Multipath Fault-Tolerance Routing Mechanism in Data Center Network

Nan Ya
College of Computer Science and Engineering
Northeastern University
Shenyang, China
Email: yenna_neu@163.com

Xingwei Wang
College of Software
Northeastern University
Shenyang, China
Email: wangxw@mail.neu.edu.cn

Shuang Zhang
College of Software
Northeastern University
Shenyang, China
Email: zhangs@swc.neu.edu.cn

Min Huang
College of Information Science and Engineering
Northeastern University
Shenyang, China
Email: mhuang@mail.neu.edu.cn

***Abstract*—This paper proposes a Fault-Tolerance Effect and Cost Function based Multipath Routing Mechanism (FEAC) in the Data Center Network (DCN). The path value is used to represent the quality of the path, measured by bandwidth, delay, packet loss rate and other information. We design a fault-tolerance effect and cost function, which is used to calculate optimal path numbers. The effect is expressed by the probability of selecting multiple paths to successfully transmit a flow. The fault-tolerance cost includes link congestion and elephant flow replication cost. Heuristic thoughts are adopted to design the algorithm of generating feasible path set. After the final path set is obtained, the data flow is copied and transmitted on it to improve the reliability of the network. The proposed algorithm is simulated on two network topologies; and it has a better performance than benchmark algorithms according to a variety of evaluation criteria.**

Keywords- Data Center Network; fault tolerance; intelligence optimization

I. INTRODUCTION

The new Data Center Network generally supports multipath, which makes the network have better reliability and greater bandwidth. Compared with single-path routing mechanism, multipath routing mechanism has obvious advantages in transmission reliability, load balancing, fault tolerance and recovery. The current data center can usually reach the scale of hundreds of thousands of servers. Because of its large scale and low cost equipment in the DCN, various faults may occur, which can roughly be divided into host faults, switch faults, link faults, etc. When a fault occurs, it not only consumes a lot of material resources and manpower, but also causes a large number of data packets to be discarded, which reduces network reliability.

The current research on the architecture can be generally divided into network-centric solutions and server-centric solutions. According to [1], the traffic of DCN is mainly divided into work-seek-bandwidth model and scatter-gather model. Network traffic is composed of a large part of the rat flows and a small part of the elephant flows [2], which means that the centralized strategies cannot be applied well. There is a wide variety of research on routing mechanisms. In [3], an energy-efficient QoS multicast routing mechanism is proposed, considering the static traffic demand of IP layer and optical layer. The bio-inspired solution has been studied to solve the routing optimization problem recently [4]. On the basis of load balancing, Renuga Kanagaelu et al. use NOX components to collect link failure information to solve the fault-tolerance problem [5]. Changlin Jiang et al. put forward a fault-tolerance routing protocol to detect and correct the malfunction link of mis-wiring problem in Clos DCN [6]. Ramos, R. M. et al. proposed a fault-tolerance routing algorithm based on the coding path, which encodes the calculated routing information into the data packet header for routing, and switches the path when it fails [7].

Most of the existing routing mechanisms in the DCN use a single path to transmit data, and mainly use centralized control methods, which easily lead to large overhead and link congestion. A multipath fault-tolerance routing mechanism based on fault-tolerance effect and cost function is depicted in this paper. The algorithm uses prediction to find a path to improve the network reliability, to reduce latency and to adapt to the characters that the traffic in the DCN are mainly delay-sensitive short flows.

II. NETWORK MODEL

The network model is an undirected graph $G=(V,E)$, as shown in Fig. 1. Node set $V=H \cup S$, includes host set $H=\{h_1,h_2,..,h_n\}$ and switch set $S=\{s_1,s_2,..,s_n\}$, edge set is $E=\{e_{ij},i \in S, j \in H \cup S\}$.

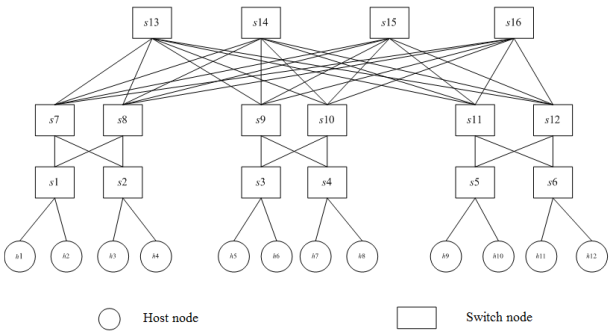


Figure 1. Simple DCN topology

978-1-5386-7445-1/18/$31.00 ©2018 IEEE
DOI 10.1109/DCABES.2018.00070

$Node=\{totalbuf_i, reamainbuf_i, port_i\}$ is the node model. $Link=\{bandwidth_{ij}, c_{ij}, delay_{ij}, lossrate_{ij}\}$ is the link model, $bandwidth_{ij}$ represents the maximum bandwidth, c_{ij} represents the available bandwidth. The routing request for flow F_k is defined as a five-element tuple $R_k=\{s_k,a_k,b_k,d_k,l_k\}$, the elements are the source address, the destination address, the minimum bandwidth, the maximum delay and the maximum loss rate respectively. The optimization goal is to maximize *WF*, which is the difference between the fault-tolerance effect and the cost function in the whole network, the mathematical model is described as follows.

$$maximize \ \ WF \tag{1}$$

s.t.

$$\forall e_{ij} \in E, \forall k, c_{ij} \geq b_k \tag{2}$$

$$\forall e_{ij} \in E, \forall k, 0 \leq delay_{ij} \leq d_k \tag{3}$$

$$\forall e_{ij} \in E, \forall k, 0 \leq lossrate_{ij} \leq l_k \tag{4}$$

$$\forall k, \sum_{i \in V} r_k(s_k, i) = \sum_{i \in V} r_k(i, a_k) \tag{5}$$

$$\forall k, \forall i, j \in V \setminus \{s_k, a_k\}, \sum_{i \in V} r_k(i, j) = \sum_{j \in V} r_k(j, i) \tag{6}$$

where (1) is the optimization objective of the fault-tolerance algorithm, and (2) to (4) denote the bandwidth, delay and loss rate constraints on the link respectively. Equation (5) indicates that the number of data flows sent by nodes is equal to the number of data flows that the destination node receives. Equation (6) indicates that the number of incoming data flows on any intermediate link is equal to the number of outbound data flows, ensuring that data packets belonging to the same flow take the same link.

III. ROUTING MECHANISM

The fault-tolerance problem mainly depends on the selection of multiple paths to be transmitted simultaneously; thus how to select the path is one of the keys. The algorithm should sort the paths when determining the number of paths, and use path value to represent the quality of the path. The specific calculation formula is as follows.

$$pathvalue_k = \alpha * f(bu_k) + (1-\alpha) * g(delay_k, lossrate_k) \tag{7}$$

$$f(bu_k) = (1 - bu_k) * (1 + bu\_vardif_k) \tag{8}$$

$$bu\_variance_k = \frac{\sum_{i=1}^{n} (bu_i - bu\_mean)^2}{n} \tag{9}$$

$$g(delay_k, lossrate_k) = \frac{1}{|\lg(delay_k * lossrate_k)|} \tag{10}$$

where (7) is the calculation of the path value, α is the weight coefficient, which is set to 0.6 by multiple experiments, bu_k is the maximum bandwidth utilization rate of all the links on the path. Equation (8) is to calculate the bandwidth factor, and *bu_vardif* is expressed by the difference of the bandwidth utilization variance before and after the flow request is received via this path. Equation (9) is the link bandwidth utilization variance, and the average bandwidth utilization is *bu_mean*.

A. Fault-Tolerance Function and Cost Function Design

In the paper, the fault-tolerance effect is represented by the probability of transmitting a flow via k paths successfully. Link congestion cost is represented by the probability of congestion in the k paths. The data flow replication cost takes only the copy cost of elephant flows into account. The trade-off function is shown as follows.

$$f_k = pb_{success}^k - cost_k \tag{11}$$

$$pb_{success}^k = 1 - \prod_{i=1}^{k} [1 - (1 - pb)^{n_i}] \tag{12}$$

$$cost_k^1 = \frac{m}{\sum_{i=1}^{k} n_i} \tag{13}$$

$$cost_k^2 = (\frac{elephantfs}{maxfs})^{1/k} \tag{14}$$

$$m = \sum_{s=1}^{k} \sum_{e_{ij} \in P_s} m_{ij} \tag{15}$$

where (11) shows the trade-off function between the fault-tolerance effect function and cost function. Equation (12) assumes that the failure probabilities *pb* of all the links are the same, n_i indicates the number of links on path P_i. m in (13) indicates the number of links that are congested, the specific formula is shown in (15), where m_{ij} indicates whether the link e_{ij} is congested. Assuming that the threshold for the link to reach congestion is ϕ, if $(bandwidth_{ij} - c_{ij}) / bandwidth_{ij} > \phi$, m_{ij}=1; otherwise m_{ij}=0. In (14), *elephantfs* indicates the size of the elephant flow.

B. The Design of Heuristic Algorithm

The goal of this algorithm is to obtain a feasible solution that is close to the optimal solution within a certain period of time. The basic idea is described as below.

(1) Generate initial solution. First, a basic two-path set $\{path_1, path_2\}$ is randomly generated, and then a new path set $\{path_1, path_2\}$ is sorted in descending order by the path value. According to the link disjoint, the path set is updated to obtain the initial solution. The initial solution is added to the initial solution of the taboo *tabu*, and the next iteration is used to constrain the initial solution. The goal is to get a good initial solution every time, which is conducive to expanding to the optimal solution.

(2) Expand the initial solution. The initial solution gets an initial value $f_{init=2}$ by (11). Each time the node needs to select the next hop among all feasible link sets, it first selects the link whose bandwidth utilization is the minimum. If the utilization ratios are the same, the minimum link delay and loss rate are taken as criteria in order. Get the $path_i$ and add

it to the initial solution, calculate the current function value f_{curr} by (11). $f_{init=2} > f_{curr}$ indicates that the algorithm does not need to continue expanding. Otherwise, the algorithm continues expanding having the original solution replaced with extended solution.

(3) Obtain a feasible solution. Save the optimal function value and the corresponding path set, compare all optimal function values after reaching the iterations, and select the path set with the largest function value as the final feasible path set.

IV. PERFORMANCE EVALUATION

In our experiments, the 4-post topology [8] is simplified as 2 pods and 4 switch nodes, and each pod includes 2 host nodes and 8 switching nodes. The Hyper-Bcube topology [9] is simplified into two layers. There are 16 host nodes and 8 switch nodes. CMFR and NFR routing algorithms are chosen as the benchmarks. The following performance criteria are adopted.

A. Variance of Link Bandwidth Utilization

The variance of the link bandwidth utilization represents the discrete degree, and can reflect the load-balancing ability of the network directly. The formula is shown in (16), where $linkbu_{ij}$ indicates the bandwidth utilization on the link e_{ij}.

$$linkbu\_variance = \frac{\sum_{ij \in E} (linkbu_{ij} - linkbu\_mean)^2}{linknum} \tag{16}$$

B. Routing End-to-End Delay

For multipath fault-tolerance routing, the maximum path delay on multiple paths reflect the algorithm efficiency. To denote the minimum path delay from source to destination for all replication flows, only the transmission delay on the link is considered, as shown in (17).

$$delay\_ft = \min \sum_{ij \in path} delay_{ij} \tag{17}$$

C. Effect of Fault-Tolerance Routing

The effect of fault-tolerance routing indicates whether the data flow can continue transmitting without affecting the network in the event of a fault. It is mathematically transformed into the average number of data flows that can successfully route when there is a link failure in the network. As shown in (18), where $flownum\_success_i$ is the number of data flows that the algorithm runs the i_{th} successful routes, and $runnum$ is the number of times the algorithm runs, which is set as 10.

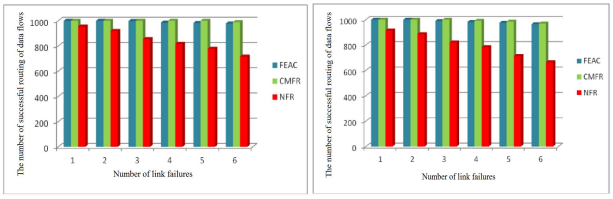
$$ft\_effect = \frac{\sum_{i=1}^{runnum} flownum\_success_i}{runnum} \tag{18}$$

D. Cost of Fault-Tolerance Routing

Suppose there is a fault link in the network, and the cost is mainly the copy cost of elephant flow in the DCN, as shown in (19), where $pathnum\_of\_elephantflow$ indicates the number of paths passed by each elephant flow request.

$$ft\_cost = \sum_{flownum} pathnum\_of\_elephantflow \tag{19}$$

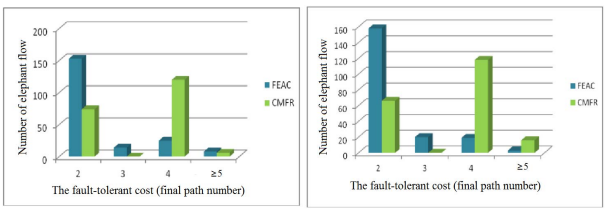
Fig. 2 shows the routing fault-tolerance effects of the three algorithms when there are 1000 data flow requests and the number of faulted links is from 1 to 6 in two topologies. It can be seen that with the increasing faulty links, the FEAC algorithm and the CMFR algorithm are basically stable at around 1000, and the successful routing number of NFR decreases rapidly with increasing faulty links .It is because the former two use the multipath feature while the latter simply selects an optimal path based on some benchmarks. When a failure occurs, the probability of route failure is very high. The number of successful data flow routing in the HyperBCube topology is smaller. With the same number of the failed links, the greater the total number of links in the network, the lower the probability that the algorithm chooses to have a faulty link in the path for the data flow, and the probability of multiple paths is lower than that of a single path. The number of successful data flows also increases.



(a) 4-post topology (b) HyperBCube topology

Figure 2. Performance comparison of fault-tolerance effect

Fig. 3 shows the fault-tolerance costs of the two algorithms when the data flow requests is 2,000. The FEAC algorithm not only considers the link congestion cost caused by network fault tolerant when calculating the number of paths, but also considers the cost caused by elephant flow replication. The CMFR algorithm does not distinguish between elephant and rat flows, so the FEAC algorithm's total elephant flow copy number is less than the CMFR algorithm's. The number of the final paths of the FEAC algorithm is basically around two, and the CMFR algorithm only considers the path delay when selecting the primary path and the secondary path, so the fault-tolerance cost of this algorithm in different topologies is slightly different.

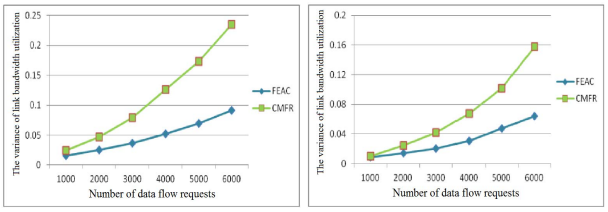


(a) 4-post topology (b) HyperBCube topology

Figure 3. Performance comparison of fault-tolerance costs

Fig. 4 shows the load-balancing degree of the link bandwidth utilization variances of the two algorithms when

the number of the data flow requests is 1000 to 6000 in two topologies. With the increase of the data flow requests, the bandwidth utilization variance of the CMFR algorithm increases faster than the FEAC algorithm, indicating that the FEAC algorithm is better and more stable than the CMFR algorithm in network load balancing. Because the former considers the fault tolerant while taking into account the load balance, the load balance factor is added when designing the path value. While the latter only relies on the minimum delay criterion when selecting a path, it does not consider link bandwidth information.



(a) 4-post topology (b) HyperBCube topology

Figure 4. Performance comparison of link bandwidth utilization variance

Finally, the efficiency of the two routing algorithms is compared by calculating the end-to-end delay distributions of the data flows with 2000 requests. Here, because the DCN has the characteristics of low delay, the network link delay is set to a random number of 0-5s. In the experiment, the end-to-end delay distribution interval is set as 5μs. Taking 4-post topology as an example, Fig. 5 shows that the CMFR algorithm is superior to the FEAC algorithm in terms of end-to-end delay. The CMFR algorithm selects the primary path based on the criterion of minimum path delay, and selects the secondary path for each node on the primary path. It is also based on the criterion that the path delay is the minimum, so for each flow request, the algorithm finds multiple paralleled paths for it, and the delay on the path is relatively small, which makes the end-to-end delay of the flow request smaller.

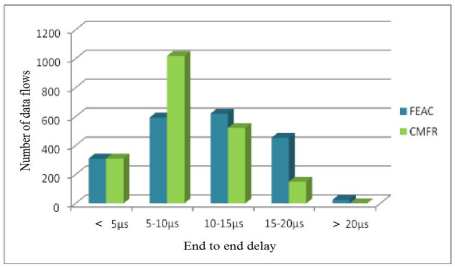


Figure 5. End to end delay distribution in 4-post topology

V. CONCLUSION

This paper makes full use of the characteristics of the new DCN topology with rich paths, by designing a multipath fault-tolerance routing mechanism, which has a good balance between the fault-tolerance effect and the fault-tolerance cost. Simulated on two topologies, the proposed algorithm is proved to have better fault tolerance and resource utilization. The future work is to optimize the algorithm and to design a more practical multipath mechanism in the DCN.

ACKNOWLEDGMENT

This work is supported by the Program for Liaoning Innovative Research Term in University under Grant No. LT2016007; the Major International(Regional) Joint Research Project of NSFC under Grant No.71620107003; the National Science Foundation for Distinguished Young Scholars of China under Grant No.71325002; the MoE and China Mobile Joint Research Fund under Grant No. MCM20160201; the National Natural Science Foundation of China under Grant No.61572123.

REFERENCES

[1] Theophilus Benson, Ashok Anand, Aditya Akella, Ming Zhang. Understanding data center traffic characteristics [A], WREN '09 Proceedings of the 1st ACM workshop on Research on enterprise networking [C], New York, 2009, 65-72.

[2] Thephilus Benson, Aditya Akella, David A. Maltz. Network Traffic Characteristics of Data Centers in the Wild [A], IMC '10 Proceedings of the 10th ACM SIGCOMM conference on Internet measurement [C], New York, 2010, 267-280.

[3] Xingwei Wang, Dapeng Qu, Min Huang, et al. Multiple many-to-many multicast routing scheme in green multi-granularity transport networks[J]. Computer Networks, 2015, 93(P1):225-242.

[4] Xingwei Wang, Hui Cheng, Min Huang. QoS multicast routing protocol oriented to cognitive network using competitive coevolutionary algorithm[J]. Expert Systems with Applications, 2014, 41(10):4513-4528.

[5] Kanagavelu, R., Mingjie, L.N., Khin Mi Mi et al. OpenFlow based control for Re-routing with Differentiated flows in DCNs [A], 2012 18th IEEE International Conference on Networks (ICON) [C], Singapore, 2012, 228-233.

[6] Changlin Jiang, Wei Liang, Mingwei Xu, Lili Liu. MTR: Fault tolerant routing in Clos DCN with miswiring links [A], Local & Metropolitan Area Networks (LANMAN) [C], Reno, 2014, 1-6.

[7] Ramos, R. M., Martinello, M., Rothenberg, C.E.. Data Center Fault-Tolerant Routing and Forwarding: An Approach based on Encoded Paths [A], 2013 Sixth Latin-American Symposium on Dependable Computing (LADC) [C], Rio de Janeiro, 2013, 104-113.

[8] Mingwei Xu, Yunfei Shang, Dan Li, Xin Wang. Greening DCNs with throughput-guaranteed power-aware routing [J], Computer Networks, 2013, 57: 2880-2899.

[9] D. Lin, Y. Liu, M. Hamdi, J. Muppala. Hyper-BCube: a scalable data center network. IEEE International Conference on Communications. Ottawa, 2012, 2918-2923.