Electronic Communications of the EASST
Volume 17 (2009)

Workshops der
Wissenschaftlichen Konferenz
Kommunikation in verteilten Systemen 2009
in Kassel
(WowKiVS 2009)

Prediction-based Decentralized Routing Algorithm

Abutaleb Abdelmohdi Turky, Andreas Mitschele-Thiel

10 Pages

Guest Editors: M. Wagner, D. Hogrefe, K. Geihs, K. David
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Prediction-based Decentralized Routing Algorithm

Abutaleb Abdelmohdi Turky, Andreas Mitschele-Thiel

Integrated HW/SW Systems Group, Ilmenau University of Technology, Ilmenau, Germany
E-mail: {abutaleb-abdelmohdi.turky,mitsch}@tu-ilmenau.de

Abstract: We introduce a new efficient routing algorithm called Prediction-based Decentralized Routing algorithm (PDR), which is based on the Ant Colony Optimization (ACO) meta-heuristics. In our approach, an ant uses a combination of the link state information and the predicted link load instead of the ant's trip time to determine the amount of pheromone to deposit. A Feed Forward Neural Network (FFNN) is used to build adaptive traffic predictors which capture the actual traffic behaviour. We study two performance parameters: the rejection ratio and the percentage of accepted bandwidth under two different network load conditions. We show that our algorithm reduces the rejection ratio of requests and achieves a higher throughput when compared to Shortest Path First and Widest Shortest Path algorithms.

Keywords: Self-organization, neural network, Ant algorithms, routing, traffic engineering

1 Introduction

With the emergence of applications with tight quality of service (QoS) requirements and the reliance of the economy on these applications and the Internet as a whole, network management techniques such as Traffic Engineering (TE) are essential. TE deals with the evaluation and optimization of the traffic capacity and the QoS of networks. Extensions of routing protocols have been proposed to support TE including connectionless routing protocols such as Open Shortest Path First (OSPF). The more recently deployed Multiprotocol Label Switching (MPLS) standard [RVC01] allows a better control for traffic routing and TE.
The efficiency of TE schemes mainly depends on route optimization. Routing algorithms [MCS03] can be classified as static or dynamic depending on the nature of information used for selecting the routes. Static algorithms depend on fixed information which does not change with time; dynamic algorithms use the current state of the network. In addition, routing algorithms can be executed either online or offline according to the point in time when the computation is done. With online routing algorithms, path requests are handled at the time of their arrival. This paper focuses on dynamic online routing.
Dynamic online routing should be decentralized. Therefore, routing decisions should be made based on local state information only and by each network node individually. Most decentralized routing approaches depend on ant agent approaches [SS03]. These algorithms are based on swarm intelligence or Ant Colony Optimization (ACO) [DMC96]. Ant systems are self-organizing, applying the principle of indirect communication between agents employing changes to their environment [KU01]. Ant routing algorithms are inspired from real ants' behaviours which have the ability of discovering the shortest path to a food source and their nest without any knowledge of geometry but with a keen sense of smell. By applying reinforcement learning techniques, the ant routing model can find the almost optimal path between the source and destination through a positive feedback mechanism.

In this paper, we introduce a new efficient routing algorithm called Prediction-based Decentralized Routing algorithm (PDR). The idea of the PDR algorithm is the combination of the link state information and the predicted link load instead of the ant's trip time to determine the amount of pheromone to deposit. A Feed Forward Neural Network (FFNN) is used to accurately predict the actual traffic behaviour and the future load on every network link. The remainder of this paper is organized as follows: section 2 provides an overview on related work. Section 3 describes our Prediction-based Decentralized Routing algorithm. Section 4 discusses the performance of our approach. Conclusions are given in section 5.

2 Related Work

The Shortest Path First (SPF) algorithm is the most commonly used algorithm within MPLS Domains. With SPF, the path that contains the least number of links between the source and destination pair is selected. Although SPF is trying to minimize resource utilization, it leads to congestions in some network links and may cause unbalanced resource utilization. Guerin [GOW97] introduced a modification to the SPF algorithm, called Widest Shortest Path (WSP), which is based on the computation of the shortest paths in the first stage and, in case there is more than one, chooses the one with maximum bandwidth (BW).

Boutaba [BSI02] introduced a dynamic online routing algorithm (DORA) that computes the Path Potential Value (PPV) array associated with a source-destination pair in the first step. PPV is used to avoid routing over links that have a high potential to be part of any other paths. Then the algorithm incorporates the PPV value with the residual link BW to form a weight value for each link that is used to compute a weight-optimized network path. DORA offers better performance than the WSP algorithm. Additionally, computations used in DORA are less expensive. Bagula [BBK04] introduced a Least Interference Optimization Algorithm (LIOA) which reduces the interference among competing flows by balancing the number and BW requirements of flows carried by a link to achieve efficient routing of bandwidth guaranteed paths.

Einhorn and Mitschele-Thiel [EMT08] have applied Reinforcement Learning to a Traffic-Engineering (RLTE) algorithm supporting distributed and self-organized QoS routing. We have applied predictions of the future load to solve the routing problem in [TMT08], predicting the available BW and integrating this in the link weight formula. We have compared the performance of our algorithm, named Predicting of Future Load-based Routing (PFRL) algorithm, with earlier routing approaches like WSP and CSPF algorithms. The PFRL algorithm has reduced the rejection ratio of requests and achieves higher throughput. However, it enhances the performance f routing approaches which work in a centralized manner only.

AntNet [CD98] is an ACO algorithm for distributed routing in IP networks. During the forward phase, each ant constructs a path by taking a sequence of decisions based on a stochastic policy parameterized by local pheromones and heuristic information. Once it arrived at the destination, the backward phase starts. The backward ants retrace the route, followed by their forward ant, and update the local routing information in all the intermediate nodes. Yun and Zincir [YZH04] introduced an adaptive routing algorithm based on the AntNet algorithm. This approach uses a new structure of the routing table to overcome the problem of unrealistic requirements for global information of the original AntNet algorithm.

The Trail Blazer (TB) routing algorithm minimizes the network congestion through local decisions, based on latency measurements collected by scout packets [GS04]. TB is meant to be an extension of existing link-state protocols such as OSPF, which provides shortest-path information to initialize the probability table. Therefore, TB does not require a learning period to discover the network topology. TB is also simpler than the AntNet algorithm. The ant agent-based QoS Multicast Routing Algorithm (QMRA) employs a scheme which uses the probability that a link satisfies some QoS requirements and the cost of a path instead of the ant's trip time or age to determine the amount of pheromone to deposit, so that it has a simpler process and requires less control parameters [YL06].

3 Prediction-based Decentralized Routing Algorithm

This section provides a detailed description of our Prediction-based Decentralized Routing (PDR) algorithm. The algorithm builds on the principles of the TB routing framework. In the TB design, each router has two tables: a link probability table *Pt* and an average transmission delay table *avg* (see Fig. 1). *Pt* contains m rows, one for each destination node. Each row has *K* entries, one for each outgoing link of the router. The entry *pt[d,i]* is the probability of sending a packet to destination *d* on the outgoing link *i*. The table *avg* has *m* entries, one for each destination node. The entry *avg(d)* is the average transmission delay from the current node to the destination d, which is computed from the last M scout packets that arrived from *d*. The scout packet is sent from the source to the destination to explore the network. At every intermediate node, the scout packet selects the outgoing link randomly. When scout packets find their destination, they return to their source on the same path they have arrived on and update their accumulated latency *td* in every intermediate node by *td =td + t(i)*,where *t(i)* is the current latency of the outgoing link *i*. Then, scout packets use the accumulated latency *td* to update the *pt* as follows:

$$f(td) = max(min(avg(d)/td,10),0.1) \quad (1)$$

$$\Delta p = \delta \times f(td) \quad (2)$$

$$pt[d,i] = (pt[d,i] + \Delta p) / (1+ \Delta p) \quad (3)$$

$$pt[d,j]_{j\neq i} = (pt[d,j]) / (1+ \Delta p) \quad (4)$$

The average latency *avg(td)* is used to scale the positive reinforcement value of the scout packet. A larger value of *f(td)* indicates a better (shorter) path. *f(td)* is limited to the range [0.1,10] to prevent wide fluctuations in Δp, which is the reinforcement value of *pt[d,j]*. δ defines the learning rate of the algorithm. All entries in *Pt* of the same destination *d* are scaled by 1+Δp to ensure that their sum remains 1.

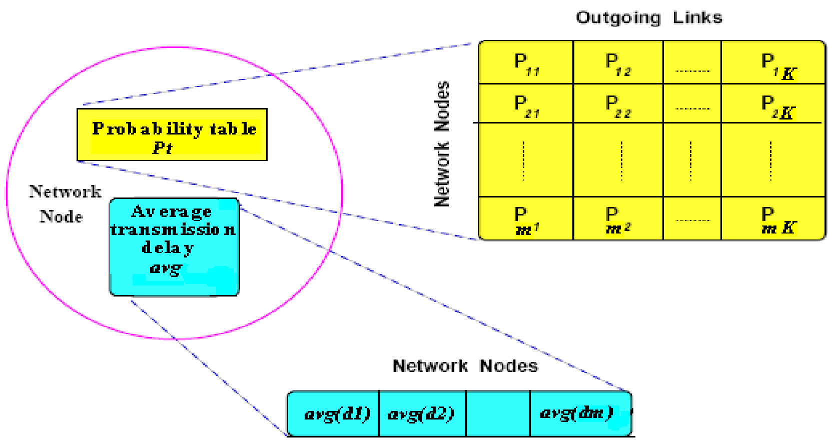

Fig. 1. The node data structures in TB algorithm [GS04].

In our approach, an ant uses a combination of the link state information and the predicted link load instead of the ant’s trip time to determine the amount of pheromone to deposit, so that it has a simpler process and less control parameters. The current latency *t(i)* of an outgoing link *i* in the TB algorithm is replaced by the Link Weight formula *LW(i)*. *LW(i)* represents a combination of PFRL and LIOA to reduce the interference among competing flows by balancing the number and required BW of flows carried by a link to achieve efficient routing.
The LIOA algorithm represents a cost metric which balances the number and the intensity of the flows offered to the routes. In LIOA design, $LW(i) = I^{lc} / (\text{Available BW})^{(1-lc)}$,where I is the number of flows carried on the link and lc is the least interference control parameter which represents a trade-off between the number and the magnitude of the flows traversing a link. On the other hand, the PFRL algorithm proposes to incorporate the future link load value in the link weight formula to optimize the performance of routing. Therefore, we propose to use the *LW(i)* formula as follow:

$$LW(i) = I^{lc}\left(\frac{(1-\alpha)}{(\text{Available BW})^{(1-lc)}} + \frac{\alpha}{(\text{Predicted available BW})^{(1-lc)}}\right) \qquad (5)$$

The *LW(i)* formula is controlled by a parameter called α ,which represents the prediction weight. A low α reduces the influence of the predicted value on the BW. A high value of α increases the influence and suppresses the current value of the available BW.
The idea behind the design of PDR is that the consideration of the future link load will enhance the routing performance of Ant-based routing algorithms. Therefore, we propose to build an accurate traffic predictor that will be able to predict the future traffic behaviour. The future load on network links depends on many parameters such as network topology, network load condition (inter-arrival rate of requests and holding time of these requests in the network) and the behaviour of the used routing algorithm. Artificial Neural networks (ANN) offer prediction capabilities for different types of network traffic and have the ability to be adaptive. Experimental results show that ANNs can accurately predict a complicated network traffic pattern efficiently [ESW06].The predictor expects the available BW on every link after a specified period of time, which is named window size (WS). The prediction should be done in a decentralized fashion . For example, a specific node is responsible for the prediction operations in a part of the network to achieve a fast prediction and to distribute the complexity of the prediction.
The structure of the used FFNN is shown in Fig. 2. It consists of three layers: the input layer contains 16 neurons; the hidden layer contains 20 neurons and only one neuron in the output. The Levenberg-Marquardt [HDB96] training algorithm is used, because it is the fastest and most accurate one in our case. We have tested different FFNN designs and different values of training period time to achieve an efficient predictor. In the training mode, a history of the last thousand (plus WS) link traffic values are used for training purpose. One training pattern contains 16 traffic values from the history in a row as input values and one expected output value. The expected output value is a history value WS time after the input values. By shifting, one thousand training patterns are generated. In the prediction mode, the last 16 traffic values are used as inputs for the FFNN. Then, the FFNN predicts a value for the link load after WS. The predictor stays in the prediction mode for one hundred traffic samples and then returns to the training mode to adapt itself again.

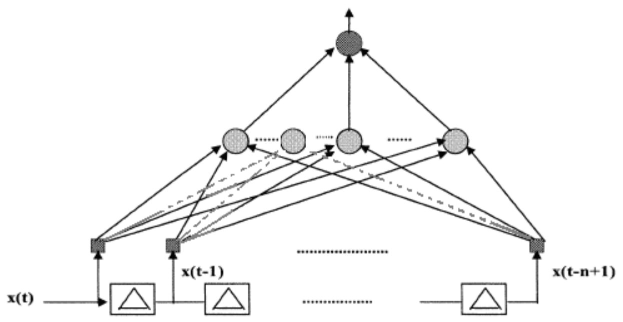

Fig. 2. Feed Forward Neural Network architecture [HDB96].

3. 1 PDR Algorithm

1. At regular intervals WS, predict the available BW for all links in the network.
2. At regular intervals N, each node generates and sends an ant to a destination.
3. When a node receives an ant and, if it is not the destination of the ant, then:
   a. It will forward the ant and selects the next link for the ant´s route randomly.
   b. The ant never selects an outgoing link that leads to a node that has been visited earlier in its path (a loop). If there is no such outgoing link, the ant will die.
4. When the current node is the destination, then, the ant will return to the source on the same path on which it has arrived.
   a. Compute *LW(i)* of outgoing link *i* on every link in the backward path.

$$LW(i) = I^{lc}\left(\frac{(1-\alpha)}{(\text{Available BW})^{(1-lc)}} + \frac{\alpha}{(\text{Predicted available BW})^{(1-lc)}}\right)$$

   where lc is the least interference control parameter, I is the number of flows carried on the link and α is the prediction weight.

   b. Update *pt* and current *avg (d)* values in every intermediate node as follows:

$$\begin{aligned} td &= td + LW(i) \\ f(td) &= \max(\min(avg(d)/td,10),0.1) \\ \Delta p &= \delta \times f(td) \\ pt[d,i] &= (pt[d,i] + \Delta p) / (1+\Delta p) \\ pt[d,j]_{j\neq i} &= (pt[d,j]) / (1+\Delta p) \end{aligned}$$

   where δ is the learning rate of the algorithm.
5. On the other hand, when a node receives a data packet, which needs to be forwarded, data packets will be routed according to the probabilities in *pt*.

4 Performance Evaluation

In this section, we evaluate the performance of PDR, based on some test scenarios and discuss the results. All test scenarios are implemented using Visual Basic and the ANN toolbox of MATLAB [NNT07]. We compare PDR with the SPF and WSP algorithms. We also prove the efficiency of prediction use when we compare PDR with two other modified version of the TB algorithm TB_DORA and TB_LIOA, which were proposed by using only DORA or LIOA

link weight formulas (without prediction) instead of the ant's trip time to determine the amount of pheromone to deposit. For the comparison we use two different network load conditions.
Two performances parameters are studied:

- the rejection ratio of paths requests and
- the percentage of accepted BW.

Our study is done for the MIRA [KKL00] network that is shown in Fig. 3.The thicker links have a capacity of 4800 capacity units while the thinner links have a capacity of 1200 capacity unit. Fig. 3 shows the location of four different source–destination pairs that are identified by (S1, D1), (S2, D2), (S3, D3), and (S4, D4). The capacities of the requested paths are randomly distributed among 10-40 capacity units. We examine the performance for two different network load conditions. The first network load condition is a moderate load condition: when the arrival of requests follows a Poisson distribution with mean $\lambda = 32.5$ requests per time-unit and the holding time of a request is based on an Exponential distribution with mean $\mu = 10$ time-units. The second is a heavy load condition: when the inter arrival of path requests λ is equal to 35 requests per time-unit and the holding time of request μ is equal to 10 time-units.

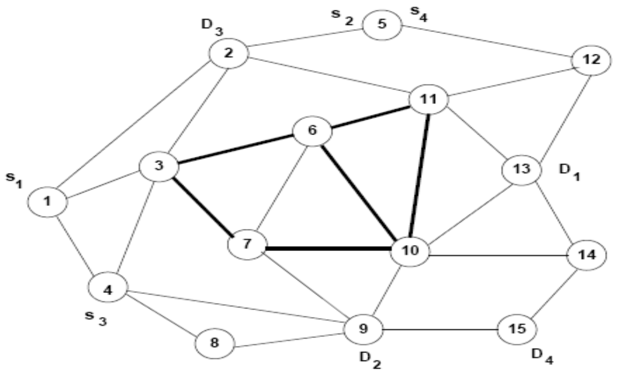

Fig. 3. MIRA network topology [KKL00].

Table 1 describes the PDR parameters and shows the range and used value in our simulation.

Name	Range	Value	Description
lc	[0,1]	0.1	least interference control parameter
N	[10,50]	20	send a scout packet every so many regular packets
M	[10,50]	20	keep the average of the last M of *td* for every destination
δ	[0,0.1]	0.01	learning rate
α	[0,1]	0.1	prediction weight
WS	[5,10]	8 time units	window size

Table 1. PDR parameters

4.1 Moderate Load Scenario

In all experiments, the X axis represents the number of requests. Fig. 4 shows the rejection ratio of requests for moderate load. Based on the result, PDR rejects the fewest number of requests, followed by TB_DORA, TB_LIOA, WSP, and finally SPF. PDR rejects approximately 3.59 % less requests than SPF and 0.44 % less than TB_DORA.

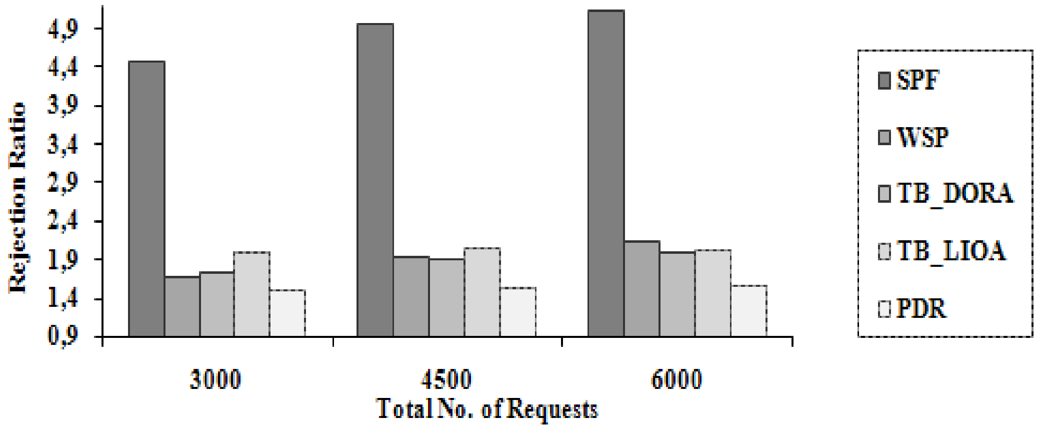

Fig. 4. The rejection ratio of requests for moderate load.

Fig. 5 shows the percentage of accepted bandwidth for moderate load. PDR accepts more BW compared to the other algorithms, followed by TB_DORA, TB_LIOA, WSP, and finally SPF. PDR accepts approximately 4.18 % more bandwidth than SPF and 0.36 % more than TB_DORA.

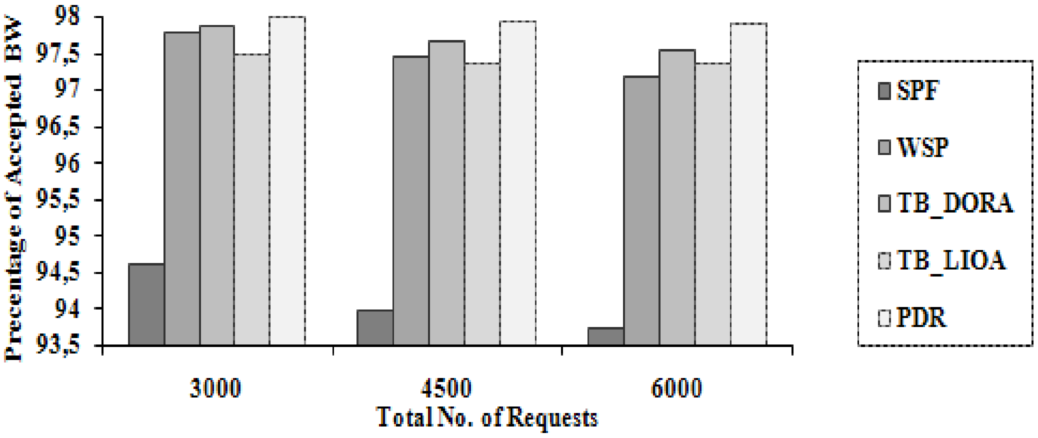

Fig. 5. The percentage of accepted BW for moderate load.

4.2 Heavy Load Scenario

Fig. 6 shows the rejection ratio of requests for heavy load. PDR rejects the fewest number of requests, followed by TB_LIOA, TB_DORA, WSP, and finally SPF. PDR rejects approximately 2.82 % less requests than SPF and 0.30 % less than TB_LIOA.

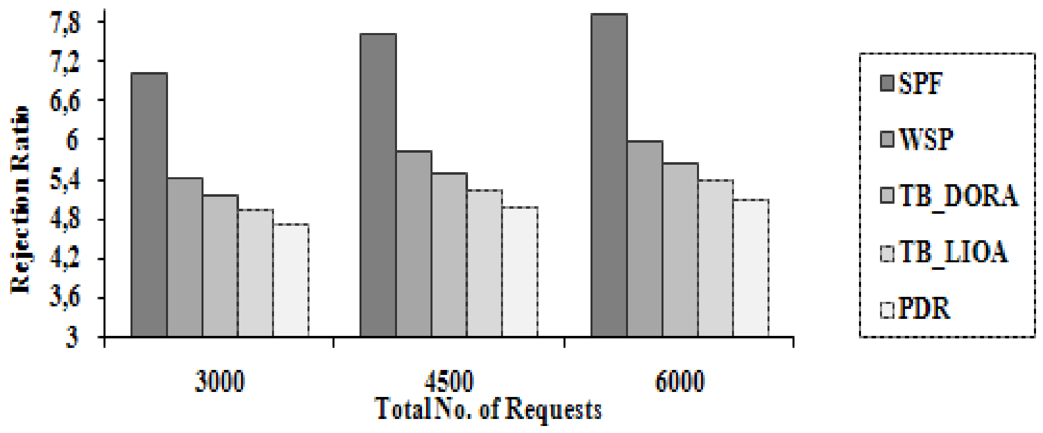

Fig. 6. The rejection ratio of requests for heavy load.

Fig. 7 shows the percentage of accepted bandwidth for heavy load. PDR accepts more BW compared to the other algorithms, followed by, TB_LIOA, TB_DORA, WSP, and finally SPF. PDR accepts approximately 3.09 % more bandwidth than SPF and 0.49 % more than TB_LIOA.

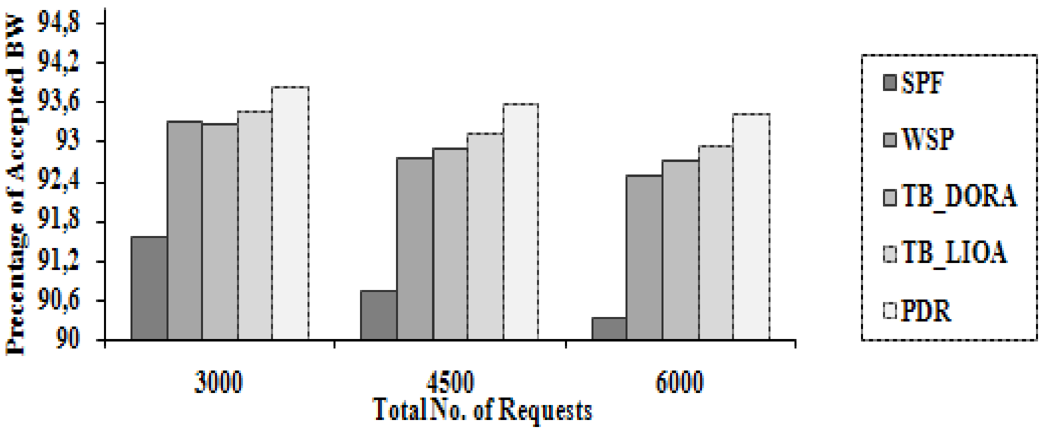

Fig. 7. The percentage of accepted BW for heavy load.

4.3 Complexity Analysis of Prediction Use

The PDR algorithm requires additional computational time to achieve enhanced routing performance. This time consists of two parts, the training time and the prediction time of the predictors. As mentioned before, the predictors are distributed on the nodes. Therefore, every node is responsible for (E/V) operation, whereas E is the number of links and V is the number of nodes. The training operation happens only one time every retraining period RP. The prediction also happens every WS period. Thus, the training requires O((E Tt) / (V RP)) time steps whereas Tt is the training time of one predictor and the prediction requires O((E Pt) / (V WS)) whereas Pt is the prediction time of one predictor. In our experiments, the used processor speed is 1,8 GHZ and 1 GB RAM. The prediction is computationally inexpensive which is equal to 0.006 sec. But the training requires more time which is equal to 0.078 sec.

5 Conclusion and Future Work

In this paper, we have proposed a new TE algorithm named PDR that can efficiently enhance the dynamic on-line routing performance. This algorithm is a member of a class of traffic-aware routing algorithms based on the behaviour of ants. We have compared the performance of the PDR algorithm with WSP and SPF under two different network load conditions and have shown that the PDR algorithm performs considerably better than WSP and SPF with respect to two performance comparison criteria: the rejection ratio of paths requests and the percentage of accepted BW.
We have also proved the efficiency of prediction, comparing PDR with two other modified versions of the TB algorithm without prediction.
As future work, we plan to test the PDR performance in a more complex network topology. Also we will test the PDR performance in link failure scenarios. In addition, a comparison of PDR with other Ant algorithms is planned.

References

[RVC01] E. Rosen, A. Viswanathan, R. Callon: Multiprotocol Label Switching Architecture. RFC 3031, Network Working Group, 2001.

[MCS03] J.L. Marzo, E. Calle, C. Scoglio, T. Anjah: QoS Online Routing and MPLS Multilevel Protection: A Survey. In IEEE Com,Mag. Vol. 41, Iss. 10, pp. 126—132, 2003.

[SS03] K.M. Sim , W.H. Sun: Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions. IEEE Transactions on Systems, Man and Cybernetics, Part A, Vol. 33,No. 5,pp. 560-572, 2003.

[DMC96] M. Dorigo, V. Maniezzo, A. Colorni. The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1):29–41, 1996.

[KU01] Daniel R. Kunkle : Self-organizing Computation and Information Systems: Ant Systems and Algorithms ,technical report, Rochester Inst. of Technology ,2001

[GOW97] R. Guerin, A. Orda, D. Williams: QoS routing mechanisms and OSPF extensions. In IEEE Global Telecommunication, pp. 1903—1908, IEEE Press, 1997.

[BSI02] R. Boutaba, W. Szeto, Y. Iraqi, “DORA: Efficient Routing for MPLS Traffic Engineering. In J. Net. & Sys. Man., vol. 10, no. 3, pp. 309--325, 2002.

[BBK04] A.B. Bagula, M. Botha, A.E. Krzesinski: Online Traffic Engineering: The Least Interference Optimization Algorithm, Proc. ICC '04, pp1232- 1236, 2004.

[EMT08] E. Einhorn, A. Mitschele-Thiel: RLTE: Reinforcement Learning for Traffic-Engineering. In 2nd International Conference on Autonomous Infrastructure, Management and Security (AIMS 2008), LNCS 5127, 2008.

[TMT08] A. A. Turky, A. Mitschele-Thiel: MPLS Online Routing Optimization Using Prediction. Proc. NET-COOP '08, 2008.

[CD98] G.D. Caro, M. Dorigo: AntNet: Distributed Stigmergetic Control for Communications Networks, Journal of Artificial Intelligence Research, 9, pp 317-365, 1998.

[YZH04] H. Yun, A. N. Zincir-Heywood: Intelligent Ants for Adaptive Network Routing. In Proc. CNSR’04, pp 255--261, 2004.

[GS04] E. Gabber, M. A. Smith: Trail Blazer: A Routing Algorithm Inspired by Ants. In Proc.

ICNP’04, pp 36--47, 2004.

[YL06] X. Yan , L. Li: Ant Agent-Based QoS Multicast Routing in Networks with Imprecise State Information. In Proc. PRIMA'06, pp. 374 – 385, 2006.

[ESW06] A. Eswaradass, X.H. Sun, M. Wu, “Network Bandwidth Predictor (NBP): A System for Online Network performance Forecasting”, In Sixth IEEE International Symposium on Cluster Computing and the Grid, pp. 265--268. IEEE Computer Society, 2006.

[NNT07] Neural Network Toolbox, MATLAP V.7, http://www.mathworks.com /products/neuralnet.

[KKL00] K. Kar, M. Kodialam, T.V. Lakshman, “Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications”, In IEEE J. Selected Areas in Comm., vol. 18 vol., 2000.

[HDB96] M.T. Hagan, H.B. Demuth, M.H. Beale, Neural Network Design, Boston, MA: PWS Publishing, 1996.