Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
September 2008, 15(Suppl.): 1–8
The Journal of China Universities of Posts and Telecommunications
www.buptjournal.cn/xben

BGP parallel computing model based on the iteration tree

XU Ke, HE Huan

Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

Abstract

Scalable architecture has become the main developing direction of core router in future Internet, and the major technical challenge faced by scalable router is the parallel computing technology in its control plane. This paper conducts an in-depth study of the parallel computing model of border gateway protocol (BGP) in scalable router. Based on hierarchical iteration tree, this paper gives a reasonable improvement of arborous BGP division scheme, which has the advantages of balanced task division and low communication cost. Detailed performance analysis is done and is compared with neighbor-based division scheme considering balance in the initial partition. At last, three implementations of the two schemes are compared and evaluated by using actual routing table data gained from Route Views. The conclusion is that, the developed division scheme based on iterative tree has a comprehensive advantage, which can provide workable models and methods to the expansion of control plane in the future scalable router system.

Keywords internet, scalable router, distributed BGP, performance evaluation, routing table

1 Introduction

With the scale of the Internet continuously expanding and new applications ceaselessly emerging, the next-generation Internet puts forward higher and higher requirements to the performance and functionality of core routers. The scalable architecture, as an effective way to enhance the performance of core routers, has received wide attention and become an important research direction and technological difficulty.

At present, the mainstream router manufacturers have introduced products based on scalable architecture [1], such as Juniper T640 [2] and Cisco CRS1 [3], they mainly concerned about the scalability on hardware data transmitting plane. In the scalability research of software control plane, the representative production is the Zebra protocol software [4] of Kunihiro Ishiguro, whose main contribution is the realization of function scalability. Pluris also made a large-scale parallel router software system [5], which is a centralized control and management of multi-node redundant backup scalable software system. Above all, none of them truly realized the parallel computation and fully used the redundant resources in control plane.

In the router control plane, the routing protocol with high computing complexity occupies most resources in the entire software system. Therefore, based on scalable router architecture, parallelizing the work of routing protocol as much as possible is an effective way of improving the handling capacity of routing software. Inevitably, BGP, as the sole inter-domain routing protocol of Internet, is also confronted with the new challenge of distributed parallel processing.

In the view of real network applications, BGP itself is also under great pressure. On one hand, full-mesh topology of BGP routers in ISP leads to $o(N^2)$ complexity of the whole system with $N(N-1)/2$ session configurations. So tiered routing mechanisms in BGP, such as confederation, route reflector and other strategies are introduced to simplify network connectivity and improve the I-BGP scalability, but it also has brought a problem of more complex network configuration. On the other hand, CIDR analysis Ref. [6] points out that the number of AS in Internet has surged to 23 138 in April, 2008 from 6 500 in 2000, and the number of large ISP's AS neighbors has overrun (such as Tire-1 core ISP UUNet AS701's neighbor has reached 2 645). What's more, the revocation of prefix is up to 22 967 per second, nearly accounting for 8.22% of the total. Therefore, the research of BGP distributed computing simplifies the BGP attributes configuration, contributes to the construction, management, and maintenance of ISP network, increases the efficiency of routing protocol processing, and aims to provide

Received date: 10-07-2008
Corresponding author: XU Ke, E-mail: xuke@tsinghua.edu.cn

better network connectivity service.

The paper is organized as follow. Section 2 is the related research of the distributed routing protocol implementation. Section 3 introduces the scheme design of improved distributed BGP iterative tree algorithm. Section 4 gives the performance analysis and experimental verification. Section 5 includes the final conclusions and future work.

2 Related work

Scalable distributed model of routing protocol computing is the major issue of scalable router software research. And the researches of distributed routing computing in OSPF protocol and BGP are of great significance in the development of scalable router and Internet backbone network. It means the routing computation can be realized by scalable distributed routing protocol.

2.1 Distributed OSPF

OSPF is a broad deployed complex routing protocol; it is composed of multiple modules such as the maintenance of neighbor relations and database module, shortest path tree (SPT) computing module. In terms of the OSPF protocol operating in routers, the main bottleneck lies in SPT computation. Although some studies show that under the request of rapid detection of changes in network topology, the maintenance of neighbor relations can induce much system expenses, such cost can be alleviated by transferring neighbor relations module to the functions of line card [7] and finally will not become the system decisive bottleneck. So the distributed OSPF protocol model based on parallel SPT algorithm is one of the important components in scalable router architecture.

Currently, there has some research on the SPT parallel computing method. Although the SPT algorithm based on the structure of BTH [8,9] can achieve $O(\log n)$ complexity of calculation, whether all the network topology can be converted into BTH structure or not has not been discussed yet. The Ref. [10] presented a 'dot set' of the network division. For single-source shortest path tree problem, the algorithm can achieve $O(n)$ complexity of computing time. However, the algorithm can not solve all the division problem of random network topology in multi-plane, which is a NPH problem, being the biggest flaw of the algorithm. The Ref. [11] also proposed a multi-level division method which is aiming at the topology structure of the mesh network. Reference [12] proposed BPA algorithm, which can be carried out for distributed SPT computing with any network topology, but there is no substantial improvement in the algorithm complexity.

2.2 BGP parallel computing model

The basic idea of implementing BGP [13] parallel routing calculation is as follows: first, according to certain methods, distribute network prefixes or neighbor-sessions into each routing computing node and store in each node's local routing library. Then, during the process of routing computation, conduct independent local calculations by each node, and then, after the completion of routing calculation, synchronize routing on the basis of all the nodes' results. In this way, computing task of the whole routing table has been divided so not only the calculation cost of each node has been reduced, but also as a result of the distributive storage of candidate routings, the requirement of single node's storage resource is decreased obviously at the same time.

Reference [14] introduces the team work idea within the agent technology to put forward a kind of distributed BGP implementation model. The algorithm uses the concept of Agent to distribute routing prefixes among computation nodes, and through the definition of internal communication protocol it achieves collaborative computing of routings and the status synchronization.

Reference [15] presents the iterative tree division prototype idea based on the dual partial order relation $\preccurlyeq$. It divides a routing computation set based on the binary partial order $\preccurlyeq$ into a collection of such routing subsets based on the same relation, so by using of such hierarchical calculation we can realize this kind of distributed parallelism. However, based on this idea, regardless of the independent-cooperative structure or a two-layer-isolated structure, routing calculation task at each sub-leaf node doesn't consider balance scheduling. In the later one, the cost of neighbor information migrating between computation nodes is neglected completely. While in the form, all the routing information locally received is completely hold for the local calculation and it only achieves the consideration on the balanced decomposition of inside nodes' routing tasks and not includes any regard about the sub-leaf nodes.

Therefore, the high parallel degree of division scheme, which considers both the equalization of routing task division among the sub-leaf nodes and the reduction of the local communication cost when removing the restriction of the second calculation, will be the main content of our research.

3 The BGP parallel computing model based on the iteration tree

In order to facilitate the description, first, the definitions of some relative concepts are given.

Iterative tree: Fig. 1 describes in the bottom-up routing calculative election tree which can be called the routing calculation iteration tree. Leaf node is the routing information

from each other's neighbors; The inner node is routing computation node, actually, is the scalable router's processing board. Define the process layer of inner calculation is l, the total number of process boards in all the layers is u, and the number of leaf nodes is n (as the unreachable path is defined in origin paper, so this n also means the neighbors' number of the router system), sub-leaf nodes are the computation nodes that process the origin neighbors' routing task, that is the nodes in layer $l = 1$.

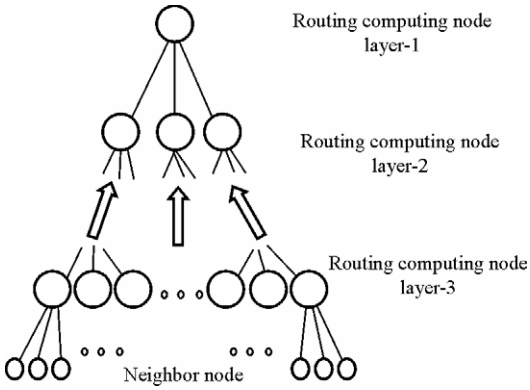

Fig. 1 The first process of computing optimal routing

k-band iteration tree: the iteration tree whose computation nodes' sub-nodes number is no more than k. k-band semi-fixed tree: iteration tree except for the sub-leaf node that collects the origin routing information from neighbors, the sub-nodes number of other computation nodes is all no more than k.

The main characteristics of the proposed level-based iteration tree expansion scheme is the considering resource constraints and communication costs, and at the same time achieving a better equalization division of neighbor nodes (that is, routing computing tasks between the nodes).

The basic process is as follows: first, chose local routing processing board to deal with the BGP path information that directly comes from its connected neighbor computation nodes, and then, take into account the more realistic problems such as the lack of routing node capacity and the balanced division. In other words, in load distribution, the local load takes priority to be assigned to the local node, which is left to be handled by the local routing processing board of the local node. Then, the remaining calculation task is re-distributed among computing nodes that are of redundancy computing power. Finally, assemble the routing calculating results of each layer step by step until having got the best routing at the root node. According to this idea, improve the original iteration tree algorithm of the independent-cooperative structure and we can obtain a distributed model and algorithm that is applicable to independent collaboration.

3.1 Problem analysis

No matter adopting which kind of neighbor-based division scheme, we should consider the integrity problem induced by the incomplete routing information convergence. The problem is described as: the routing information from its neighbor conversation is stored in their respective corresponding routing library to be processed by the local computation node. However, as different BGP neighbor conversations are likely to receive candidate routings of the same network prefix and they might not belong to the same routing computation node. In order to get global optimal routing and maintain a unified view on the routing table, the integrity problem within routing information processing arises inevitably. That is to say, for the network routings with different destinations, every routing computation node only needs to notify them to the other computation nodes, but for routings with the same destination, it needs a second calculation before the release.

In the neighbor-based division model, we could use centralized and distributed scheme to maintain the consistency of routing. However, according to the scheme mentioned above, the best route must be obtained after two rounds of calculations, whether it is centralized scheme or distributed one. This undoubtedly increases the burden of routers, especially when there are a lot of neighbor computation nodes. Moreover, centralized scheme may incur the single-node-fault problem, and there is an expansibility problem (the difficulty of expansibility caused by full-mesh communication of computation nodes) in distributed scheme. Here we could consider the expansible hierarchical connected structure.

With respect to the expansible hierarchical connected structure, certain quantity of computing work is assigned to each node. It not only gets rid of the second-time computing limitation of neighbor-based division model, but avoids the single-node-fault problem caused by centralized scheme as well. Furthermore, it improves the expansibility of distributed schemes. This paper makes proper improvements based on the original model, stresses the performance analysis and gives compared evaluation with the performance analysis results of the improved neighbor-based division scheme.

Computing thinking is as: based on the unique routing choice requirements, the iteration must come to an optimal path in the end, and it seemed to be a nested computing mode, which is also the reason for the name of hierarchical iteration model. According to a certain standard, routing information is divided into small sets to conduct parallel computing, then the results are re-allocated to the new sets to be computed in parallel, until all of these local optimal solutions are converged to the last one layer, that the unique global optimal solution can be obtained, which can be comprehended as an arborous computing structure with inner nodes to be calculating nodes

and leaf nodes to be routing conversation neighbors.

Consider different ways of work distribution, Fig. 1 shows the double-layer-isolated structure (data plane and control plane are independent) division scheme and Fig. 2 shows the independent-cooperative structure (improved at local priority) division scheme.

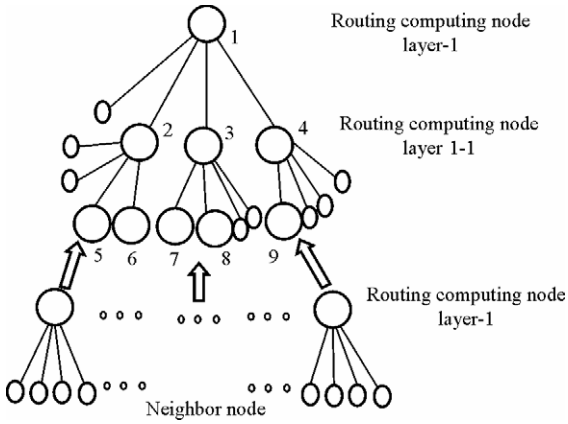

Fig. 2 The second process of computing optimal routing

3.2 Scheme design

According to computing thinking mentioned as above, we improve the double-layer-isolated structure algorithm and derive a distributed model and algorithm that is applicable to independent collaboration. The detailed description is as follows: distributed thinking aims to construct a load balanced k-band tree ultimately. For the independent-corporative structure, the redundant computing resource of sub-leaf nodes can be used by hierarchical computation. Define the load of sub-leaf nodes as the initial load, presents by R_{ini}, the remainder computing capacity is $R_{\text{re}} = k - R_{\text{ini}}$ If for a sub-node $R_{\text{ini}} \geqslant k$, so, define $R_{\text{ini}} = 0$, which means it will not be seen as a calculating node appeared in inner nodes (only be left in sub-node layer). In addition, define the new routing calculating board with the capacity of $R_{\text{re}} = k$.

$$R_{\text{re}} = \begin{cases} k - R_{\text{ini}}; & k \geqslant R_{\text{ini}} \\ 0; & k < R_{\text{ini}} \\ k; & \text{new proceeing board} \end{cases} \tag{1}$$

Theory 1 For the number of inner nodes u and leaf nodes n of k-band iteration tree satisfy:

$$u \geqslant \frac{n-1}{k-1} \tag{2}$$

Proof Known by definition, all the inner nodes have less than k sub-nodes. Firstly, because each node (except for root node) has a corresponding edge, the total number of edges e is $e = u + n - 1$. Secondly, as for the sub-nodes of each inner node is less than k, so $e \leqslant uk$; above all, $uk \geqslant u + n - 1$, at the same time, as the sub-nodes of each inner node is more than 1, therefore: $u(k-1) \geqslant n-1 \Rightarrow u \geqslant (n-1)/(k-1)$.

When each of the inner nodes has exactly k sub-nodes, it can be equal sign.

Theory 2 For the number of inner nodes u and the origin sub-leaf nodes' remainder computing capacity of k-band semi-fixed tree satisfy the below inequality:

$$\sum_{u} R_{\text{re}_i} \geqslant u - 1 \tag{3}$$

That is to say that except for the root node, all the computing work of the other computation nodes has been converged layer by layer completely. The proof method is as Theory 1, yet ignoring the work division among sub-leaf nodes, leaving the total number of considered division nodes as $u-1$. When all the nodes except for the sub-leaf nodes which have no remainder load capacity of k, the inequality can be equal sign. By the derived iteration tree, it is easy to calculate the number of increased new processing boards by finding who satisfies $R_{\text{re}} = k$.

The concrete independent-collaborative division process is as follows: first, arrange processing boards by remainder load capacity $R_{\text{re}} = k - R_{\text{ini}}$. Then calculate new $R_{\text{re}}[]$, for example, as Fig. 2 shows, here we can see $k = 4$, let the number one node (remainder load capacity is 3) to be the root node, computation node of number 2, 3 and 4 respective remainder load capacity are 2, 2 and 1. Knowing that $R_{\text{re}}[] = \{0, 5, 7, 8, \ldots\}$, obviously $x = 2, 3, 4 \leqslant R_{\text{re}}[1] - 1 = 4$, so they all belong to number 1 computation node. Similarly, with such algorithm we can make the iteration tree as low as possible. In addition, it is clear that when for the entire initial nodes if $R_{\text{re}} = k - R_{\text{ini}} = 0$, the structure is equivalent to the two-layer isolated structure. The only different between the two structures is that at this division structure the initial work distribution has been done among sub-leaf nodes by nature, and what needs to do is the convergence of routing information from inner computation node.

The following is the algorithm description of the independent-cooperative structure algorithm k-band semi-fixed iteration tree, referencing iteration tree model in Fig. 2

1) Calculate all the sub-leaf nodes' initial available load capacity $R_{\text{re}}[]$, and sort them at a descending order 1, 2,…, u.

2) Initialize remainder capacity array, $R_{\text{re}}[0] = 0$, and as new board's capacity is k, (note there are MAX processing boards), so $R_{\text{re}}[u+1]$, $R_{\text{re}}[u+2], \ldots, R_{\text{re}}[MAX] = k$.

3) Re-calculate $R_{\text{re}}[i]$ as follows and take nodes (leaf or computation nodes) as task units to distribute to computation nodes who have remainder computing capacity.

int $i = 1$

do{

$R_{re}[i] = R_{re}[i] + R_{re}[i-1];$

$i = i + 1;$

}while($R_{re}[i] \leqslant u - 1$).

4) In the end, $i - u$ is the increased number of new processing boards, if there exists any, the serial number of old nodes are pushed by turn, and then insert the new board to the front.

5) Chose number 1 node to be the root node of the iteration tree.

6) As to the remaining nodes, insert them into its corresponding father-node in turn by their serial number order to conduct routing calculation. The distribution method is as: if the leaf node number x, find one pair of adjacent elements that meet $R_{re}[i-1] \leqslant x \leqslant R_{re}[i] - 1$, then the father-node of the node is number i.

The time complexity of this division scheme is $o(nu + s\log_k(n+u))$, which is composed of the work-dividing time and the routing time (s is the total number of routing information).

4 Performance analysis and evaluation

4.1 Performance analysis

$-u$ is the number of routing computation nodes. $-n$ is the number of all the neighbor (leaf) node. $-l$ is the depth of iteration trees, or number of convergence layers. $-k$ is the max sub-nodes number of each inner node of k-band iteration tree. $-s$ is the number of routings from neighbors. $-\alpha$ is the transmission-computing ratio.

1) Calculation acceleration

$$S_{cal} = \frac{t_{old}}{t'_{new}} = \frac{s(n-1)}{(s+(l-1)(k-1)+(l-1)\alpha)} \approx \frac{n-1}{k-1} \tag{4}$$

As for core router, s is much bigger than the value of 1 and k. And if we ignore the transmission time between layers, the acceleration can be closed to Eq. (4). As shown in Fig. 3 (ignoring the transmission cost), it is the relationship among calculation acceleration, the number of neighbors and the number of computing nodes.

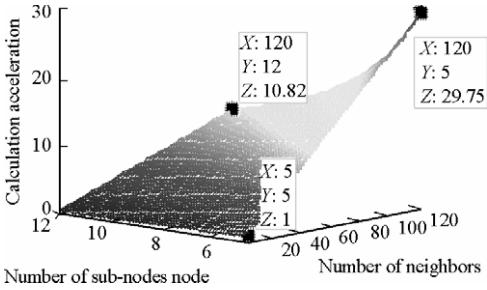

Fig. 3 Computing performance analysis of BGP iteration tree division scheme

The conclusion is that for the improved iteration tree model, the calculation acceleration is proportional to the number of neighbors m and is inversely proportional to the number of leaf nodes k. And compared with neighbor-based scheme, for the entire distribution system adopting neighbor-based division scheme, since it takes into account of the routing release and the second calculation problem, the increase in calculation performance is not remarkably enhanced with the increasing of neighbors. Considering the fact that the more computing nodes, the more routings will be calculated by second calculation, so it does not always means better. For example, as for a small neighbor scale, the calculation acceleration is not sure to be enhanced with the increasing of computation nodes, so for our model it is different with the conclusion derived by the neighbor-based division scheme.

As shown in Fig. 4, the larger is k, the smaller is the calculation acceleration S_{cal}. And intuitively, it leads to the fewer parallel computing boards and the lower altitude of the iteration tree to conduct pipeline computation ($l = \log n$). So, the parallel degree is not high. Therefore it is better to integrate the neighbor amount and other factors to decide the number of processing boards needed, but not just pursue the high-efficiency and give blind investment.

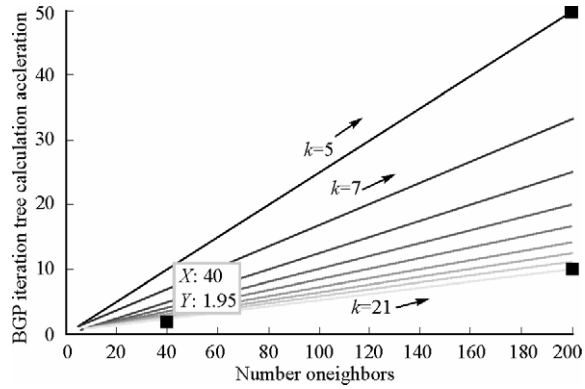

Fig. 4 BGP iteration tree-acceleration analysis

Ignoring the transmission time, we get conclusions as above. Then, the overall computing performance is analyzed under the consideration of different transmission-computing time ratio.

$$S_{cal} = \frac{s(n-1)}{s(k-1)+(l-1)\alpha} \tag{5}$$

In terms of transmission-computing time ratio, apply it to the scope of $[1, 10^4]$. With formula $n = k^l$ we compute l to do the performance analysis. For the sake of convenience, we set $n = 27$, $k = 3$. And then the relationship among routing amount, transmission-computing time ratio and calculation acceleration is analyzed.

Figure 5 gives the relationship between calculation acceleration and routing amount at different α (transmission-computing

time ratio) in the BGP iteration tree division algorithm (13 cooperation nodes). From the figure we can see regardless of the value of α, when the routing is more than 3 500 000, the acceleration will gradually stabilize, and get closed to the ideal theoretical value 13.

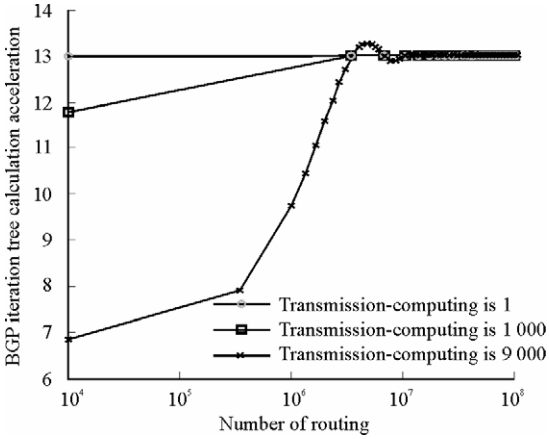

Fig. 5 The routing-calculation acceleration relation

Figure 6 gives the relationship between calculation acceleration and transmission-computing time ratio at different *s* (routing amount). Regardless of the value of *s*, when the α (transmission-computing ratio) is smaller than 1 000, the acceleration will gradually get closed to the theoretical value 13.

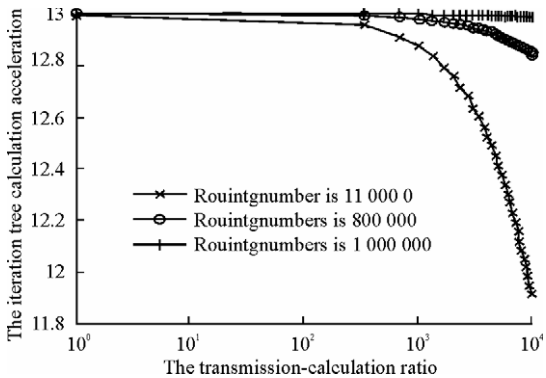

Fig. 6 The α -calculation acceleration relation

The Fig. 7 is the map of relationship among routing amount 0–10^7, transmission-computing ratio 0–10^4 and the calculation acceleration.

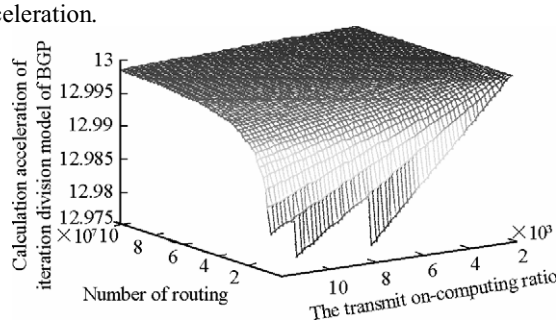

Fig. 7 The calculation acceleration analysis (transmission-computing ratio, routing amount)

To summarize, under the circumstance that we have already known the routing table size and considered the impact from system transmission-computing ratio, we can measure the number of each node's child-node in the routing calculation cluster system in order to decide the optimum number of processing boards needed. We can also decide the optimal child-node number of computation nodes with the known number of neighbor-nodes (leaf-nodes) and processing boards, and then analyze whether they meet the system performance requirements.

2) Storage compression ratio

$$S_{\text{mem}} = \frac{n_{\text{new}}}{m_{\text{old}}} = \frac{k}{n} \tag{6}$$

System cost ratio

$$R_{\cos} = \frac{(n-1)k}{(k-1)n} \geqslant 1 \ (n \geqslant k) \tag{7}$$

3) Failure recovery acceleration ratio

$$S_{\text{rec}} = \frac{t_{\text{re\_old}}}{t_{\text{re\_new}}} = \frac{snt_{\text{com}}u(1-k)}{(1-k^l)(s(k-1)t_{\text{com}})} = \frac{n}{k-1} \tag{8}$$

The evaluation of such scheme: firstly, from the calculation amount perspective, in this scheme the computing tasks are distributed in various computing nodes in a certain quantity, unlike the neighbor-based model division scheme which that encumbered with enormous amount of computation work and low efficiency in the second round computation. Secondly, as to the storage performance, single node storage is close to k/n, unlike in the neighbor-based division scheme that of which can reach to the best result of the whole routing as $(m+t)/(mt)$. And when $m = t$, we can get a near linear compression ratio, which is $2/m$ (ordinary nodes in the centralized scheme). What's more, the worst is equal to the origin routing amount. When neighbor number m is enormous, it closes to $1/t$. At last, from the failure recovery respective, it not only gets rid of the second-time computing limitation of neighbor-based division model, but avoids the single-node-fault problem caused by centralized scheme as well. Furthermore, it improves the expansibility of distributed schemes.

However, compared with neighbor-based division scheme, there are also some obvious drawbacks.

First, based on the routing data analysis, routings to the same destination network aren't so much, so the computing resources is mainly consumed by the huge number of different routings;

Secondly, the communication cost in the routing election process of hierarchy iteration tree should be considered. In the computing model, the layers interconnected. And with the liner increase of layers, the communication link increased correspondingly.

In addition, unlike the neighbor-based division scheme, in the improved iteration tree model, a candidate route may be

compared with other routes for several times before it is removed, so it does not reduce comparison times by nature.

Therefore, the model gets calculation acceleration at the cost of communications between layers and the increased processing boards by using a distributed parallel algorithm.

Table 1 shows when the neighbor-scale is small, we adopt the improved neighbor-based work distribution scheme (considering equilibrium) and hierarchy iteration model with two-layer isolated structure algorithm to compare the computation acceleration (ignoring the impact of transmission time).

Table 1 Calculation performance comparison between two computation models

Neighbor amount		40	80	160	200
Neighbor-based division scheme (t)	5	3.08	3.81	4.32	4.44
	13	2.49	4.18	6.23	7.06
	21	1.75	3.26	5.59	6.55
Iteration tree division-scheme (k)	5	9.75 (10)	19.75 (20)	39.75 (40)	49.75 (50)
	13	3.25 (4)	6.58 (7)	13.25 (14)	16.58 (17)
	21	1.95 (2)	3.95 (4)	7.95 (8)	9.95 (10)

The above computing results shows, at ideal circumstance (routing table is big enough and without impact of transmission cost), when routing computation nodes' number is certain (neighbor-based division) or the sub-leaf nodes' number of inner nodes (iteration tree) is fixed, the calculation acceleration is enhanced with the increase of neighbors. However, in view of the increasing speed of the acceleration, the iteration tree model can reach a comparatively higher value. For example, when the route system has 200 neighbors, the iteration tree model uses 10 processing boards to reach 9.95 calculation acceleration, and the neighbor-based uses 13 processors only to reach 7.06; also, in 80 neighbors, the former model uses 4 processing boards to reach 3.95 calculation acceleration, and the later uses 5 only to reach 3.81; even iteration tree with 20 processor can reach 19.75 acceleration, but neighbor-based with 21 processor only to reach 3.26 where the performance decreases on the contrary.

4.2 Experiment verification based on actual routing table

In this section, we select actual routing table data to carry out relevant simulation verification, and by using the two BGP distributed models' real division schemes, we compare real calculation performance, storage capability, reliability and so on with each models' respective theoretical values. Finally, we compare the experimental results among schemes.

Because in the present network there are routing information service (RIS) projects which provide records of BGP routing table data, and there are also actual routing table data from AS6447 (University of Oregon Route Views Project) [16], which supports the analysis of BGP routing tables and network reachability. This paper just adopts the real-time data downloaded from the Route Views [17] and analyzes it.

After processing the data of routing tables, we know that there are 269 152 reachable network prefix and 10 204 583 network routes, thus there are 37.9 reachable routes for each network on average. This does not agree with the neighbor number (36) from statistics. After viewing the routing tables, we found that some AS connection are back-ups, which are AS2914, AS3130, AS3549, AS286, AS852 and AS6453. Thus it is not surprising to get that conclusion. Moreover, according to the statistics, the routes to certain prefix network are generally between 35 and 42, which just prove our claim. Because there are exactly 36 neighbors, among whom 6 neighbors have back-up routes, we consider the setting before division; exactly two levels of iteration tree could perform the task computation.

When performing the comparison experiment, we select the improved neighbor-based division scheme derived from agent thought (taking task assignment balance into account) to verify and evaluate the performance, neighbor-based division scheme (considering task assignment balance, see Fig. 8) and hierarchical iteration tree (see Fig. 9).

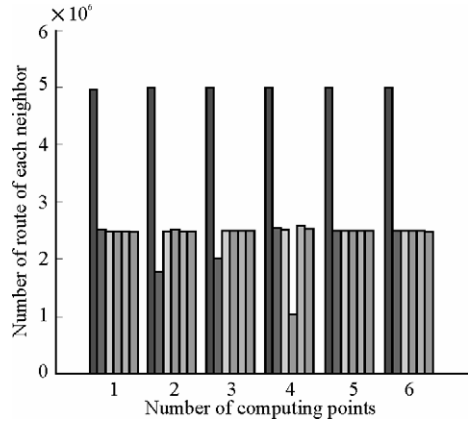

Fig. 8 The improved neighbor-based division results

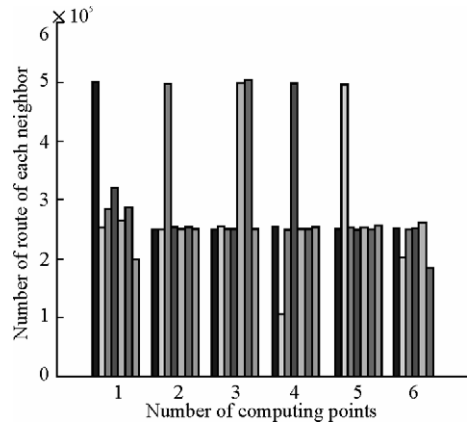

Fig. 9 The two-layer isolated iteration tree division results

The neighbor-based scheme needs 6 computation nodes, the improved two-layer-isolated structure of the iteration tree

model request 6, and the independent-cooperative structure scheme request 7 computation nodes (detailed division is ignored). The analysis is as Table 2.

Table 2 The performance analysis on division-results of three schemes by two models

Performance analysis	Calculation acceleration		Storage compression ratio (centralized-node)		Failure recovery acceleration ratio (non-centralized-node)	
	Theoretical	Actual	Theoretical	Actual	Theoretical value	Actual
Neighbor-based	3.000	3.041	0.305 6	0.302 4	5.375	5.078
Two-layer isolated structure	5.833	4.375	0.194 4	0.219 7	6.000	4.840
Independent-collaborative structure	7.000	5.833	0.166 7	0.170 5	7.200 0	6.473

From the experiment results, we can conclude that when computation nodes are all 6, the independent-collaborative iteration tree has a better performance than the neighbor-based division scheme, which fits the theoretical analysis in 4th section well. And we can predict that the more layers, the more obvious is such difference. In terms of storage performance, the memory occupancy rate of Iteration tree is 21.97%, and for neighbor-based model, the centralized storage nodes occupy 30.24%, the non-centralized nodes occupy 19.16%. They are different at about10% with each other. In view of failure recovery, the failure recovery acceleration ratio of the improved independent-collaborative Iteration tree algorithm and neighbor-based model with failure of non-centralized node have difference less than about 5%, but if the former compares with the centralized-node failure of the later model, practically speaking, the failure recovery acceleration ratio of iterative tree model is enhanced nearly 60% of the neighbor-based model.

When the distributed work of sub-leaf nodes and the neighbor-based computation nodes are the same, from comparison as above, considering the equilibrium division of the iteration tree, this paper verifies it can achieve better performance, and the only thing required is the extra computation node for the needed changes from 6 to 7. As a result, the most important thing in choosing a proper deployment is to give an overall consideration on the actual economic situation and the performance requirements.

5 Conclusions and future works

The scheme proposed here can provide balanced distribution of route computation nodes' tasks in the condition of meeting the resource constraints and reducing the cost of communication. Its essence is that when giving priority to the division way that local load distributed to itself, we also take into account the tasks migration problem, which is caused by inadequate capacity of routing node or for balance division. That is to say, when doing the load assignment, local load is preferred to be distributed to the processing board of local sub-leaf node, which the node belongs to. And then, the remaining tasks will be re-assigned again among the nodes with redundant calculation ability. Finally, the routing calculating results of each layer are converged step by step until having got the best routing at the root node.

Using actual routing table data, we select three implement schemes of the two division models to do experiments. Then, we compare the results in pairs, and conclude that iteration tree model with independent-collaborative structure has the optimal performances. Under this routing data test condition, its calculation capability increases 21% than that of another scheme from the same model, also is more than twice of the capability of the improved neighbor-based division scheme. What's more, its storage compression ratio is also better than the improved neighbor-based division scheme. And the failure recovery capability increases 32.1% than that of another scheme from the same model, is also 1.3 times of the improved neighbor-based division scheme. Above all, on considering of equilibrium and high parallelization, the paper provides a scheme which achieves the best performance of the all the mentioned ones, though at the cost of an extra computation node.

The following works are our further concern: first, to assign the routes coming from the same neighbor-node to different computation nodes in order to attain a more fine-grained division which can achieve a better equilibrium. Second, in this paper, we use the hypothesis that the initial interfaces are connected manually. Although this is easy to realize in reality, the network topology is always changing in real time. So it can give some inspiration to study the dynamic auto-update division scheme. One thing is much interesting that when analyzing the routing table data, we find that parts of neighbor AS routers are double back-up, which increase the routes' number notably. How to deal with such data and the influence they made on performance will be studied in the near future. Finally, due to non-consideration of actual transmission–calculating ratio, there will be different degree of deviations based on different orders of magnitude. Through more research, this should be more precise in future analysis.

Acknowledgements

This work is supported by the National Basic Research Program of China (2003CB314801) and the Hi-Tech Research and Development Program of China (2008AA01A326).

To p. 18

and handover key generating mechanism, constructed and analyzed the system model, we have presented the combination scheme of mobile handover and AAA function for fast and secure handover so as to provide reduced handoff latency while maintaining the security capability. The analytical results show that the proposed combination scheme is superior to the previous simple one. From the comparison of the data of the cost of vehicle and pedestrian, the faster the MN moves, the lower the PMR value will be, and this proves the advantage of this scheme.

Acknowledgements

This work is supported by the Hi-Tech Research and Development Program of China (2007AA01Z226).

References

1. Johnson D, Perkins C, Arkko J. Mobility Support in IPv6. IETF RFC3775, June 2004
2. Rajeev Koodli, Fast handovers for mobile ip, draft-ietf-mipshop-fmipv6-rfc4068bis-01.txt, 3 March 2007
3. Soliman H S, et al. Hierarchical Mobile IPv6 Mobility Management. IETF RFC 4140, Aug 2005
4. Jung H Y, et al. Fast Handover for Hierarchical Mobile IPv6 (F-HMIPv6. IETF Draft draft-jung-mobopots-fhmipv6-00.txt, April 2005
5. Calhoun P, et al. Diameter protocol. IETF RFC3588, September 2003
6. Wei D, Liu Y H, Yu X G, Li A D. Research of Mobile Ipv6 Application Based on Diameter Protocol.2006 International conference on Multi-Symposiums, Computer and Computational Sciences (IMSCCS'06), IEEE Computing Society Press, 2006
7. Laurent M, Dupont F. Inter-domain security for mobile Ipv6. The 2nd European Conference on Universal Multiservice Networks (ECUMN 2002), IEEE press, Apr 2002, 238–245
8. Faccin S M, et al. Mobile Ipv6 authentication, authorization and accounting requirements. draft-le-aaa-mipv6-requirements-02.txt, April 2003
9. Narayanan V, et al. Establishing Handover Keys using Shared Keys. draft-vidya-mipshop-handover-keys-aaa-04.txt, IETF Draft, March 6, 2007

From p. 8

References

1. Iyer S, McKeown N. Analysis of the parallel packet switch architecture. IEEE/ACM Transactions on Networking, 2003, 11(2): 314–324
2. Juniper Networks, Inc. T640 routing node and TX matrixTM platform: architecture. White Paper (Part Number 350031-001), 2004. http://www.juniper.net
3. Cisco Systems, Inc. Next generation networks and the cisco carrier routing system. White Paper, 2004. http://www.cisco.com
4. GNU Zebra . Routing software. [2008-3-24]. http://www.zebra.org/history.html
5. Sam H. Pluris massively parallel routing. White Paper, 1999. Pluris Inc
6. CIDR Report [EB/ OL] [2008-2-10]. http :/ / www. cidr-report. org
7. Basu A, Riecke J G. Stability issues in OSPF routing. Proceedings of SIGCOMM'01, Aug 27–31, 2001, San Diego, California, USA: 225–236
8. Shan Z, Huang G M. A new parallel and distributed shortest path algorithm for hierarchically clustered data networks. IEEE Transactions on Parallel and Distributed Systems, 1998, 9(9): 841–855
9. Antonio J K, Huang G K, Wei K. Tsai a fast distributed shortest path algorithm for a class of hierarchically clustered data networks. IEEE Transactions on Computers, 1992, 41(6): 716–724
10. Edith C. Efficient parallel shortest-paths in digraphs with a separator decomposition. AT&T Bell Laboratories Murray Hill, 1993
11. Romeijn E H, Smithy R L. Parallel algorithms for solving aggregated shortest path problems. 1997
12. Zhang X P, Zhang N, Wu J P, et al. BPA—a parallel shortest path algorithm for cluster-router. PDCS 2006, Dallas USA
13. Rekhter Y, Li T A. Border Gateway Protocol 4 (BGP-4). RFC 1771 (Draft Standard), March, 1995. http://www.ietf.org/rfc/rfc1771.txt. Obsoleted by RFC 4271. 2006. RFC 4271
14. Zhang X Z, Zhu P D, Lu X C. Fully-distributed and highly-parallelized implementation model of BGP4 based on clustered routers. Proceedings of Networking ICN 2005: 4th International Conference on Networking, Apr 17–21, 2005, Reunion Island, France, 2005: 433–441
15. Wu K, Wu J P, Xu K. A tree-based distributed model for BGP route processing. International Conference on High Performance Computing and Communications (HPCC), 2006: 119–128
16. University of Oregon RouteViews Project, Advanced Network Technology Center .University of Oregon. [2008-2-21]. http://www.routeviews.org/
17. Meyer D. University of Oregon Route Views Archive Project. [2008-4-22]. http://archive.routeviews.org/route-views3