accounts for about 40% of the overall router power [7]. Considerable effort has been invested to improve the throughput and power-efficiency of TCAM [7], [8]; however, no technique has achieved both 400-Gbps and SRAM-like power, which are required for future core routers.

A promising approach to high-throughput and low-power packet processing is to reduce the number of TCAM accesses. Packet processing cache (PPC), which is a high-throughput and low-power memory that retains the results of TCAM lookups, can be used for this purpose [9]–[14]. Different from content caches used in content delivery networks (CDNs) [15], [16], PPC keeps the information needed for delivering packets in core routers (e.g., routing information). Internet traffic has the temporal locality (i.e., many packets with a similar header arrive at a core router in a short period), so that many packets frequently search identical data from TCAM [9]. This fact means that there are many opportunities to reuse the results of TCAM lookups. The core routers that employ PPC can reduce the number of TCAM accesses by searching PPC instead of TCAM.

Previous studies have reported that PPC can successfully reduce a large number of TCAM accesses; however, even with the state of the art technologies that reduce capacity and conflict misses in PPC [10], [17], many packets still need to access TCAM. Figure 1 shows the breakdown of cache miss in PPC with various numbers of entries (from $2^{10}$ to $2^{18}$ entries). As shown in this figure, about 10% of packets cause compulsory miss in PPC even if PPC has the entries enough to minimize capacity and conflict misses (i.e., $2^{17}$–$2^{18}$ entries).

As a way to reduce the compulsory miss of PPC, we proposed response prediction cache (RPC) [19]. RPC speculatively stores data in PPC rather than prefetching data from TCAM. RPC predicts the data related to a response
flow at the arrival of the corresponding request flow, based on the request-response model of internet communications. Since RPC requires no additional access to TCAM at the prediction, it can remove TCAM accesses if a packet hits the predicted data within PPC. The paper [19] showed that RPC can reduce cache miss to 15.3% on average for a PPC with a specific configuration (i.e., 32KB).

This paper is an extended version of the work presented in [19]. In this extended version, we explore the impact of the proposed technique on PPC exposed to various area constraints. In addition, we reviewed all descriptions of the original paper and then rewrote many sentences clearly with some additional data.

The main contributions of this paper are listed as follows.

- We developed a new technique to reduce compulsory misses in PPC, which have not been reduced by the previous work.
- We built a simulation environment of our RPC, and by using this environment, we showed that RPC can reduce PPC misses by 13.4% in upstream and 47.6% in downstream on average for real network traces.
- We showed that our technique is implementable with ASIC. We also showed that A-RPC has an area overhead of 0.91%, an increase of 38% in throughput, and an energy saving of 26%, compared to conventional PPC.

The rest of this paper is organized as follows. We first give a more detailed description of PPC in Sect. 2. Section 3 introduces our key concept, data prediction for response flows, and Sect. 4 describes RPC and A-RPC in detail. We present our experimental results in Sect. 5, and finally conclude this paper in Sect. 6.

2. Packet Processing Cache

We provide the overview of the PPC architecture and then introduce the existing techniques that reduce PPC misses.

2.1 Architecture

A packet that arrives in a core router needs to search the data required for packet transfer from some tables (e.g., a routing table, an ARP (address resolution protocol) table, an ACL (access control list), and a QoS (quality of service) table), which are typically configured with TCAM. Several fields of the packet header are used for this search operation. More specifically, the five-tuple of the header (i.e., source and destination IP addresses, source and destination port numbers, and protocol number), which is defined as a flow, is often used in IPv4 networks. Therefore, TCAM produces the same output for multiple packets that have an identical flow.

PPC utilizes the above nature of packet processing to reduce the number of TCAM accesses. Figure 2 shows the overview of PPC. PPC basically focuses on IPv4-based routing. Typical PPC is indexed with CRC-hash values calculated from five-tuples. A cache tag consists of 104-bit flow information (i.e., a five-tuple). A data field contains the results of multiple table lookups. The typical size of the data field is 15 bytes. More specifically, the data field includes a 1-byte output interface number in the routing table, a 12-byte destination MAC address in the ARP table, a 1-byte filtering result in ACL, and a 1-byte priority number in the QoS table.

PPC can be extended to store the results of some header modifications and packet inspection if the results depend on five-tuples. PPC can substitute multiple operations executed by a core router in IPv4 networks for one cache access. Note that recent backbone networks often build non-IP-based networks, such as MPLS and VXLAN, on IPv4 networks. PPC cannot provide the functions defined by these non-IP-based routing schemes. However, it is considered that PPC can be applied in some parts of these networks because they require IPv4 routing for some purposes (e.g., MPLS label distribution and routing in VXLAN tunnels).

The throughput and energy of core routers with PPC highly depend on the PPC hit rate, because PPC, which is configured with SRAM, can serve a larger number of data with lower energy than TCAM. Many efforts to reduce PPC misses have been carried out; however, the miss rate of the state-of-the-art PPC is still high (about 10%) [17].

2.2 Related Work

One approach to improving PPC is to reduce capacity misses. Some tag-compression techniques have been used to increase the number of cache entries under a given area budget. Chang et al. propose Digest Cache, in which a 32-bit hash value compressed from an original five-tuple is used as a cache tag [11]. One problem with Digest Cache is the hardware cost required to prevent hash conflicts. Ata et al. propose Tuned Cache that uses the limited flow information (source/destination IP addresses and lower port numbers) as a cache tag [12], but Tuned Cache is unable to conduct fine-grained packet processing such as firewall, because of the lack of the entire flow information.

For reducing conflict misses, some intelligent replacement algorithms have been proposed [10], [13], [14].
Yamaki et al. pointed out that the least recently used (LRU) algorithm, which is used in typical caches, is not sufficient in PPC because LRU keeps mice flow entries in PPC for a while, which are never accessed by the other packets [10]. To quickly evict mice flow entries from PPC, they proposed a novel replacement algorithm called elevator cache (ELC) and showed that ELC could reduce the number of PPC misses. They also proposed a technique to filter flows composed of a single packet (e.g., domain name system (DNS) and network attack packets) from PPC, as an alternative [20].

Tanaka et al. proposed to hierarchize PPC to reduce the overall PPC miss and provided comprehensive evaluations in multi-layer PPC with various configurations [17]. They showed that the combination of 8KB L1 cache, 128KB L2 cache, and 4MB L3 cache could reduce PPC miss rate from 17.2% to 4.5% on average. As a result, it can achieve 1Tbps packet processing with 0.72x power consumption compared to the conventional (i.e., 32KB single-layer) PPC.

These methods successfully reduced the number of PPC misses. However, PPC still shows the miss rate of 10% for several types of networks, and most of the PPC misses are compulsory misses. For this, a new technique to reduce compulsory misses in PPC is needed.

3. Data Prediction for Response Flows

Many internet applications communicate between servers and clients based on a request–response model; a client sends a request to a server, and the server then replies to the client. This feature involves two types of compulsory misses in PPC used in routers on a routing path. One is a miss caused by a request packet that is first sent from a client to a server, and the other is a miss caused by a response packet that is first sent from the server to the client.

Meanwhile, the five-tuple of a response packet can be predicted from the corresponding request packet. The source and destination IP addresses of the response packet can be computed by swapping those of the request packet. Likewise, the source and destination port numbers of the response packet can be computed with those of the request packet. Moreover, the protocol numbers of the two packets are always the same value.

The results of the table lookups by the response packet can also be predicted from the request packet. The result of the routing table lookup by the response packet depends on which interface the request packet comes from. The result of the ARP table lookup by the response packet is equivalent to the source MAC address of the request packet. Furthermore, the results of the ACL and QoS table lookups by the response packet are usually the same as those by the request packet. A few exceptional situations may occur, but they can be detected by adding an asymmetry flag to each table entry. The asymmetry flag is set to an entry at the first write if the table contents of the request and response are asymmetrical.

These insights bring a new opportunity in optimizing PPC; a core router speculatively stores the data for an upcoming first response packet in PPC at the arrival of the corresponding first request packet. The key idea is that the complete data for the response packet can be created from the request packet without additional TCAM accesses in most cases. Therefore, the TCAM lookups can be removed from response-packet processing if the response packet arrives timely at the router.

Since the effectiveness of our approach depends on how much response flows are routed to the same router as the corresponding request flows, we first investigated the percentage of symmetrically-routed flows in various network traffic. Figure 3 shows the traffic breakdown for eleven network traces listed in Table 2. Note that we defined downstream as the direction which included a larger number of the response flows regardless of the inbound or outbound of the network because the traces do not include the information for inbound or outbound (in many cases, downstream is equal to inbound). Figure 3 (a) shows that approximately 70% of all flows in real networks are symmetrically routed. This indicates that prediction for response flows has potential in reducing 35% of compulsory misses. In addition, as shown in Fig. 3 (b), symmetrically-routed response flows account for almost half of the flows in downstream traffic. Thus, it is expected that our approach will reduce a large number of compulsory misses, especially in downstream traffic.
4. Response Prediction Cache

We first present RPC, an architecture for predicting the PPC data for response flows, and then introduce Adaptive-RPC (A-RPC), which is a technique for further improvement in RPC. Finally, we discuss the limitation of RPC.

4.1 Architecture

Figure 4 shows the overview of RPC, assuming a simple router that has two line cards (LC1 and LC2). This example represents the situation that a request packet from a network arrives at LC1 and is forwarded from LC2 to another network. SRAM on each line card is used as a packet buffer, while TCAM is used as a table memory. PPC and an RPC module are implemented on a packet forwarding ASIC in each line card.

The data prediction for the first response packet is conducted as follows: (a) First, the packet processing engine (PPE) in LC1 identifies a first request packet by checking whether an arrived packet misses PPC. The PPE sends the packet to LC2 for the delivery in either case, while in case of a PPC miss, the PPE sends the results of the table lookups and a PPC miss flag to LC2, where the corresponding response packet will arrive. (b) Second, the RPC module in LC2 computes the tag and data fields for the corresponding response packet using the information received from LC1 if the received cache miss flag is enabled. The computed data is stored in the queue in LC2. (c) Finally, the data is written to PPC in order when a write port of PPC is free.

The router has a time lag between sending the first response packet to the network and writing the predicted data to PPC in LC2; however, this time lag is relatively small when compared to the typical server response time. Therefore, RPC can finish writing predicted data to PPC in most cases until the first response packets arrive at the router.

Figure 5 shows the detailed implementation of PPE, PPC, and RPC. A packet forwarded to a PPE is analyzed in the parser, and the five-tuple is then extracted from the packet. The extracted five-tuple is sent to the hit/miss checker and the hash module, while the whole packet data is stored in SRAM. The hit/miss checker judges the PPC miss or hit of the packet by accessing to PPC through the PPC controller. At this time, the hash value and the five-tuple is used as the index and tag, respectively. If a cache hit occurs, the packet is dealt with the data retrieved from PPC. On the other hand, in case of a cache miss, the packet is forwarded to the TCAM controller in order to lookup TCAM. After the TCAM lookups, the packet is sent to an output LC, and the cache is updated with the result of the TCAM lookups.

The RPC module consists of three main components: an aligner, a hash module, and a queue. The aligner creates the tag and data fields for a first response packet by using the data of the corresponding request packet received from the other LC. The hash module calculates the cache index of PPC from the five-tuple of the predicted response packet. The predicted data are stored in the queue and dequeued by the arbiter in the PPC controller.

4.2 Adaptive Approach

RPC can be applied to both communication directions in a core router (i.e., upstream and downstream), but the effectiveness depends on how much symmetrically-routed response flows are within a direction, as shown in Fig. 3. Conventionally, since there is bias in the rate of request or response flows depending on the direction, applying RPC to downstream allows more opportunities for reducing compulsory misses caused by response packets. Meanwhile, RPC may increase PPC misses for upstream because the data created by the packets that cause conflict and capacity misses pollute PPC.

From this reason, we proposed adaptive RPC (A-RPC) to select the use of RPC in each direction within a core router. Following the normal boot process of the router, each RPC modules do simple training in a short period called a judgment phase, and then enables or disables RPC for the direction based on the training result. The router configuration determined in a judgment phase does not change under
operation because many servers and clients keep their topological location for a long time.

Figure 6 shows the processing flow of the training when PPC has 256 cache lines. PPC is escorted by A-RPC modules before and behind. The A-RPC modules are activated only during the judgment phases and turned off during the normal operation for the power-saving purposes.

In the judgment phase, PPC is split into two areas and RPC is applied to only one area. A hash value is also divided by two in the modifier module to access two areas. This operation is simply done by 1bit right shift and bit manipulation for the most significant bit. A packet that arrives at a line card accesses both areas to simulate the impact of RPC on PPC misses. More specifically, A-RPC counts the number of PPC misses in each area using two miss counters. A-RPC decides to use RPC if the miss counter in the area that uses RPC shows a smaller value than that in the other area.

4.3 Discussion

4.3.1 Attack Tolerance

When network attacks such as a port scan attack and a denial of service (DoS) attack occur, RPC may increase PPC misses because it updates PPC with the data for response packets never arrived (i.e., typical servers do not respond to attack packets). However, this problem can be mitigated by attack-aware cache [20]. Attack-aware cache identifies attack flows by monitoring specific port numbers and the number of flows generated by each source IP address. RPC can prevent the insertion of useless data into PPC by using attack-aware cache before data prediction.

4.3.2 Asymmetric Routing

Some packets in real networks are routed by asymmetric routing that delivers requests and responses on different paths. Asymmetric routing causes two problems in RPC: routing response packets to wrong directions at a branch router that asymmetrically routes packets, and creating unused PPC entries at the routers between the branch router and a server. These problems can be alleviated by setting an asymmetry flag for each entry in PPC. If a response flow hits PPC and the asymmetry flag in the retrieved entry is set, RPC discards the routing information denoted by the entry and then searches the routing table. Asymmetry flags can be automatically set in branch routers, while they can be set in the other routers by receiving asymmetric signals from branch routers. By adding the asymmetry flag to the predicted PPC entries, RPC can be applied in any routers without inconsistent packet routing. However, it makes the RPC effect small, and thus, it is better not to use the asymmetric flag if it is possible. The following discussions comprehensively show the cases that RPC can be applied without asymmetric flags under asymmetric situations.

(1) AS-path level: when a border router connects more than two ASs, asymmetric routing may occur. However, the asymmetric flag is not required of the predicted PPC entries for downstream because these entries do not cause inconsistent routing. Thus, if a large number of packets are returned to the router with symmetric paths, it is expected that RPC performs effective.

(2) Router level: When routers adopt offloading techniques, such as LAG (link aggregation) and ECMP (equal cost multi path), asymmetric routing may occur. However, RPC can be applied to such routers without the asymmetric flag in the following two cases. The first case is that these techniques are operated as per-flow packet allocation. In this case, symmetric paths are promised at the flow level. The second case is that these techniques are realized by unifying physical interfaces as a single logical interface. Packets are automatically allocated to each physical interface by specifying the logical interface as the output interface of predicted PPC entries.

(3) IGP level: If a router may include asymmetric paths at IGP level, the asymmetric flag is required to apply RPC. However, the asymmetric routing is not always permitted in networks, especially in enterprise networks. They are often designed to deny asymmetric routing considering the network security and management. For example, firewalls check bidirectional paths of flows to operate "stateful inspection" and ensure the traceability of flows. For this reason, most router vendors also provide functions for denying asymmetric routing as uRPF (unicast reverse path forwarding). In such networks and the routers turned on uRPF, RPC can be applied without the asymmetric flag.

5. Evaluation

We first describe our experimental methodology. Next, we show our experimental results. In this paper, we evaluated our technique using various metrics: PPC miss rate, area cost, throughput, and energy.

5.1 Experimental Methodology

The RTL simulator bundled in Synopsys Design Compiler was used for the experiment in area cost, while our in-house PPC simulator was used for the other metrics. Our simulator feeds the packets from a network trace at the appropriate
cycle designated by timestamps, and it imitates the architecture shown in Fig. 5 and emulates the cycle-level behavior of PPC such as reading packets, extracting the flow information, creating the cache index, and referring to PPC and TCAM. We implemented both RPC and A-RPC on this simulator.

In practice, we need to design high-performance PPC while keeping the area overhead small, but the minimum area overhead acceptable to packet forwarding ASIC depends on use cases of core routers. Therefore, in this paper, we impose various constraints on the area of PPCs. Table 1 shows details of the PPC configurations we evaluate. SMALL is supposed in many previous studies and configured as single-layer PPC. As shown in Table 1, SMALL remains a large number of capacity misses and conflict misses. LARGE is configured as three layers and shown in the paper [17] as the configuration to achieve 1Tbps with the lowest power. LARGE can reduce most of capacity misses and conflict misses; however, it additionally requires 68% of the packet buffer area. Moreover, we considered MEDIUM as the medium-level constraint between SMALL and LARGE. MEDIUM is configured as two layers and achieves the PPC miss rate comparable to LARGE. The area overhead of MEDIUM is 9.5% of the packet buffer area. In any case, a large number of compulsory misses are still remained and degrade PPC performance.

Each PPC is constructed as an inclusive cache with LRU (least recently used) replacement policy. To process packets without packet loss, we assumed that a buffer that retains packets waiting for TCAM access has a sufficient number of entries.

As workloads, we used various types of real-network traces shown in Table 2. These traces were captured from networks in universities, laboratories, and Internet consortiums. More specifically, each trace was captured in a border router of the network and separately contains upstream and downstream packets running on a link between the router and the upstream AS. However, the network topologies of most traces are not specified. For this, the simulation is based on the following two assumptions. (1) AS-path level asymmetry is not included in the traces because networks in such locations generally connect just one transit peer. (2) Router level asymmetry is not considered because it does not become a problem for RPC in many cases, as mentioned in Sect. 4.3.2. Consequently, we ignored the influence of asymmetric routing in the simulation.

5.2 Experimental Results

5.2.1 PPC Miss Rate

Figure 7 shows the PPC miss improvement by RPC when compared to conventional PPC. The traces are displayed in the order from the higher bandwidth to the lower bandwidth. In addition, we present a detailed analysis of the PPC misses. Figures 8 and 9 show the breakdown of PPC hit rate. Each figure separately represents the results per traffic direction.

As shown in the figures, RPC can reduce the number of cache misses for many types of network traces (by up to 75.7% in MEM). In particular, RPC can greatly improve the PPC miss rate in the case of LARGE (by 13.4% in upstream and 47.6% in downstream on average). This is because LARGE has the PPC capacity enough to retain predicted data and many first response packets can therefore hit them without evicting the other useful entries.

In the case of LARGE, the PPC miss improvement almost corresponded to the percentages of the symmetrically-routed response flows shown in Fig. 3 (b). This result reinforces the suggestion in Sect. 3 that the effect of RPC depends on the percentages of symmetrically-routed flows in networks. This is why RPC showed different impacts on PPC miss rate in directions. In the case of MEDIUM and LARGE, RPC much more improved the PPC miss rate of the downstream in many network traces when compared to that of the upstream. However, contrary to our expectation, several network traces showed the opposite results, especially in the case of SMALL. This is because second and

| Abbreviation | Source | Captured point | Packets per sec. (upstream) | Packets per sec. (downstream) |
|--------------|--------|----------------|-----------------------------|-----------------------------|
| Core         | An academic network (not open to the public) | 268,147 | 369,578 |
| UPCB         | Catalan research network (Internet consortium) | 55,965 | 58,110 |
| UFL          | University of Florida | 48,911 | 51,181 |
| FRG          | Front Range Gigapop (Internet consortium) | 30,503 | 38,777 |
| PSC          | Pittsburgh Supercomputing Center | 26,897 | 19,393 |
| MRA          | Merit network (Internet consortium) | 16,223 | 23,175 |
| BWY          | Columbia University | 17,959 | 20,377 |
| WIDE         | WIDE network (Internet consortium) | 9,614 | 15,022 |
| TXG          | Texas University | 11,465 | 12,332 |
| ODU          | Old Dominion University | 6,979 | 6,718 |
| MEM          | University of Memphis | 1,178 | 992 |

Table 1 Details of PPC configurations. Area ratio means the percentage of PPC area in packet buffer area.

| Capacity [byte] | SMALL | MEDIUM | LARGE |
|-----------------|-------|--------|-------|
| 32K              | 16K/256K | 8K/128K | 4M     |
| 4/ - / -         | 4/4 / - | 2/4/4 |
| 0.75%            | 9.5%   | 68%    |
| Miss rate (total) |       |       |       |
| (Compulsory)     | 26.2%  | 13.0%  | 10.1%  |
| (Capacity)       | 10.0%  | 10.0%  | 10.0%  |
| (Conflict)       | 5.32%  | 1.26%  | 0.13%  |
| (Capacity)       | 10.9%  | 1.77%  | 0%     |
subsequent response packets frequently miss PPC due to the small capacity and RPC has a possibility to predictively create PPC entries for these packets. It is considered that the network configuration and internet access patterns are important factors to decide the RPC effect. However, it is difficult to reveal it only by using the information at a trace level and captured-point level. Therefore, for making further discussion of the RPC effect and its dependency on network characteristics, it is required to use additional information of networks more than network traces.

5.2.2 Miss Improvement by A-RPC

Next, we evaluated the sensitivities of packet count used for training A-RPC (denoted as $N$). $N$ was varied in the range of 100 to 100,000. We assumed that the continuous $N$ packets that are randomly picked from each network trace come into a line card during the judgment phase, thereby removing the impact of traffic localities. Figure 10 shows the average PPC miss rates of A-RPC in the case of SMALL with 20 judgments for each trace. We also show the PPC miss rates of RPC and an ideal case for reference. The figure indicates that A-RPC with $N = 10,000+$ can decide the use of RPC well and achieve nearly equivalent performance to the ideal.

Table 3 shows the summary of average PPC miss rates of various PPC capacities and configurations. A-RPC can improve the PPC miss rate by 4.88% in the case of SMALL and 3.93% in the case of MEDIUM when compared to simple RPC. However, if the PPC capacity is large enough, A-RPC cannot improve the PPC miss rate. This is because RPC does not impact on the PPC miss rate in this case, and applying RPC is always effective. From these results, A-RPC can be trained with a small number of packets, and we can implement a miss counter by a 14+ bit register.
5.2.3 Impact of Attacks

We evaluated the attack tolerance of RPC in cooperation with attack-aware cache, using a custom workload including attacks. This workload was based on PSC traffic, and seven attacks captured from other traces were mixed into PSC traffic every 10 seconds. Figure 11 shows the PPC miss rates of conventional PPC, RPC, and RPC with attack-aware cache. The figure shows only the result of MEDIUM.

As shown in the figure, RPC shows the increase in PPC misses for the upstream due to the inefficient prediction (up to 1.84x larger PPC misses when compared to conventional PPC). In contrast, RPC with attack-aware cache shows small increase in cache misses for the upstream (up to 1.04x larger misses when compared to conventional PPC). This result indicates that attack-aware cache is effective in preventing inefficient predictions caused by attacks.

5.2.4 Hardware Cost

To assess the hardware cost of PPC including A-RPC, we implemented our A-RPC excluding TCAM, with Verilog-HDL and 45-nm Free PDK OSU Library [22]. The area of the combination logic in PPC was computed by logical synthesis with Synopsys Design Compiler O-2018.06-SP4. The areas of some memory components, namely the cache memory, the state memory for LRU, and the buffer, were estimated by using CACTI 6.5 [23].

Table 4 shows the area per module in the PPC system. We estimated PPC area as SMALL, and thus the PPC area becomes larger in the case of MEDIUM and LARGE. The table indicates that the area of the PPC memory is dominant in the PPC system. Note that the PPC controller accounts for 5% of all area because it includes the replacement mechanism of PPC. RPC and A-RPC modules are relatively small (0.73% and 0.18% of the whole area, respectively). According to [24], the area of a recent TCAM package used in routers is 729 mm² and is considerably larger than PPC. Thus, our A-RPC can reduce the load of TCAM with small hardware costs.

5.2.5 Throughput and Energy

Finally, we estimated the throughput and energy consumption of the table lookups. The throughput and energy models of PPC systems were considered in the papers such as [13], [17]. We constructed the models based on [17] because it improved the models of [13] and targeted multi-level PPCs.

The throughput of the table lookups with PPC, represented as $T_{ppc}$, can be calculated as:

$$T_{ppc} = \frac{l}{d_{ppc}} = \min \left\{ \frac{l}{d_{l1}}, \frac{l}{d_{l2}} \cdot \frac{m_{l1}}{m_{l2}}, \frac{l}{d_{tcam}} \cdot \frac{m_{l1}}{m_{l2}} \right\}$$  (1)

Eq. (1) shows the throughput model of two-layer PPC. It easily extends to single-layer or three-layer PPC. $d_{l1}$, $d_{l2}$, and $d_{tcam}$ represent the latencies of L1 PPC, L2 PPC, and TCAM, respectively; $m_{l1}$ and $m_{l2}$ represent the PPC miss rates of L1 PPC and L2 PPC, respectively. The latencies of PPCs were estimated with CACTI 6.5, whereas the latency of TCAM was computed with a TCAM power and timing model [6]. $l$ represents the packet length. In this paper, we
assumed that $l$ is 64 bytes as the shortest packet length because the worst-case throughput is traditionally presented as the router throughput.

Meanwhile, the energy consumption of the table lookups per packet with PPC, represented as $E_{ppc}$, can be calculated as:

$$E_{ppc} = (E_{ppc}^{dyn} + E_{ppc}^{dyncam} \cdot m_1 + E_{ppc}^{dyncam} \cdot m_2 \cdot m_3 \cdot n) + (P_{stat}^{l1} + P_{stat}^{l2} + P_{stat}^{tcam}) \cdot d_{ppc}$$

Here, $E_{ppc}^{dyn}$, $E_{ppc}^{dyncam}$, and $E_{ppc}^{dyncam}$ represent the dynamic energy per access of L1 PPC, L2 PPC, and TCAM, respectively; $P_{stat}^{l1}$, $P_{stat}^{l2}$, and $P_{stat}^{tcam}$ represent the static power of L1 PPC, L2 PPC, and TCAM, respectively. Same as the latencies, the energy and power were estimated with CACTI 6.5 and a TCAM power and timing model [6]. $n$ represents the number of TCAM accesses needed to process a packet. In this paper, we assumed that a router had the four tables shown in Fig. 2 (i.e., $n = 4$).

Table 5 shows our estimation results. In the case of LARGE, A-RPC can achieve 1.38x table-lookup throughput with 74% energy per packet, compared to the conventional PPC. When compared to the TCAM-only system, the proposed architecture showed 29.5x higher table lookup throughput with 4.6% energy consumption. The cache miss reduction of RPC directly impacts on both the throughput and energy consumption because TCAM accesses are still a bottleneck of table lookups and a major cause of increasing the power consumption of a router.

### 6. Conclusions

This paper presented a novel technique called RPC to reduce compulsory misses in PPC. Our experimental results showed that RPC could reduce the number of cache misses of large-capacity PPC by 13.4% in upstream and 47.6% in downstream on average. In addition, we extended RPC to A-RPC that selectively uses RPC for the further improvement in PPC misses. Our A-RPC could effectively select the use of RPC by using 10,000 or more packets for training. Finally, we showed that A-RPC could achieve 1.38x table-lookup throughput with 74% energy consumption per packet when compared to conventional PPC.

### Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP18K18022.

### References

[1] The Ministry in Japan, “Tabulation and estimation of internet traffic in Japan.” http://www.soumu.go.jp/main_content/000604136.pdf. (Accessed on 1/20/2020).

[2] METI, “Green it initiative in Japan.” http://www.meti.go.jp/english/policy/GreenITInitiativeInJapan.pdf. (Accessed on 1/20/2020).

[3] Cisco Systems, Inc., “Cisco visual networking index: Forecast and trends, 2017-2022 white paper.” http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-net-working-index-vni/white-paper-c11-741490.html. (Accessed on 1/20/2020).

[4] J. Fan, C. Hu, K. He, J. Jiang, and B. Liuy, “Reducing power of traffic manager in routers via dynamic on-off-chip scheduling,” Proc. 31st Annual IEEE International Conf. Computer Communications (IEEE INFOCOM 2012), pp.1925–1933, March 2012.

[5] X. Zheng and X. Wang, “Comparative study of power consumption of a netfpga-based forwarding node in publish-subscribe internet routing,” Computer Communications, vol.44, pp.36–43, 2014.

[6] B. Agrawal and T. Sherwood, “Ternary cam power and delay model: Extensions and uses,” IEEE Trans. Very Large Scale Integration (VLSI) Systems, vol.16, no.5, pp.554–564, May 2008.

[7] M. Nawa, K. Okuda, S. Ata, Y. Kuroda, Y. Yano, H. Iwamoto, K. Inoue, and I. Oka, “Energy-efficient high-speed search engine using a multi-dimensional tcam architecture with parallel pipelined subdivided structure,” Proc. 13th IEEE Annual Consumer Communications Networking Conference (CCNC 2013), pp.309–314, Jan. 2016.

[8] S. Gamage and A. Pasqual, “High performance parallel packet classification architecture with popular rule caching,” Proc. 18th IEEE International Conf. Networks (iCON 2012), pp.52–57, 2012.

[9] G. Chandramohan and G. Ramaswamy, “Improving performance of digest caches in network processors,” Proc. 15th International Conf. High performance computing (HiPC 08), vol.5374, pp.6–17, 2008.

[10] H. Yamaki and H. Nishi, “Line replacement algorithm for 11-scale packet processing cache,” Adjunct Proc. 13th International Conf. Mobile and Ubiquitous Systems: Computing Networking and Services (MOBIQUITOUS2016), pp.12–17, 2016.

[11] Y.-K. Chang and F.-C. Kuo, “Packet processing with blocking for bursty traffic on multi-thread network processor,” Proc. IEEE International Conf. High Performance Switching and Routing (HPSR 2009), pp.1–6, June 2009.

[12] S. Ata, M. Murata, and H. Miyahara, “Efficient cache structures of ip routers to provide policy-based services,” Proc. IEEE International
YAMAKI et al.: RPC: AN APPROACH FOR REDUCING COMPULSORY MISSES IN PACKET PROCESSING CACHE

Conf. Communications 2001 (ICC 2001), pp.1561–1565, June 2001.

[13] G. Liao, H. Yu, and L. Bhuyan, “A new ip lookup cache for high performance ip routers,” Proc. 47th Design Automation Conference (DAC 2010), pp.338–343, 2010.

[14] C. Kim, M. Caesar, A. Gerbear, and J. Rexford, “Revisiting route caching: The world should be flat,” Proc. 10th International Conf. Passive and Active Network Measurement (PAM ’09), pp.3–12, 2009.

[15] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl, “Globally distributed content delivery,” IEEE Internet Computing, vol.6, no.5, pp.50–58, Sept. 2002.

[16] E. Nygren, R.K. Sitaraman, and J. Sun, “The akamai network: A platform for high-performance internet applications,” SIGOPS Oper. Syst. Rev., vol.44, no.3, pp.2–19, Aug. 2010.

[17] K. Tanaka, H. Yamaki, S. Miwa, and H. Honda, “Multi-level packet processing caches,” 2019 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS), pp.1–3, April 2019.

[18] WIDE MAWI WorkingGroup, “Mawi working group traffic archive.” http://mawi.wide.ad.jp/mawi/. (Accessed on 1/20/2020).

[19] H. Yamaki, H. Nishi, S. Miwa, and H. Honda, “Data prediction for response flows in packet processing cache,” Proc. 55th Annual Design Automation Conference (DAC 2018), pp.1–6, ACM, 2018.

[20] H. Yamaki and H. Nishi, “An improved cache mechanism for a cache-based network processor,” Proc. 2012 International Conf. Parallel and Distributed Processing Techniques and Applications (PDPTA 2012), pp.428–434, 2012.

[21] RIPE Network Coordination Centre, “Ripe ncc internet measurement data repository.” https://labs.ripe.net/datarepository. (Accessed on 1/20/2020).

[22] J. Stine, I. Castellanos, M. Wood, J. Henson, F. Love, W.R. Davis, P.D. Franzon, M. Bucher, S. Basavrajaija, J. Oh, and R. Jenkal, “Freepdk: An open-source variation-aware design kit,” Proc. IEEE International Conf. Microelectronic Systems Education (MSE ’07), pp.173–174, 2007.

[23] N. Muralimanohar, R. Balasubramonian, and N. Jouppi, “Optimizing numa organizations and wiring alternatives for large caches with cacti 6.0,” Proc. 40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007), pp.3–14, 2007.

[24] RENESAS, “20 mbit quad-search content addressable mem.” http://www.renesas.com/ja-JP/doc/products/memory/r10cp00-01eu0100,team.pdf. (Accessed on 1/20/2020).

Hayato Yamaki received the Ph.D. in Engineering from Keio University, Japan, in 2016. He is now an assistant professor at the University of Electro-Communications. His research interests are internet architecture and computer architecture. He is a member of ACM, IEEE, and IEICE.

Hiroki Honda received the B.S., M.S., and Ph.D. degrees in electrical engineering from Waseda University, Japan, in 1984, 1986 and 1991, respectively. He is currently a professor at The University of Electro-Communications, Japan. He was the chair of IEEE Computer Society Japan Chapter from 2009 to 2011. His research interests include parallel processing, parallelizing compiler, parallel computer architecture. He is a member of IEEE, IEEE-CS and ACM.

Hiroki Nishi received the B.E., M.E., and Ph.D. degrees from Keio University, Japan, in 1994, 1996, and 1999, respectively. He is currently a Professor of Keio University. The main theme of his current research is in the building of the total network system including integrating hardware and software architecture. In particular, service-oriented router and smart city/community are the centers of his research topics.

Shinobu Miwa received the Ph.D. degree in Informatics from Kyoto University in 2007. He joined Lawrence Livermore National Laboratory as a Visiting Scientists and Professionals, and is now an Associate Professor at the University of Electro-Communications. His research interests are computer architecture and high performance computing. He is a member of ACM and IEEE.