A New Precomputation Scheme for MPLS Traffic Engineering Routing

Zhaowei Meng[1], Jinshu Su[1], Vittorio Manetti[2]

[1] School of Computer, National University of Defense Technology,
Changsha 410073, P.R.China
{zwmeng, sjs}@nudt.edu.cn
[2] COMICS Lab, Dipartimento di Informatica e Sistemistica,
Universit`a di Napoli Federico II,
Via Claudio 21, 80125 Napoli, Italy
stavallo@unina.it

Abstract. This paper presents a new precomputation algorithm for Multi Protocol Label Switching (MPLS) traffic engineering routing. The prior MPLS routing algorithms try to minimize the interference between different source-destination pairs by circumventing the critical links. But the process of identifying critical links is very computationally expensive. The main contribution of this paper is a new precomputation approach of route selection considering the interference. The proposed algorithm reduces online computing complexity through efficient precomputation. From the simulation results, the proposed algorithm outperforms prior algorithms in terms of efficiency and complexity.

1 Introduction and Related Works

Nowadays, the most frequently used routing algorithm in Internet is the Shortest-Path-First (SPF) algorithm. This algorithm may potentially cause some links being bottleneck and lead to poor resource utilization. So based on SPF, many algorithms which consider load balancing have been proposed, but MIRA [1] is the first algorithm which utilizes the knowledge of SD pairs and considers the interference phenomena. The problem of minimum interference routing is to find a path that maximizes the maxflow between all other SD pairs. This problem is shown to be NP hard. So M. Kodialam et al. give a heuristic algorithm - MIRA. The core notion of MIRA is “critical link”. MIRA tries to avoid routing LSPs on such critical links of other SD pairs. So it performs better than former algorithms.

But MIRA also has some shortcomings. One shortcoming is that some links which are believed as non-critical by MIRA are shown to be indeed very important. S.Suri et al. illustrated this point by some special topologies [2]. Bin Wang et al. propose NewMIRA algorithm which utilizes maxflow value of SD pair and sub-flow value on the link to estimate its importance [3]. In [4], the authors provide an algorithm which divides the link criticality into multiple classes.

The other shortcoming of MIRA is that the complexity of maxflow computation is very high. This limits MIRA's application in practical networks. W.S.S. et al. propose a new notion of interference [5] which uses the number of possible paths per link to denote link's criticality. W.S.S.'s approach reduces the complexity. In [6], the authors use the notion of criticality threshold to precompute more effectively.

This paper propose a new pre-computation approach for routing bandwidth guaranteed label switch path, which tries to consider influence resulted by all critical or "non-critical" links. Our approach reduces online computation complexity through efficient pre-computation. Extensive simulations were carried out to evaluate the performance of the proposed algorithm. The result shows that our approach performs better than former algorithms.

The rest of this paper is structured as follows. Section 1 reviews the main idea of MIRA and some related works. In Section 2 we propose a new pre-computation algorithm for routing bandwidth guaranteed flows, and describe it in detail. In Section 3, the efficiency of our new algorithm is evaluated and finally, Section 4 concludes our work.

2 Proposed Algorithm

2.1 System Model

Given a network represented by a directed graph (V,E) where V is a set of nodes and E is a set of links. The number of nodes is n and the number of links is m. The LSP setup requests are between specific source nodes and destination nodes. The SD pairs are $\{S_0,D_0\},\{S_1,D_1\},\ldots,\{S_p,D_p\}$, where p is the number of SD pairs. We denote all these SD pairs by a set P. Each LSP set-up request arrives at ingress node. The requests arrive online, one by one, and there is no prior knowledge for future demands. Each ingress router knows the whole network's topology and state information of the links. The initial capacity of link l is denoted as $R(l)$, while the current available bandwidth is $r(l)$. The LSP request r_i is defined by a triple (s_i,d_i,b_i), where $(s_i,d_i) \in P$, and b_i is the amount of bandwidth required by the LSP. The objective is to find a feasible path (if exists) for LSP request r_i, otherwise the request will be rejected. In this paper, we focus on the routing of bandwidth guaranteed paths. No rerouting or request splitting is allowed.

2.2 Proposed Algorithm's Details

In this section, we will present our approach's details. We consider not only the critical links, but also "non-critical" links. The sum of sub-flows belong to different SD pairs traveling through the link will be used to estimate its interference degree. The residual bandwidth and hop counts are also considered. We will define a novel link weight function here:

$$w(l)=\begin{cases}\dfrac{b*\sum_{(a,b)\in P\setminus(s,d)}\alpha_{ab}f_{ab}(l)}{r(l)*\alpha_{sd}f_{sd}(l)}+C & if\ \ f_{sd}(l)\neq 0\\ \dfrac{b*\sum_{(a,b)\in P\setminus(s,d)}\alpha_{ab}f_{ab}(l)}{r(l)*\alpha_{sd}r(l)}+C & if\ \ f_{sd}(l)=0\end{cases} \quad (1)$$

Where α_{sd} represents the relative importance of SD pair *(s,d)*. And $f_{sd}(l)$ is the amount of sub-flow traveling through the link *l* when the maximum flow between *(s,d)* is achieved. The item *C* is a control constant.

In the link weight function, link's residual bandwidth *r(l)* denotes the link's ability to hold future LSP requests. If the residual bandwidth is bigger, the link weight will be smaller and the algorithm will try to route LSP through such links. And if $\alpha_{sd}f_{sd}(l)$ is bigger and $\sum_{(a,b)\in P\setminus(s,d)}\alpha_{ab}f_{ab}(l)$ is smaller, we believe that routing request through link *l* will cause less impact on other SD pairs. So the algorithm will trend to route LSP on such links. Otherwise the algorithm will trend to avoid such links.

If $f_{sd}(l)=0$, it shows that there is no flow traveling link *l* when the maximum flow is achieved. Because the way to achieve maximum flow is not distinct, it doesn't mean the request can't travel through link *l*. So we use $r(l)$ as a substitute for $f_{sd}(l)$.

Item C is a dynamic control constant. If constant *C* is chosen very big, the algorithm behaviors like SPF. If *C* is chosen relatively small, the algorithm trends to minimize the interference and balance the load.

But calculating this link weight still needs *p* maxflow computations. This is expensive for online routing. In order to reduce the online complexity, our algorithm adopts two phases - precomputation phase and online routing phase. Through effective precomputation, our approach could reduce online complexity successfully.

In the pre-computation period $t_k(k=0,1,2,...)$, we compute maximum flow for each SD pair and record the sub-flow at that time $f_{sd}(l,t_k)$. These values will be used by online phase to predict the degree of link congestion. The pre-computation phase runs periodically, or anytime the topology and the SD pairs changed.

In the online phase, given an LSP request *(s,d,b)* to be routed, we try to estimate the impact of routing current request on the link to other SD pair by a link weight function as below.

$$w(l)=\begin{cases}\dfrac{b*\sum_{(a,b)\in P\setminus(s,d)}\alpha_{ab}f_{ab}(l,t_k)}{r(l)*\alpha_{sd}f_{sd}(l,t_k)}+C & if\ \ f_{sd}(l)\neq 0\\ \dfrac{b*\sum_{(a,b)\in P\setminus(s,d)}\alpha_{ab}f_{ab}(l,t_k)}{r(l)*\alpha_{sd}r(l)}+C & if\ \ f_{sd}(l)=0\end{cases} \quad (2)$$

The algorithm's detailed pseudo code is listed below.

The Proposed Algorithm

INPUT: A residual graph $G = (V, E)$, a set P of all the source-destination pairs and LSP request $r(s, d, b)$, which is a request for b bandwidth units between pair (s, d).
OUTPUT: A path from s to d with b bandwidth units.
PRECOMPUTATION PHASE:

At the pre-computation time point $t_k (k = 0,1,2,...)$, $\forall (a,b) \in P$

1: Compute the maximum network flows,

2: $\forall l \in E$, record the amount of flow passing through the link- $f_{ab}(l, t_k)$.

ONLINE PHASE:
1: Compute the weight $w(l)$ for all $l \in E$ according to equation (2) .
2: Eliminate all the links whose residual bandwidth less than b.
3: Run Dijkstra algorithm using $w(l)$ as the weights in the reduced network.
4: Create an LSP from s to d with b bandwidth units and update the links' available bandwidth.

2.3 Complexity Analysis

In precomputation phase, computing maxflow for each SD pair using the highest label preflow-push algorithm needs $O(n^2\sqrt{m})$. There are p SD pairs, so the computation need totally $O(pn^2\sqrt{m})$. This is on the same level as MIRA and NewMIRA's $O((p-1)n^2\sqrt{m})$ .In online phase, Step 1 and step 2 both need $O(m)$. Step 3 needs $O(n^2)$. So the online computation complexity is only $O(n^2)$.

If the network state changes frequently, the algorithm will have to execute pre-computation more frequently and the total run time of algorithm will increase. But in the worst case, its complexity is no higher than MIRA and NewMIRA.

3 Performance Studies

The network topology and SD pairs used in the simulation are shown in figure 1. This topology was first used by M.S. Kodialam et al. in [1].

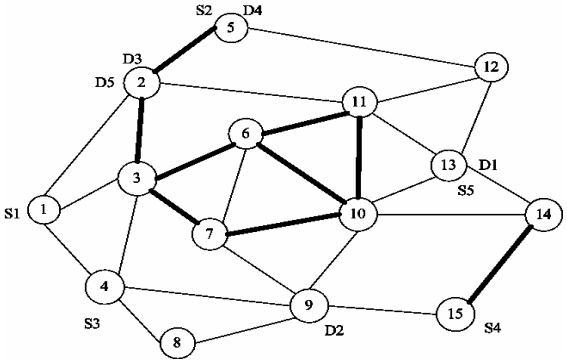

Figure 1. The KL topology [1]

Lighter links have capacity of 1200 bandwidth units, while the darker ones have 4800 units. Links are bi-directional. Requests were randomly generated using the uniform distribution of bandwidth demand in the interval [1, 4]. The LSPs are long lives. 8,000 requests were randomly generated among the five SD pairs. Our algorithm runs at the intervals k=128. In this scenario, we assume the accurate resource availability information is available when selecting the route. And the constant C in the weight function is set to zero.

Figure 2 presents the number of rejected requests. From the figure we can see that both our proposed algorithm and NewMIRA rejected fewer requests than SPF and WSP all the time. And our algorithm rejected fewer requests than NewMIRA after 5000 requests, when the pre-computation period is 128.

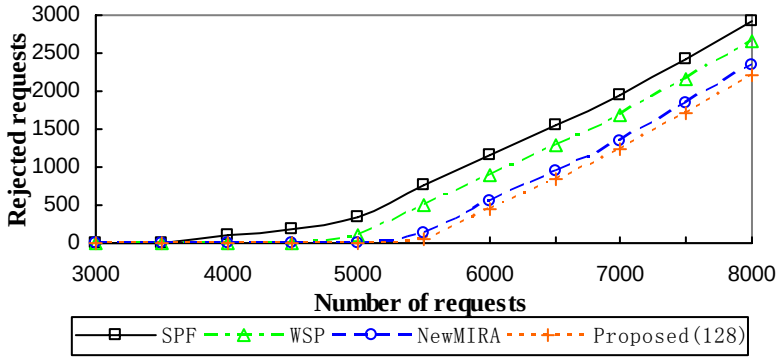

Figure 2.Total rejected requests in KL topology

Figure 3 shows the amount of accepted bandwidth till up to total 8000 requests. After 7000 requests, the network is almost saturated. From the figure we can see that our proposed algorithm also accepts more bandwidth than NewMIRA when the precomputation period is 128. These two algorithms perform better than SPF and WSP.

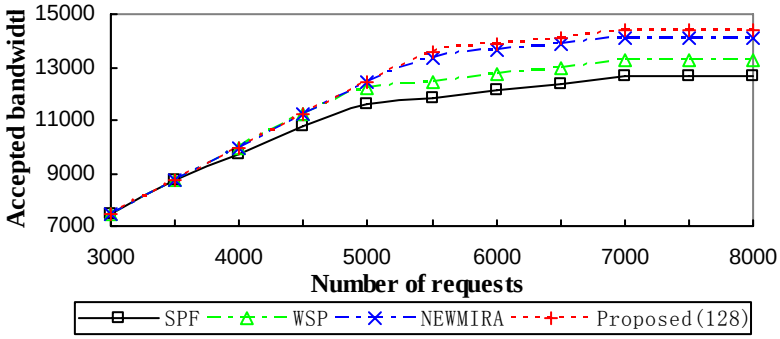

Figure 3. Total accepted bandwidth in KL topology

From table 1, we can found that the total computation time of our proposed algorithm is much less than the NewMIRA algorithm, although a little higher than CSPF algorithm. When the precomputation period increases, the computation time will also decrease. But if the network status and traffic requests surge frequently,

longer precomputation period (i.e. 512, 1024) will leads to worse performance. So there is a compromise to be considered when adjusting the precomputation period.

Table 1.Total computation time till 8000 requests in KL topology

Algorithm	CSPF	WSP	NewMIRA	Proposed(128)
Run time (sec)	0.80	0.80	4.68	1.02

4 Conclusion

In this paper, we proposed a novel precomputation algorithm for traffic engineering routing. Our proposed algorithm considers both the critical and non-critical links. Through effective precomputation, our approach reduces online complexity greatly. Simulation results show that the proposed algorithm performs better than former algorithms. We will try to test the proposed algorithm's performance in more practical scenarios, and investigate that how the change frequency of topology affects the algorithm's performance. We will also try to consider re-routing and multi-path routing with MPLS traffic engineering in the future.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No.90604006.

References

1. M.S. Kodialam, T.V. Lakshman, Minimum Interference Routing with Applications to MPLS Traffic Engineering, in INFOCOM 2000, Tel Aviv, Israel, (2000) 884–893
2. Subhash Suri, Marcel Waldvogel, Priyank Ramesh Warkhede. Profile based Routing: A new Framework for MPLS Traffic Engineering. In: Quality of future Internet Services,Volume 2516 of Lecture Notes in Computer Science; (2001) 138-157.
3. Bin Wang, Xu Su, C. L. Philip Chen. A New Bandwidth Guaranteed Routing Algorithm for MPLS Traffic Engineering. In: IEEE International Conference on Communications 2002(ICC'02); New York, NY, USA; (2002) 1001-1005.
4. János Tapolcai, Péter Fodor, Gábor Rétvári, et al. Class-based minimum interference routing for traffic engineering in optical networks. In: 1st EuroNGI Conference on Next Generation Internet Networks Traffic Engineering; Rome, Italy; (2005) 31-38.
5. Wisitsak Sa-Ngiamsak, Ruttikorn Varakulsiripunth. A Bandwidth-Based Constraint Routing Algorithm for Multi-Protocol LabelSwitching Networks. In: IEEE ICACT 2004; Phoenix Park, Korea; (2004) 933-937.
6. Gábor Rétvári, József J. Bíró, Tibor Cinkler, et al. A precomputation scheme for minimum interference routing: the Least-Critical-Path-First algorithm. In: INFOCOM 2005; Miami, Florida, USA; (2005) 260-268.