ISSN 0866 708X

VIỆN KHOA HỌC VÀ CÔNG NGHỆ VIỆT NAM
VIETNAM ACADEMY OF SCIENCE AND TECHNOLOGY

TẠP CHÍ
KHOA HỌC VÀ CÔNG NGHỆ
JOURNAL OF SCIENCE AND TECHNOLOGY

Tập 50 - Số 2A
2012

TẬP ĐOÀN BƯU CHÍNH VIỄN THÔNG VIỆT NAM
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Chuyên san năm thứ 3

CÁC CÔNG TRÌNH NGHIÊN CỨU VỀ ĐIỆN TỬ, VIỄN THÔNG VÀ CÔNG NGHỆ THÔNG TIN

KỈ NIỆM 15 NĂM
Ngày truyền thống
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
(17/09/1997 - 17/09/2012)

TẠP CHÍ
KHOA HỌC VÀ CÔNG NGHỆ

Tổng biên tập

Nguyễn Xuân Phúc

Viện Khoa học vật liệu, Viện KHCNVN, Hà Nội; phucnx@ims.vast.ac.vn

Phó Tổng biên tập

Nguyễn Dũng, *Viện Cơ học và Tin học ứng dụng, Viện KHCNVN, Tp Hồ Chí Minh; nguyendung@vast-hcm.ac.vn*

Thái Hoàng, *Viện Kỹ thuật nhiệt đới, Viện KHCNVN, Hà Nội; thoang@vnd.vast.ac.vn*

Nguyễn Ngọc San, *Học viện Công nghệ Bưu chính Viễn thông, Hà Nội; khoaqtsdh@hn.vnn.vn*

Uỷ viên Hội đồng biên tập

Lê Huy Bá, *Trường Đại học Công nghiệp Tp Hồ Chí Minh, Tp Hồ Chí Minh; lehuyba@hcm.vnn.vn*

Lê Thanh Bình, *Viện Công nghệ sinh học, Viện KHCNVN, Hà Nội; ltbinh24@yahoo.com*

Phạm Quang Bắc, *Tổng Công ty Dầu khí, Hà Nội*

Nguyễn Hoài Châu, *Viện Công nghệ Môi trường, Viện KHCNVN, Hà Nội; nhchau.cet@gmail.com*

Nguyễn Hữu Dũng, *Trường Đại học Bách khoa Hà Nội, Hà Nội; dung-sdh@mail.hut.edu.vn*

Nguyễn Công Định, *Học viện Kỹ thuật Quân sự, Hà Nội; dinhnc@mta.edu.vn*

Nguyễn Văn Gia, *Viện Cơ học và Tin học ứng dụng, Tp Hồ Chí Minh; nvg40@yahoo.com*

Nguyễn Công Hào, *Viện Công nghệ Hóa học, Viện KHCNVN, Tp Hồ Chí Minh; nguyenconghao@yahoo.com*

Phan Bùi Khôi, *Trường Đại học Bách khoa HN, Hà Nội; phanbuikhoi@mail.hut.edu.vn*

Phạm Hữu Lý, *Viện Hóa học, Viện KHCNVN, Hà Nội, huuly@yahoo.com.vn*

Phạm Quốc Long, *Viện Hóa học Các hợp chất thiên nhiên, Viện KHCNVN, Hà Nội; mar.biochem@fpt.vn*

Trần Đình Long, *Trường Đại học Bách khoa Hà Nội; evnbk@hn.vnn.vn*

Châu Văn Minh, *Viện Hóa sinh Biển, Viện KHCNVN, Hà Nội; cvminh@vast.ac.vn*

Hoàng Minh, *Học viện Công nghệ Bưu chính Viễn thông, Hà Nội; hoangminh@ ptit.edu.vn*

Nguyễn Xuân Nguyên, *Viện Hóa học Các hợp chất thiên nhiên, Viện KHCNVN, Hà Nội; nxnguyen_ctc@yahoo.com*

Lê Văn Nhương, *Trường Đại học Bách khoa Hà Nội. Hà Nội; levannhuongtc@yahoo.com*

Nguyễn Hữu Phương, *Trường ĐH Khoa học tự nhiên, Tp Hồ Chí Minh; nhphuong@hcmuns.edu.vn*

Phạm Văn Thiêm, *Trường Đại học Bách khoa Hà Nội, Hà Nội; thiempv-edc@mail.hut.edu.vn*

Hồ Sĩ Thoảng, *Tổng Công ty Dầu khí, Tp Hồ Chí Minh; hosithoang@gmail.com*

Phan Minh Tân, *Sở Khoa học Công nghệ Tp HCM, Tp Hồ Chí Minh; tanpm@tphcm.gov.vn*

Phan Đình Tuấn, *Trường Đại học Bách khoa Tp HCM, Tp Hồ Chí Minh; pdtuan@hcmut.edu.vn*

Thư kí Toà soạn: **Lê Thị Thiên Hương;** *tc_khcn@yahoo.com*

Địa chỉ: Tòa soạn và Trị sự: Tầng 1, Nhà A16, 18 Hoàng Quốc Việt, Cầu Giấy, Hà Nội;
Điện thoại/Fax: (84)-4-9422825; Email: *v-jst@vjs.ac.vn*, Website: ***http://vjs.ac.vn/index.php/jst***

TẬP ĐOÀN BƯU CHÍNH VIỄN THÔNG VIỆT NAM
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Chuyên san năm thứ 3 Tập 50 - Số 2A/2012

ĐIỆN TỬ, VIỄN THÔNG VÀ CÔNG NGHỆ THÔNG TIN

KỈ NIỆM 15 NĂM Ngày truyền thống
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
(17/09/1997 - 17/09/2012)

Tạp chí Khoa học và Công nghệ 50 (2A) (2012) 75-86

BANDWIDTH GUARANTEED ROUTING ALGORITHMS FOR TRAFFIC ENGINEERING: AN APPLICATION CASE ON MPLS NETWORKS

Cao Thai Phuong Thanh[1,*], **Tran Cong Hung**[2]

[1]*Saigon University, 273 An Duong Vuong Str. 5 Dist. Ho Chi Minh city, Vietnam*
[2]*Post & Telecommunications Institute of Technology, 122 Hoang Quoc Viet, Hanoi, Vietnam*

[*] Email: *ctpthanh@sgu.edu.vn*

Received: 15 June 2012, Accepted for publication: 10 August 2012

ABSTRACT

Due to the fast growths of computer networks, traffic engineering which tries to satisfy both quality of services and resource utilization requirements is an important research area. Among traffic engineering (TE) mechanisms, routing algorithm – a strategy to select paths for traffic – plays a crucial role and there have been many TE routing proposals. This paper presents an application case of those algorithms on MPLS networks by giving their key ideas and mathematical descriptions, and then by various experiments so to analyze the routing performances with different metrics. The paper also discusses the trend of routing algorithms and future research directions.

Keywords: bandwidth guaranteed routing, traffic engineering, MPLS.

1. INTRODUCTION

Besides traditional network services, next generation network applications such as voice over IP, video on demand and web games require certain quality of services (e.g. minimum available bandwidth guaranteed) described as customer service-level agreements (SLAs). Network providers try not only to satisfy those SLAs but also to optimize the network resources for profit reasons. As a result, traffic engineering which is defined as techniques to manage traffic flows through networks with the joint goals of service performance and resource optimization has attracted much attention.

The most important technique to implement TE is routing scheme, which selects routes of data flows satisfying QoS demands. Basically, there are two types of routing algorithms: proactive and reactive [1]. The former uses fixed information to configure routing so the network's operation is simple but not robust to traffic variations; whereas, the latter considers the state of network (i.e. residual bandwidths) when handling requests. As a result the performance of reactive routing network can be optimized at all times though it takes extra cost to monitor the network state. This paper focuses on the reactive dynamic routing algorithms with bandwidth constraint.

On the other hand, Multiprotocol Label Switching (MPLS) [2] is a technology in high speed networks which forwards labelled data packets along Label Switching Paths (LSPs). Those paths are pre-defined using signalling protocols such as Resource Reservation Protocol (RSVP) and Label Distribution Protocol (LDP). This feature of MPLS facilitates TE operations because traffic flows can be set up and managed explicitly. This paper presents an application case of TE routing algorithms on MPLS environments.

Specifically, key ideas of those algorithms are reviewed and, for the first time, their mathematical formulations are presented in the unified manner. Moreover, the performances are compared and analysed by various experiments with metrics of accepted ratio and calculation time. The rest of the paper is organized as follow. In section 2, after definitions and notations, bandwidth guaranteed routing algorithms are classified into three categories: single pair aware, minimum interference and learning machine. Section 3 presents simulated experiments and analysis. Finally, section 4 discusses the evolutional trend of routing algorithms and the future research directions.

2. ROUTING ALGORITHMS

2.1. Problem definition and notations

A network topology with n node and m links is considered. Each link has its own capacity and residual bandwidth at a given time. Traffic demands, which require certain bandwidths from ingress to egress nodes, are handled by the routing algorithm so as to maximize the number of accepted demands. Table 1 lists the mathematical notations.

Table 1. Notations used in formulations

Symbol	*Description*
$G(N,L)$	A direct graph presents the network topology $N(\lvert N\rvert = n)$ is a set of nodes $L(\lvert L\rvert = m)$ is a set of links
$c(l)$	Capacity bandwidth of link l
$r(l)$	Residual bandwidth of link l
$d(s,d,b)$	A traffic demand from ingress node s to egress node d with required bandwidth b
D	A set of all ingress-egress pairs
p_{sd}	A routing path from s to d
P_{sd}	A set of all paths from s to d

The goal of routing algorithms is:

$$\begin{cases} \mathit{Maximize\ number\ of\ satisfied\ demands} \\ \qquad\qquad \mathit{subject\ to} \\ \text{Find } p_{sd} \text{ for } d(s,d,b) \,/\, \forall l \in p_{sd} : r(l) \geq b \end{cases} \tag{1}$$

Since ingress-egress pairs have commodity integral flows (i.e. common links as well as sequences of links), the routing problem summarized in (1) is NP-hard [3]. Moreover, the algorithms can be generalized as table 2.

Table 2. General TE routing algorithm

Input	A network graph $G(N, L)$ with sets of link capacities and residual bandwidths A traffic demand $d(s, d, b)$
Output	A satisfied bandwidth path from s to d, p_{sd}, toward the optimal goal in (1). Or no route satisfying the request
General algorithm	1. Calculate all link weights $w(l)$ 2. Remove links that have residual bandwidth less than b 3. Find the least cost path p_{sd} based on weights of remaining links

2.2. Single pair aware algorithms

The simplest solution is Minimum Hop Algorithm (MHA) where all weights are statically equal to 1. Dijkstra or Bellman-Ford algorithm is applied to find least hop counts paths. It means shortest paths are always selected and their links are quickly congested whereas others underutilize. To improve this property, [4] propose Shortest Widest Path routing (SWP) which chooses paths having maximum bottleneck link bandwidth. If several paths get the same residual bandwidth then hop count is computed. Additionally, Widest Shortest Path algorithm (WSP) is suggested in [5]. Like MHA, WSP find shortest paths first, and the largest residual bandwidth is only considered when there are equal-length routes.

Not restricted to one property, Bandwidth Constrained Routing Algorithm (BCRA) [6] combines three parameters (link capacity, residual bandwidth, and path length) to calculate link weights.

$$w(l) = cost(l) * load(l) + 1$$

$$cost(l) = \frac{10^8}{c(l)}$$

$$load(l) = \frac{c(l) - r(l)}{c(l)}$$

Heavy weight values mean low capacities and/or heavy loads so those links are likely avoided, whereas hop count is reflected by the addition of 1. Experiments confirm that such combination of properties, especially the current link loads, improves the routing performance.

Despite of using different network parameters, the above algorithms are classified as single-pair-aware because they greedily find good routes for the being-demanded ingress-egress pair but not consider other ones. It might noticeably affect future other requests. As a result, minimum interference solutions are proposed.

2.3. Minimum interference algorithms

The first Minimum Interference Routing Algorithm (MIRA) uses the maxflow-mincut characteristic [7]. When a routing request arrives, a maxflow residual graph is computed to determine a mincut set for each other ingress-egress pair. According to the maxflow-mincut theory [8], a bandwidth decrease of a link belonging to the mincut set will lead to the same amount reduction of the corresponding maxflow value. It means if such links are selected to route the current request, they will interfere with future demands of other pairs (i.e. narrow maxflow of them). MIRA defines those links as critical and assign more weights to them by the equation:

$$w_{sd}(l) = \sum_{(s',d')\in D\setminus(s,d)} \alpha_{s'd'} \; ; iflis criticalof(s',d')$$

$$\alpha_{s'd'} \text{ reflects the importance of the pair } (s',d')$$

MIRA computes link weights from all ingress-egress pairs except the current demanded pair. Such current pair is excluded because the algorithm aims to prevent interference with the other ones. When all pairs are treated equally ($\alpha = 1$), a weight is the frequency of link's criticality, and the more critical a link is, the less it is chosen. Evaluations prove that MIRA outperform MHA in term of accepted ratio. Inspired by MIRA, different minimum interference algorithms are proposed.

Authors of NewMIRA [9] comment that MIRA only takes links of mincut sets into account, whereas all links that put up maxflows might affect future demands. Therefore, NewMIRA calculates a link's criticality by its load contribution to maxflow and the residual bandwidth.

$$w_{sd}(l) = \sum_{(s',d')\in D\setminus(s,d)} \frac{f_l^{s'd'}}{\theta^{s'd'} . r(l)}$$

$$\theta^{s'd'} \text{ is the maxflow of the pair } (s'd')$$

$$f_l^{s'd'} \text{ is the subflow of } \theta^{s'd'} \text{ through link } l$$

High interfering links are ones that largely contribute to maxflows and/or have small remaining bandwidths. Similar to MIRA, the NewMIRA also excludes the current pair *(s. d)* from weight calculation.

Dynamic Online Routing Algorithm (DORA) [10] does not use maxflows for interference but use the numbers of time links appearing in disjointed routing paths. Specifically, a link criticality of one *(s, d)* is decreased if that link is a part of any path from *s* to *d*, and increased if it belongs to paths of the other pairs.

$$criticality_{sd}(l) = \sum_{(i,j)\in D} \sum_{p_{ij}\in dP_{ij}} v_l$$

$$v_l = \begin{cases} 0 \; ifl \in p_{ij} \\ -1 \; ifl \in p_{ij} and ij \equiv sd \\ 1 \; ifl \in p_{ij} and ij \not\equiv sd \end{cases}$$

$$dP_{ij} \text{ is the disjointed path set of the pair } (i,j)$$

Realizing that the criticalities are computed solely by the network topology (i.e. by identification of disjointed paths); those values are prior-determined and only recalculated once the topology changes. This calculation is called the offline phase to differ from the online reactive routing phase. When a routing request arrives, weights are formed from the corresponding criticalities and the current bandwidths.

$$w_{sd}(l) = (1-\propto).N_{C_{sd}(l)} + (\propto).N_{RRB(l)}$$

$N_{C_{sd}(l)} \in [0,\ 100]$ is the normalization of the $criticality_{sd}(l)$

$N_{RRB(l)} \in [0, 100]$ is the normalization of the reciprocal of $r(l)$

$\propto\ \in [0,1]$ is the proportion parameter

Because the interference is pre-determined, DORA selects routes more quickly than the above algorithms, especially when there are many ingress-egress pairs.

Similar to DORA, Bandwidth Guarantee with Low Complexity algorithm (BGLC) [11] has two phases. The critical values are directly proportional to the frequencies of links in all possible paths rather than the disjointed paths of just other pairs. In addition, the online phase also involves the residual bandwidths.

$$w(l) = criticality(l).\frac{1}{r(l)}$$

$$criticality(l) = \sum_{(i,j)\in D} \sum_{p_{ij}\in P_{ij}} \frac{v_l}{|P_{ij}|}$$

$$v_l = \begin{cases} 0\ if\ l \in p_{ij} \\ 1\ if\ l \in\ p_{ij} \end{cases}$$

$|P_{ij}|$ is the number of all paths from i to j

Besides the minimum interference ideas, additional routing algorithms are recently proposed in the extent of learning machine applications.

2.4. Learning machine algorithms

Random race – a machine learning technique – is applied in Random Race based Algorithm for TE (RRATE) [12] to improve route computation time. The routing race approach is summarized as follow:

- The offline phase selects k shortest paths for each ingress-egress pair *(s, d)* as racing candidates and initialize a race reward value $x_{i\ sd}$ for the path i of *(s, d)*.
- The online phase includes two stages: learning and post-learning. These stages are conducted separately for each ingress-egress pair.
- In the learning stage, when a demand *d(s, d, b)* arrives, costs of the k selected paths are computed based on the number of critical links and the maximum residual bandwidths. Specifically, critical links are determined by the MIRA's maxflow-mincut definitions (i.e. critical links belong to mincut sets).
- High cost values mean there are more critical links and/or small remaining bandwidths. Therefore, routes are chosen in the increasing order of costs. For example, the smallest cost path is first checked for bandwidth requirement. If all links satisfy the demand then

traffic is routed through that path; otherwise, the second smallest cost path is considered and so on. Additionally, whenever a path is selected, its corresponding is accumulated by 1. The racing of those reward values continues until one of the paths reaches a pre-defined threshold *N*. Then, k paths of the *(s, d)* pair are sorted by the increasing order of their respective rewards. Further requests of *(s, d)* are handled by the post-learning stage.

$$cost(p_{i\_sd}) = k_1 . C_{i\_sd} + {k_2}/{R_{i\_sd}}$$

$C_{i_{sd}}$ is the number of critical links in the path p_{i_sd}

$$C_{i\_sd} = \sum_{l\in p_{i\_sd}} \sum_{(s,d)\in D} v_l$$

$$v_l = \begin{cases} 1\ if\ l\ is\ critical\ of\ (s,d) \\ 0\ if\ l\ is\ not\ critical\ of\ (s,d) \end{cases}$$

$R_{i_{sd}}$ is the maximum remaining bandwidth in p_{i_sd}

if $d(s,d,b)$is routed through p_{i_sd}

$$R_{i\_sd} = max_{l\in p_{i\_sd}}(r(l) - b)$$

k_1 and k_2 are the moderation parameters

- In the post-learning stage, there is no computation but a route is selected within the sorted paths. Particularly, paths are verified against bandwidth requirement in the order of racing positions (i.e. the reward values). If the first route does not satisfy the demand, next ones are inspected. The process repeats until a satisfied route is found or all k paths are checked.
- Both two phases are reset if the network topology changes. However, normal networks do not change frequently so the post-learning stage of RRATE reduces the routing decision time.

Using the same random race technique, Paths Optimal Ordering Algorithm (POOA) [13] modifies RRATE in several aspects.

- The offline phase not only selects k shortest paths for each ingress-egress pair but also computes critical values. A link criticality relates to its subflows constituting the maxflows. Given that a link l belongs to one or more paths from *s* to *d*, the criticality is determined as:

$$criticality(l) = \frac{\sum_{(s,d)\in D} f_l^{sd}}{\sum_{(s,d)\in D} \theta^{sd}}$$

f_l^{sd} is the subflow of link l throw the $maxflow\ \theta^{sd}$

- The learning state (online phase) calculates path costs by the criticalities and the residual bandwidths.

$$cost(p_{i\_sd}) = \sum_{l \in p_{i\_sd}} \frac{criticality(l)}{r(l)}$$

Similar to RRATE, paths with low costs are considered first and the winning route is rewarded. However, the POOA racing threshold is not those reward values but is the whole position orders of k paths. In particular, after each demand is routed, the learning stage sorts k paths by their accumulated rewards. If that order is changed comparing to the order of the last demand, then the race is reset. In other words, the race ends on the condition that the path order continuously remains k times. (The POOA threshold is fixed to k instead of another N value of RRATE.)

- The post-learning state is the same as RRATE. It is noticed that the POOA cost computation is faster than RRATE's because the criticalities are pre-computed. Nevertheless, the later race process may be longer than the former due to the racing mechanism. Experiments will further compare and analyze them.

Another machine learning approach is introduced in the Predicting of Future Load-based Routing algorithm (PFLR) [14]. PFLR predicts future available bandwidth from history routing data by the feed forward neuron network. Those prediction values are then able to incorporate to different metrics to form link weights for the current routing request. For example, PFLR's link weights are combined from the residual and predicted bandwidth.

$$w(l) = \frac{(1-\alpha)}{r(l)} + \frac{\alpha}{predicted\_r(l)}$$

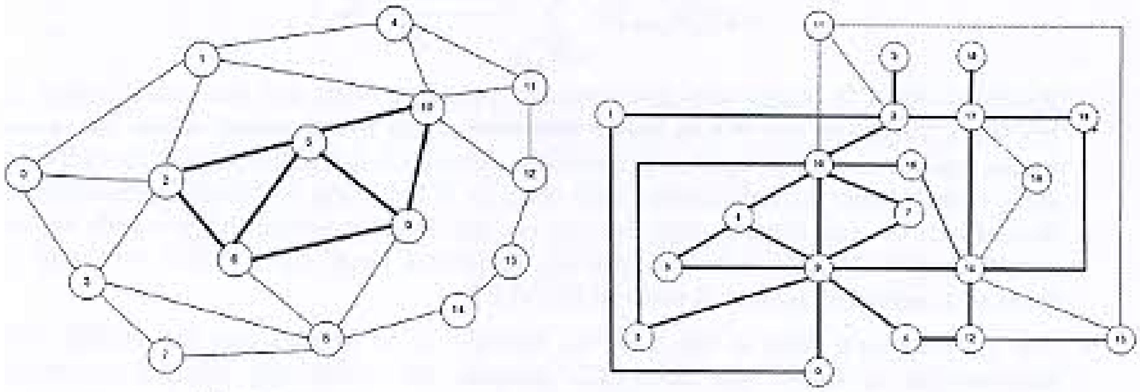
$\alpha$is the prediction proportion parameter

3. EXPERIMENTS & ANALYSIS

3.1. Simulation environment

All the above algorithms except PFLR are implemented in the network simulator NS2 [15, 16]. PFLR is excluded because its prediction approach is meaningfully evaluated only if there is real network routing data or the syndicated data have prior-patterns. Meanwhile, this paper limits to the arbitrary routing data.

Two different network topologies are simulated: one (Figure 1a) is adapted from many previous TE routing works [7, 10, 13], and the other (Figure 1b) inherits the real CESNET MPLS topology [17]. Both networks' links are bidirectional and have two types of capacity. The higher (the thicker links in the figure) is 4800, 10000 and the lower (the thinner ones) is 1200, 1000 bandwidth units respectively.



(a) MIRA topology

(b) CESNET topology

Figure 1. Network topologies

For each topology, three routing scenarios are evaluated. The first scenarios constantly demands 2000 static paths that stay in the network forever after being setup. The second one sets 2000 requests dynamically. Those requests arrive randomly according to the Poisson distribution of mean λ demands per time unit; whereas their holding (routing) time are distributed by the Exponential mean μ time units. The third scenario is mixed between 200 static and 1800 dynamic requests. In this case, the distribution values are λ and μ. Furthermore, four ingress-egress pairs of (0, 12), (4, 8), (3, 1), and (4, 14); and eight pairs of (0, 18), (1, 11), (3, 16), (4, 7), (5, 13), (6, 19) , (15, 0) and (19, 8) are set for MIRA and CESNET topologies respectively. The former network has random bandwidth demands between 10, 20, 30, and 40 units. Meanwhile, the later arbitrarily needs 40, 80, 120, or 160 units for a request.

Two metrics are used to evaluate the algorithms in the aspect of the optimal goal described in (1). Firstly, percentage of accepted requests is compared. Obviously, the higher the accepted percent is, the better an algorithm performs. The second metric is the average of computing time which is counted when a request arrives until it is accepted or rejected. This metric indicates the complexity of the reactive online routing phase, and should be minimized.

3.2. Evaluation results

To obtain confident results, experiments are repeated several times with either different requests or algorithms' parameters. Comments in this paper describe the overall observation although there are few exceptions. Table 3 shows example evaluating values with following parameters.

- For DORA, the bandwidth proportion $\alpha = 0.5$;
- For RRATE, the moderation parameters $k_1 = k_2 = 0.5$, the number of pre-selected path k, and the racing threshold N;
- For POOA, $k = N$.

Among single pair aware algorithms, the traditional ones (MHA, SWP and WSP) accept least number of demands in most experiments, whereas BCRA which considers more network's properties achieves better performances. There is an exception where WSP gains second highest percentage in the example results of mixed request experiment (table 3.c). It might cause by equal-length paths in the network topology. Nevertheless, such result is not the overall trend. On the other hand, in respect to computing time, this category contains the fastest and slowest algorithms which are MHA and SWP respectively. The former is fast due to no additional calculation except the shortest path algorithm. The latter is slow because it finds all possible paths between source and destination nodes.

In addition, interference minimum algorithms attain considerable improvements of accepted percent over shortest path algorithms. Specifically, DORA accepts 7 % higher number of requests than MHA in the static test (table 3.a). Table 3.b also depicts a 10 % improvement of NewMIRA over WSP. However, there is no leading algorithm within the interference minimum category. For example, DORA obtains the highest value (25.35 %) in the static experiment but the lowest one (81.15 %) in the dynamic test. Meanwhile, in this test, NewMIRA outperforms the others with 87.05 % comparing to the second highest of MIRA only at 83.15 %. In the term of route selecting time, there is a significant difference between one-phase and two-phase algorithms. Particularly, the computing time of MIRA is even 10 ten times greater than DORA and BGLC's because MIRA recalculates maxflows for each routing demand whereas the others predetermine link's criticalities and need much less time to form link weights.

Table 3. Comparison of accepted percent (in %) – computing time (in millisecond) subject to number of requests (NoR)

NoR	MHA	WSP	BCRA	MIRA	NewMIRA	DORA	BGLC	RRATE	POOA
100	100-0.13	100-0.18	100-0.27	100-2.04	100-2.07	100-0.29	100-0.36	100-2.06	100-0.46
500	64.00-0.13	64.00-0.16	66.80-0.23	67.00-1.59	67.00-1.53	63.40-0.27	67.00-0.31	67.00-0.75	62.60-0.45
1000	36.20-0.12	37.20-0.14	41.40-0.22	41.50-1.24	41.50-1.14	39.70-0.27	41.50-0.33	38.20-0.44	39.20-0.43
1500	24.13-0.12	24.80-0.14	29.00-0.20	31.53-1.10	28.60-0.95	31.07-0.26	27.87-0.30	25.47-0.34	26.13-0.41
2000	18.10-0.12	18.60-0.13	21.75-0.19	23.65-1.02	21.45-0.86	25.35-0.25	20.90-0.28	19.10-0.29	12.60-0.40

(a) Results of static requests on MIRA network

NoR	MHA	WSP	BCRA	MIRA	NewMIRA	DORA	BGLC	RRATE	POOA
100	100-0.16	100-0.21	100-0.35	100-4.74	100-2.52	100-0.37	100-0.31	100-5.25	100-0.59
500	86.00-0.17	83.80-0.21	84.80-0.33	84.60-4.55	89.60-2.61	85.80-0.37	86.20-0.30	94.00-3.06	88.20-0.45
1000	79.50-0.17	77.20-0.21	81.10-0.33	80.20-4.47	85.90-2.62	80.10-0.37	78.60-0.29	90.80-1.73	85.10-0.42
1500	79.27-0.17	76.73-0.21	82.53-0.33	81.07-4.67	87.13-2.85	80.33-0.36	79.93-0.30	90.40-1.21	86.20-0.40
2000	80.40-0.17	78.35-0.21	82.70-0.33	83.15-4.80	87.05-2.88	81.15-0.36	81.30-0.29	88.10-0.94	86.30-0.38

(b) Results of dynamic requests on CESNET network

NoR	MHA	WSP	BCRA	MIRA	NewMIRA	DORA	BGLC	RRATE	POOA
100	100-0.59	100-0.68	100-0.35	100-5.30	100-2.49	100-0.50	100-0.30	100-5.37	100-0.59
500	77.00-0.25	78.00-0.30	76.80-0.33	76.80-4.67	76.40-2.39	78.40-0.39	76.40-0.29	78.00-3.21	76.00-0.47
1000	70.10-0.21	71.30-0.26	71.70-0.32	70.50-4.45	69.90-2.27	71.60-0.37	70.50-0.28	72.40-2.13	69.80-0.42
1500	66.60-0.20	69.87-0.24	68.20-0.32	67.13-4.39	65.93-2.22	68.20-9.37	68.00-0.28	70.47-1.73	66.13-0.42
2000	65.95-0.19	70.20-0.23	66.65-0.33	66.50-4.37	64.15-2.20	68.05-0.37	69.60-0.28	71.20-1.46	64.90-0.40

(c) Results of mixed requests on CESNET network