Real-Time Dynamic Decomposition Storage of Routing Tables

Wenlong Chen[1], Lijing Lan[1], Xiaolan Tang[1(✉)], Shuo Zhang[1], and Guangwu Hu[2]

[1] College of Information Engineering, Capital Normal University, Beijing 100048, China
tangxl@cnu.edu.cn

[2] Graduate School at Shenzhen, Tsinghua University, Shenzhen, China

Abstract. The decomposition storage of routing tables can effectively alleviate the storage problem caused by the Internet routing expansion. However, the existing researches based on certain routing tables cannot apply to unknown ones or support dynamic changes of routing tables. This paper presents a real-time dynamic decomposition storage model, named RDDS, which takes the IP prefix with 8 bits length mask as a pocket prefix and distributes the routing entries (REs) to different line cards (LCs) according to different pocket prefixes. In the light of the mapping relationship between the destination IP addresses of packets and the pocket prefixes, the forwarding engine determines the host LC of one IP packet and complete forwarding in the LC. RDDS needs a mapping table to locate the host LC in the process of packets forwarding, which only introduces very little logical processing. Therefore, RDDS hardly affect the overall performance of packets forwarding in routers. Experimental results show that RDDS achieves real-time part-storage and the load balancing of REs in LCs. Considering the frequent update of routing tables, RDDS is more significant than other storage models in real environments.

Keywords: Routing tables · Real-time · Dynamic · Decomposition storage · Pocket prefix · Mapping table

1 Introduction

With the rapid expansion of Internet, the kernel routing table of the Internet keeps fast growing and becomes a big challenge for backbone routers. At present, high performance routers use Ternary Content Addressable Memory (TCAM) to store routing tables. As opposed to Static Random Access Memory (SRAM), the storage space of TCAM is small, and the price is expensive. Hence, these undoubtedly increase the burden of storage. The current decomposition storage methods aim at routing tables with specific capacity. However, in the realistic environment, the routing table capacity in routers usually is unknown and it always change dynamically.

S. Wang and A. Zhou (Eds.): CollaborateCom 2016, LNICST 201, pp. 353–362, 2017.
DOI: 10.1007/978-3-319-59288-6_32

This paper presents a real-time dynamic decomposition storage (RDDS) model to store routing tables in distributed routers. REs load-balanced means that total REs is allocated to each LC averagely, which prompts LCs to complete the routing and the packets forwarding together. In order to achieve the dynamic load balancing, we choose the first 8 bits of destination IP prefixes as distribution units. And according to the different 8 bits prefix, we decompose and store the routing table. Each LC stores only a partial routing table, and with the increase in the number of LCs, the number of dynamic REs in each LC will continue to decrease. At the same time, we use the first 8 bits prefix as an index value to form a mapping relationship between LCs and index values. For an IP packet to be forwarded, the forwarding engine locates the LC where the RE for the packet is stored, and finds the related information based on longest prefix matching (LPM). RDDS is more suitable for the dynamic changes of routing tables. In addition, the first 8 bits of destination IP prefix of each REs are unique, so the redundant storage phenomenon does not appear in each LC.

Compared to previous researches, RDDS has several obvious advantages. (1) The unknown routing tables can be dynamically decomposed and stored, hence, it has a strong random distribution capacity. (2) RDDS can locate the positions of LCs directly according to the index values. (3) As the value of the first 8 bits is unique, each LC does not store REs repeatedly.

2 Related Work

In recent years, an increasing number of researches strive to optimize the decomposition storage of routing tables and routing lookup engines.

The decomposition storage of routing tables can be expedited by various approaches. Lin *et al.* [4] introduce a logical caching mechanism where they expected to divide the routing tree into multiple sub-trees with identical sizes. Many other studies [5,6] develop more decomposition storage algorithms benefiting from a new feature of IP prefix bits. The key idea is to split the routing table into multiple sub-tables. For example, the Decomposed Storage of FIB (DSF) algorithm [5], Enhanced DSF (EDSF) [5], and the Speedy Packet Lookups (SPAL) algorithm [6].

There are also several efforts for performing other performance optimization in routers. Li *et al.* [7] use an alternative next hop Forwarding Information Base (FIB) aggregation algorithm to reduce the forwarding table sizes, which ensures packets can be forwarded to the destination IP address through any paths not the best path. Fast lookup and update of routing tables with a hash table is achieved according to the short prefix and the long prefix in [8]. The articles [10–12] are committed to improving routing tables lookup efficiency instead of the decomposition storage of routing tables.

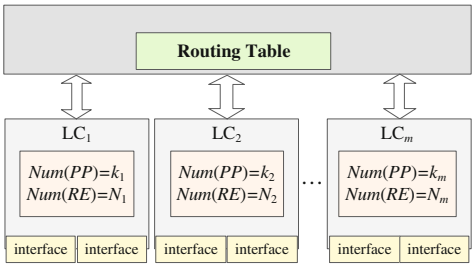


Fig. 1. Dynamic decomposition storage

3 Real-Time Dynamic Decomposition Storage Model

3.1 RDDS Overview

The full backup storage makes the load of routers increase as REs continue to increase [1–3,9]. And the existing decomposition storage methods do not apply to the dynamical update of routing tables. RDDS decomposes and stores REs to LCs on the basis of the non-full backup storage and ensures that the decomposition storage is dynamical and balanced. Our distribution principles are: (1) the REs attributed to the same PP are stored in the same LC. (2) each LC owns multiple PPs. (3) the RE capacity of each LC are as same as possible. As shown in Fig. 1, each LC just stores a portion of the routing table.

Definition 1. Equilibrium Degree (ED): ED is used to describe the relationship between the actual value of each LC's routing storage capacity and the average storage capacity of each LC. ED is calculated by (1):

$$ED = \frac{|RE\_LC_m - RE_{ave}|}{RE_{ave}} \tag{1}$$

RE_LC_m is the actual routing capacity of LC_m in RDDS, RE_{ave} is the average capacity of each LC. This paper focuses on the real-time ED of RE's storage capacity, which ensures that ED is minimum as much as possible at any time. When the value of ED is 0, it indicates that all LCs are absolute load-balanced.

Definition 2. Reduction Ratio (RR): RR is the ratio that describes the relationship between RE's capacity of a LC in RDDS and the capacity in Full Backup Storage. We obtain the value of RR by (2):

$$RR = \frac{RE}{RE_{full}} \tag{2}$$

Definition 3. Pocket Prefix (PP): Define the first n bits prefix of the routing address mask as a PP, the paper sets n as 8.

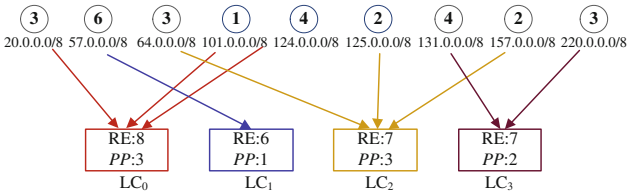


Fig. 2. Decomposition example in 4 LCs

In the dynamic decomposition storage of routing tables, there are 223 PPs (the multicast address is not taken into account) and each PP is independent. For example, the PP of 198.0.0.0/8 is 198.

We expect that in Fig. 1 RDDS ensures that N_1, N_2, ... , N_m are almost identical and the value of ED is the minimal. $Num_m[i]$ represents the RE number of i_{th} PP of m_{th} LC.

Equation (3) is the total number of REs stored in LC_m :

$$N_m = \sum_{i=1}^{k_m} Num_m[i] \tag{3}$$

3.2 Rules of Dynamic Decomposition Storage

RDDS dynamically decomposes and stores the routing table to achieve the load balancing of each LC. The decomposition storage ideas are as follows:

1. For each RE, we choose the first 8 bits of destination IP prefix as a PP and compare it with PPs of each LC. If there is the same PP_m, we store this RE to the LC that PP_m belongs to. Otherwise we use the first 8 bits of the mask prefix as a new PP, and assign the PP in the light of following steps.
2. For each allocation, according to the RE storage capacity of each LC, we preferentially assign the PP to the LC whose storage capacity is the smallest.
3. If there are multiple LCs with same minimum RE capacity, we preferentially assign the PP to the LC whose number of PP is the smallest.
4. If the minimum RE storage capacity and PPs of multiple LC are same, we assign the PP according to the serial number of LCs from small to large.
5. The new RE is stored in the LC owning the new PP.

As shown in Fig. 2, we choose 4 LCs as an example. The number in each circle is the number of REs that the corresponding PP contains.

In the process of assign PPs to LCs, We define the mapping relationship between PP and LC as $MR_{pre}(PRE_{sign}, LC_m)$. PRE_{sign} is the prefix sign of the RE, and LC_m represents the LC that the RE of PRE_{sign} belongs to. We use an array to save this relationship. So the relationship between PRE_{sign} and LC_m is expressed as: $MR_{pre}[PRE_{sign} - 1] = m$.

Table 1. Fourteen dynamic arrival REs

No.	RE	No.	RE
①	120.0.0.0/8	②	120.90.224.0/19
③	64.128.0.0/9	④	112.0.0.0/8
⑤	101.9.129.0/24	⑥	56.196.0.0/16
⑦	21.101.144.0/20	⑧	64.152.0.0/14
⑨	220.213.0.0/16	⑩	120.233.0.0/17
⑪	156.128.0.0/9	⑫	124.186.156.0/20
⑬	120.32.0.0/11	⑭	220.52.0.0/14

Table 2. The process of dynamic decomposition storage

Step	LC_0	LC_1	LC_2	LC_3	ED
(1)	A: ①				1.500
(2)	A: ① ②				1.500
(3)	A: ① ②	B:③			0.750
(4)	A: ① ②	B:③	C:④		0.500
(5)	A: ① ②	B:③	C:④	D:⑤	0.300
(6)	A: ① ②	B:③ E:⑥	C:④	D:⑤	0.335
(7)	A: ① ②	B:③ E:⑥	C:④ F:⑦	D:⑤	0.215
(8)	A: ① ②	B:③ ⑧ E:⑥	C:④ F:⑦	D:⑤	0.250
(9)	A: ① ②	B:③ ⑧ E:⑥	C:④ F:⑦	D:⑤ G:⑨	0.167
(10)	A: ① ② ⑩	B:③ ⑧ E:⑥	C:④ F:⑦	D:⑤ G:⑨	0.200
(11)	A: ① ② ⑩	B:③ ⑧ E:⑥	C:④ F:⑦ H:⑪	D:⑤ G:⑨	0.137
(12)	A: ① ② ⑩	B:③ ⑧ E:⑥	C:④ F:⑦ H:⑪	D:⑤ G:⑨ I:⑫	0
(13)	A: ① ② ⑩ ⑬	B:③ ⑧ E:⑥	C:④ F:⑦ H:⑪	D:⑤ G:⑨ I:⑬	0.116
(14)	A: ① ② ⑩ ⑬	B:③ ⑧ E:⑥	C:④ F:⑦ H:⑪	D:⑤ G:⑨ ⑭ I:⑬	0.143
	PP: 1 RE: 4	PP: 2 RE: 3	PP: 3 RE: 3	PP: 3 RE: 4	

We randomly select 14 dynamic REs and store them in 4 LCs. As shown in Table 1, the sequence number of REs is the order of REs arriving.

According to the definition of PP, we know that 14 REs belong to 9 different PPs respectively: A: 120, B: 64, C: 112, D: 101, E: 56, F: 21, G: 220, H: 156, I: 124. Initially, the storage capacity of the routing table and each LC are both 0. When ① arrives, we store it in LC_0. When ② arrives, we also store it in LC_0, because PP of ② is the same to ①. For ③ ④ ⑤, according to the different PP, we store them in LC_1, LC_2 and LC_3 by order. So far, the RE storage capacity of LC_0, LC_1, LC_2, LC_3 are 2, 1, 1, 1, respectively. For ⑥, we should choose the LC whose RE capacity is the smallest. There are three smallest RE capacities

and they are all 1. Besides, the PP capacity in each LC is also all 1. According to the serial number of LCs, we preferentially store ⑥ in LC_1 and ⑦ in LC_2. Similarly, we distribute 14 REs to 4 LCs. Finally, the RE capacity of 4 LCs is: LC_0: ① ② ⑩ ⑬, LC_1: ③ ⑥ ⑧, LC_2: ④ ⑦ ⑪ and LC_3: ⑤ ⑨ ⑫ ⑭. After the dynamic decomposition storage of REs, we expect to achieve the same REs storage capacity for each LC and the minimum ED value. In this example, the PP capacities in each LC are different. But the storage capacities of REs in different LCs are almost same, namely: 4, 3, 3, 4. As shown in Table 2, we know that the result of routing storage is very balanced at any time.

In the implementation process, we first initialize the corresponding parameters. In this algorithm, RE_i indicates the dynamic RE; PP_j ($1 \leq j \leq 255$) represents a pocket prefix; LCm ($0 \leq m \leq n\text{-}1$) represents any LCs; PP[m]($0 \leq m \leq n\text{-}1$) represents an array that stores PPs of one LC; RE[m]$0 \leq m \leq n\text{-}1$ represents an array that stores REs of one LC; $MR_{pre}[PRE_{sign}\text{-}1]$ represents an array that stores the serial number of LC that PRE_{sign} belongs to, PRE_{sign} is the prefix sign of the RE. The implementation algorithm is shown in Algorithm 1.

Algorithm 1. Distribute Dynamic RE.

Input:
RE_i;
Output:
PP[m];
RE[m];
Initialize PP[m] and RE[m] are 0;
Let LC_0 be the line card with the minimum amount of REs;
Set min=0;
Get the first 8 bits prefix of RE_i: j;
if $MR_{pre}[j\text{-}1]$==-1 **then**
  //The PP of RE_i does not exist.
  Send RE_i to the LC_{min};
  $MR_{pre}[j\text{-}1]=min$;
  PP[min]++;
  RE[min]++;
  Calculate the amount of REs of every LC;
  Get the serial number of LC with the minimum amount of REs: i;
  Set min=i;
else
  //The PP of RE_i already exists in one LC.
  m=$MR_{pre}[j\text{-}1]$;
  Send RE_i to the LC_m;
  RE[m]++;
end if
Return RE[m], PP[m];

3.3 Packet Forwarding Process

As for the packet forwarding, we define 3 different LC roles. The LC who has the RE that matches the destination IP address of the packet is called host LC (LC_{host}). At the same time, the LC that receives the packet is called its receiving LC (LC_{in}). The LC that sends out the packet is called its sending LC (LC_{out}). For each packet, we get the next hop through LPM in its LC_{host}. Therefore, a router receives an IP packet, it is necessary to find its LC_{host} according to the mapping table. Then, the packet is sent to the LC_{host} for LPM. When a LC receives a packet, the lookup process has two cases depending in the different LC_{in}. (1) LC_{in} is LC_{host}: When finding that LC_{in} of the packet is its LC_{host}, we get the forwarding information through LPM and forward this packet. (2) LC_{in} is not LC_{host}: We get its LC_{host}, where this packet will be sent to. Then we can get the out-interface and the next hop through LPM, and forward this packet. For both of the above cases, if the out-interface information is in LC_{host}, the packet is forwarded directly. Otherwise, it needs to be forwarded to the LC_{out} to complete the final transfer task.

During the packet forwarding of RDDS, some packets need once or twice inter-card communication. However, kernel routers commonly use TCAM to complete LPM, and the lookup speed of TCAM can reach nanosecond level. Therefore, it does not affect the network service quality. As for the internal switching bandwidth, due to the bottleneck of the processing speed of distributed routers is LCs rather than the inner switch network, we can solve this problem by improving the acceleration ratio of switch fabrics.

3.4 Routing Information Update

The RE adding is based on the allocating principles in Sect. 3.2. According to the mapping relationship between PRE_{sign} and LC, we know whether the PP of the new RE exists in LCs. If exists, the new RE will be stored in the LC that the PP belongs to; if not exists, the PP of this new RE will be served as a new PP. For the RE deletion: (1) If the PP that the deleted RE belongs to contains only this one RE, we delete both the RE and the PP. Furthermore, the mapping table will also be updated; (2) If the PP that the deleted RE belongs to contains multiple REs, we just delete this RE.

4 Performance Evaluation and Experimental Analysis

According to expression (1), we know that the value of ED is almost tiny at any time in RDDS. It indicates that RDDS achieves load balancing of LCs in dynamic changes. When n LCs store m REs in a complete balance, the desired capacity of REs required in each LC is m/n. In RDDS, the capacity of REs that each LC needs to store is shown in expression (3). Furthermore, the additional storage space that RDDS needs is also tiny. Each LC just needs additional 446 bytes to store a mapping table.

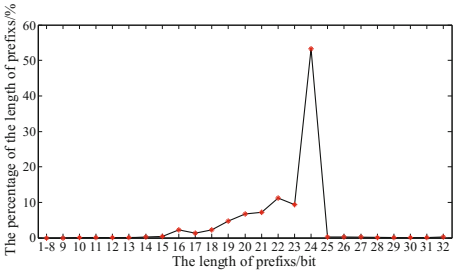
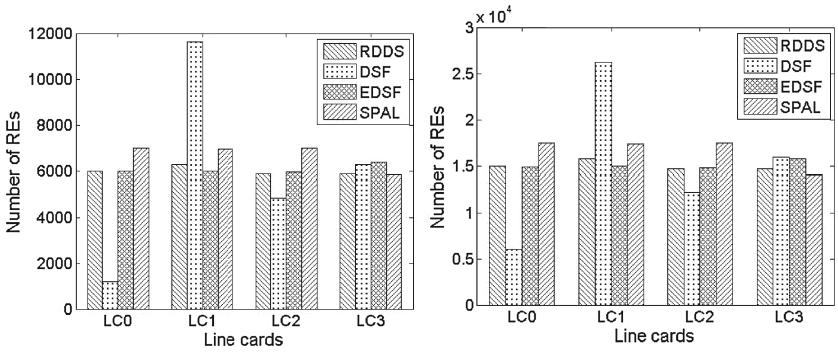


Fig. 3. The distribution of the length of prefixes



(a) Decomposition storage of (b) Decomposition storage of

Fig. 4. REs decomposition storage in different models

In addition to the theoretical analysis, we have run a series of experiments to measure the RDDS's performance. This experimental data are selected from 9 different ASs [1,9]. In order to prove the credibility of RDDS, we make a statistical analysis of the prefix mask length of all REs in 9 different ASs, as shown in Fig. 3. According to the statistics on the percentage of each prefix mask length, we find that the prefix mask length of all REs less than 8 bits does not exist, so using the first 8 bits of the IP prefix of REs as a PP is feasible.

This experiment uses 5 PCs. PC_1 is the master card and $PC_2 \sim PC_5$ is $LC_0 \sim LC_3$, respectively. We inject all routing information about $data_1$ [9] and $data_2$ [9] to PC_1, and compare RDDS with other three models, namely DSF [5], EDSF [5], SPAL [6].

From Fig. 4(a) and (b), we know that EDSF achieves the load balancing of each LC, but it causes some consumption during decomposition and storage space. Although the RE capacity of each LC is almost identical in SPAL, it introduces uncontrolled redundancy. Compared with these three models, RDDS not only achieves a better load balancing of LC, but also does not introduce any redundant REs storage in each LC.

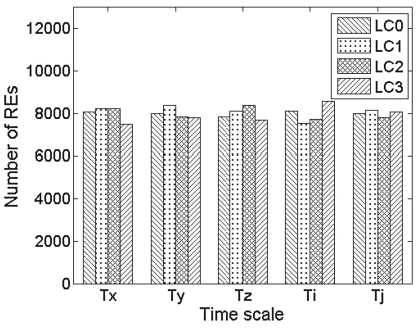


Fig. 5. REs capacity at random times

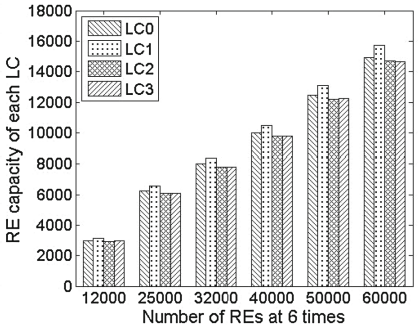


Fig. 6. REs capacity comparison

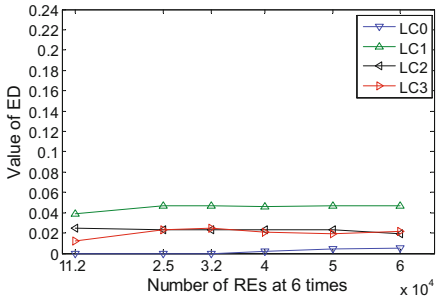


Fig. 7. 6 groups of ED values

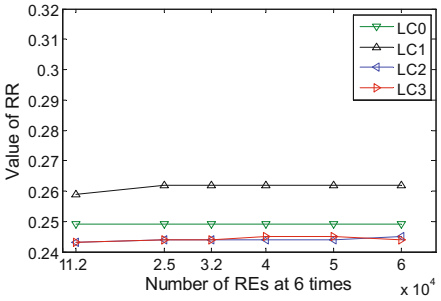


Fig. 8. REs compression ratio in each LC

We have an experiment about the decomposition storage of routing tables in dynamic random changes, which is intended to prove that RDDS has a better randomness. As shown in Fig. 5, the number of REs in 4 LCs are most balanced at any time, and the maximum absolute difference of the REs capacity in each LC is no more than 900.

In order to better explain the characteristics of RDDS, we carry out 6 experiments with the different number of REs. Form Fig. 6, we find that the absolute difference between REs capacities in each LC and average REs capacities is no more than 600. Hence, it strongly proves that RDDS has a wonderful and stable performance.

As shown in Fig. 7, we know that RDDS effectively ensures that the storage capacity of REs in LCs at any moment are almost the same. When the value of ED is 0, RDDS achieves the absolute equilibrium storage of every LC. With the number of REs increasing, the ED of the REs capacity in each LC keeps the steady state. Figure 8 has a visual description of the compression ratio. Real experiments have indicated that the values of RR are uniformly distributed in the near of 0.25 (In the extremely balanced situation, the RR of routing tables in 4 LCs is 0.25.). Moreover, the minimum absolute difference between RR of RDDS and 0.25 is 0.

5 Conclusion

This paper proposes the RDDS that achieves real-time dynamic decomposition storage of REs in the distributed routers. It regards the first 8 bits of IP prefix as a pocket prefix. When REs arrive, they are stored on different LCs according to their pocket prefixes. And, RDDS ensures that the value of ED is always the minimum. Furthermore, compared with the full backup storage, the REs number of each LC is reduced sharply in RDDS. Experimental results show that RDDS not only ensures the non-full backup storage, but also is adaptable to the dynamic update of routing tables. Besides, RDDS does not produce redundant storage in LCs.

Acknowledgment. This work is supported by the National Natural Science Foundation of China under Grant No. 61373161, 61502320, 61300171. Science & Technology Project of Beijing Municipal Commission of Education under Grant No. KM201410028015, the Youth Backbone of Beijing Outstanding Talent Training Project under Grant No. 2014000020124G133, and Guangdong Natural Science Foundation No. 2015A030310492.

References

1. BGP Routing Table Analysis Reports. http://bgp.potaroo.net
2. Carpenter, B., Crowcroft, J.: IPv4 Address Behaviour Today. RFC 2101, February 1997
3. Gerich, E.: Guidelines for Management of Ip Address Space. RFC 1466, May (1993)
4. Lin, D., Zhang, Y., Hu, C., Liu, B., Zhang, X., Pao, D.: Route table partitioning and load balancing for parallel searching with TCAMs. In: IEEE International Parallel and Distributed Processing Symposium, Long Beach, CA, USA, pp. 1–10, March 2007
5. Chen, W., Xu, M., Yang, Y., Han, D.: Decomposed storage model of FIB for cluster router. Chin. J. Comput. **34**(9), 1611–1620 (2011)
6. Tzeng, N.-F.: Routing table partitioning for speedy packet lookups in high-performance distributed routers. IEEE Trans. Parallel Distrib. Syst. **17**(5), 481–494 (2006)
7. Li, Q., Wang, D., Xu, M., Yang, J.: On the scalability of router forwarding tables: Nexthop-selectable FIB aggregation. In: IEEE INFOCOM, Shanghai, China, vol. 42(4), pp. 321–325 (2011)
8. Zhang, X., Perrig, A., Zhang, H.: Centaur: A hybrid approach for reliable policy-based routing. In: IEEE 33rd International Conference on Distributed Computing Systems, Montreal, Quebec, Canada, pp. 76–84, June 2009
9. BGP Routing Table. http://www.cidr-report.org/as2.0/
10. Yang, T., Duan, R., Lu, J., Zhang, S., Dai, H., Liu, B.: CLUE: Achieving fast update over compressed table for parallel lookup with reduced dynamic redundancy. In: IEEE 32nd International Conference Distributed Computing Systems. Macau, China, pp. 678–687, June 2012
11. Li, Q., Xu, M., Chen, M.: NSFIB construction & aggregation with next hop of strict partial order. In: IEEE INFOCOM, Turin, Italy, pp. 550–554, April 2013
12. Chen, W., Liu, Y., Wang, H.: On storage partitioning of internet routing tables: A P2P-based enhancement for scalable routers. Peer-to-Peer Netw. Appl. **8**(6), 952–964 (2015)