DYNAMIC WEIGHT MAPPING ADAPTIVE ROUTING (DWMAR) FOR IP NETWORKS

by

FENG XIA

B. Eng., Southeast University, Nanjing, P. R. China, 1998

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

THE UNIVERSITY OF BRITISH COLUMBIA

December 2004

Abstract

This thesis addresses the difficulties of both Shortest Path First (SPF) and traditional adaptive routing protocols. SPF does not respond to the link flow change so that it cannot find optimized routes for the network. Traditional adaptive routing protocols do respond to link load changes. However, the network flow varies so widely that a fixed global weight mapping function cannot be suitable for all the scenarios. That is why traditional adaptive routing protocols are prone to being unstable and counteract the benefits obtained from the link weight adaptation.

Dynamic Weight Mapping Adaptive Routing (DWMAR) is then proposed to provide an optimized and stable adaptive routing protocol. DWMAR proposes an online dynamic weight mapping function which is customized for every link, so that more links will work under an efficient traffic load and have more chances to reach a stable state. To make up for the unstable nature of adaptive routing, DWMAR adopts different adaptive policies on different link loads. Theoretically, DWMAR is a stable algorithm.

Finally, the performance of DWMAR is tested in simulations. The results are compared to the non-adaptive SPF algorithm and a fixed weight mapping adaptive routing, Load Sensitive Adaptive Routing (LSAR). DWMAR shows better performance than the other two algorithms in terms of network efficiency. DWMAR also shows its stability in the simulation results.

Table of Contents

Abstract ii
Table of Contents iii
List of Tables v
List of Figures vi
Acknowledgement vii
1. INTRODUCTION 1
1.1. Introduction to Internet Routing 1
1.1.1. The OSI Reference Model 1
1.1.2. Internet Routing 2
1.1.3. IP Routing Protocols 4
1.1.3.1. Intradomain and Interdomain 4
1.1.3.2. Distance-Vector and Link-State Protocols 6
1.2. Motivation and Objective 7
1.2.1. Limitations of Current Link-State Routing Protocols 8
1.2.2. Related Work 8
1.2.3. Objective 9
1.3. Thesis Outline 9
2. A REVIEW OF ADAPTIVE ROUTING 10
2.1. The Commencement of Adaptive Routing 10
2.2. Minimal Delay Adaptive Routing 10
2.3. Load Sensitive Adaptive Routing 13
2.4. Summary 14
3. DYNAMIC WEIGHT MAPPING ADAPTIVE ROUTING 15
3.1. Assumptions in DWMAR 15
3.2. Stability Analysis in DWMAR 15
3.3. Sectioned Linear Weight Mapping Function 16
3.4. Dynamic Weight Mapping Functions 18
3.5. DWMAR Algorithm 20
4. SIMULATION 23
4.1. Simplification of Parameter Settings 23
4.1.1. Fixing w_{max} 23
4.1.2. Fixing u_1 26
4.1.3. Parameter Settings in Simulation 28
4.2. Static Traffic 28
4.2.1. The Performance of DWMAR 30
4.2.2. The Stability of DWMAR 34
4.3. Dynamic Traffic 38
4.4. Summary 42
5. CONCLUSIONS 44

6. FUTURE WORK 46
7. REFERENCES 47

List of Tables

Table 1.1 Distance Vector and Link-State Protocols....6
Table 4.1 Parameters in DWMAR Simulation....28
Table 4.2 Network Topologies Used in Static Traffic Simulation....29
Table 4.3 Generating Files of Topologies Used in Static Traffic Simulation....30
Table 4.4 Traffic Patterns for Simulated Applications used in Dynamic Traffic Simulation....38
Table 4.5 Session Percentage in Dynamic Traffic Simulation....39
Table 4.6 Simulation Results of Dynamic Traffic for SPF and DWMAR....40
Table 4.7 Simulation Results of Dynamic Traffic for SPF and LSAR....41

List of Figures

Figure 1.1 OSI Reference Model 2
Figure 1.2 IP Packet Header Format 3
Figure 1.3 Intradomain and Interdomain routing 5
Figure 2.1 Instability of Minimal Delay Adaptive Routing 11
Figure 2.2 The System Block Diagram of Minimal Delay Adaptive Routing. 12
Figure 2.3 Two Types of Equilibrium Points of Adaptive Routing. 13
Figure 2.4 The Weight Mapping Function of LSAR. 14
Figure 3.1 A Simplified Scenario of Adaptive Routing in Large Scale Network 16
Figure 3.2 Weight Mapping Function in DWMAR 17
Figure 3.3 Decide the Adaptive Range of Weight Mapping Function 19
Figure 3.4 Status Machine of DWMAR 21
Figure 4.1 Impacts of Varying w_{max} 25
Figure 4.2 Impacts of Varying u_1 27
Figure 4.3 Topology 1 (16 Nodes, 27 Links) 29
Figure 4.4 Simulation Results of Static Traffic on Topology 1 (16 Nodes, 27 Links) 30
Figure 4.5 Simulation Results of Static Traffic on Topology 2 (42 Nodes, 91 Links) 31
Figure 4.6 Simulation Results of Static Traffic on Topology 3 (63 Nodes, 155 Links) 31
Figure 4.7 Simulation Results of Static Traffic on Topology 4 (63 Nodes, 157 Links) 32
Figure 4.8 Simulation Results of Static Traffic on Topology 5 (90 Nodes, 194 Links) 32
Figure 4.9 Simulation Results of Static Traffic on Topology 6 (90 Nodes, 211 Links) 33
Figure 4.10 Convergence Time for Topology 1 34
Figure 4.11 Convergence Time for Topology 2 34
Figure 4.12 Convergence Time for Topology 3 35
Figure 4.13 Convergence Time for Topology 4 35
Figure 4.14 Convergence Time for Topology 5 35
Figure 4.15 Convergence Time for Topology 6 35
Figure 4.16 The Relationships between Performance and Stability 36

Acknowledgement

I would like to thank my thesis supervisor, Dr. Mabo R. Ito, for his guidance throughout this thesis. His instructions showed me the right way to the success, and his suggestions accelerate the completion of this work. Without his guidance and support, this work could not have been accomplished.

I would also like to thank Ms. Jane Pavelich. She helped me a lot to improve the writing style of my thesis.

Finally, I would like to thank my girl friend, Maggie, and my parents, for their love and encouragement.

Feng Xia

The University of British Columbia

2004

1. INTRODUCTION

Adaptive routing commenced earlier than static weight routing on the Internet, but it was proved to be unstable[1][4], which discouraged the research of adaptive routing. Currently, the routing protocols running on the Internet are all static weight routing protocols. However, as the Internet grows, the deficiency of static weight routing protocols increasingly appears. People are beginning to look back at adaptive routing.

1.1. Introduction to Internet Routing

Routing is a key function of a network. An optimized routing will maximize the network proficiency, while an inappropriate routing will reduce the network performance and waste the network resources. Internet engineers always work on finding an algorithm which can maximize the usage of network resources.

1.1.1. The OSI Reference Model

The Open System Interconnection (OSI) reference model was established in 1984. From then on, it serves as one of the most basic, yet essential, elements of computer networking. OSI is an abstract model and offers a very practical structured introduction to many networking concepts.

OSI models the host-to-host networking as a vertically layered stack, which contains 7 layers as shown in Figure 1.1.

Layer 7	Application Layer
Layer 6	Presentation Layer
Layer 5	Session Layer
Layer 4	Transport Layer
Layer 3	Network Layer
Layer 2	Data Link Layer
Layer 1	Physical Layer

Figure 1.1 OSI Reference Model

The application, presentation and session layers are defined as "upper layers", and the others are "lower layers". Upper layers perform application-specific functions such as data formatting, encryption, and connection management. Lower layers provide more primitive network-specific functions like routing, addressing, and flow control.

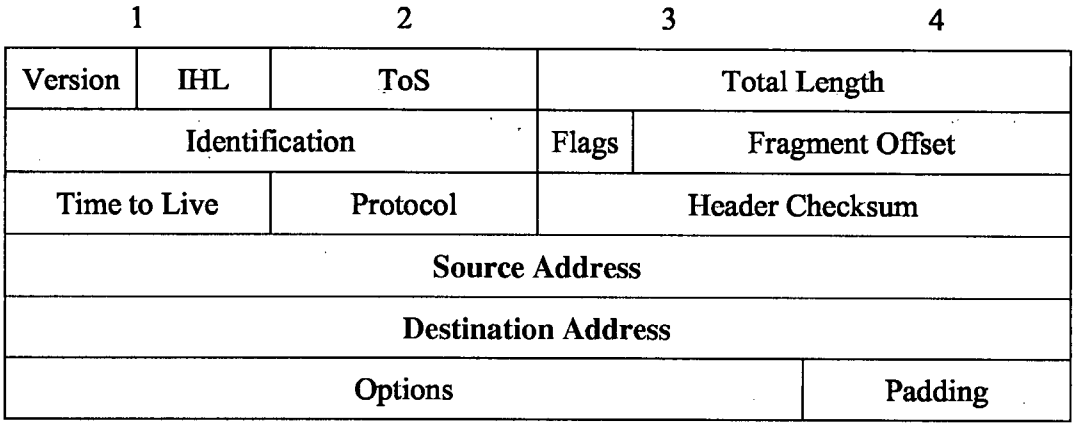
1.1.2. Internet Routing

Currently, the Internet Protocol (IP)[30] is the dominant network layer protocol. IP emerged in the early 1980's as part of the Transmission Control Protocol[31] / Internet Protocol (TCP/IP) protocol suite. The TCP/IP protocol suite comprises many protocols, such as IP, TCP, User Datagram Protocol (UDP)[32] and Internet Message Control Protocol (ICMP)[33].

TCP and UDP are transport layer protocols. They run over IP and provide guaranteed (for TCP) or best effort (for UDP) transport service for applications. Most of the Internet application protocols run over TCP, such as Hyper Text Transfer Protocol (HTTP)[34], File Transfer Protocol (FTP)[29], and Telnet.

IP is successful because of its simplicity and flexibility. The IP network is a connectionless network, which means no prior connection setup is required for both of

the communication ends. In an IP network, data is presented in self-contained routable units known as datagrams or packets.



IHL: IP Header Length
ToS: Type of Service

Figure 1.2 IP Packet Header Format

Each node (end host or intermediate router) in the IP network is assigned one or more addresses. Each IP packet contains the address that it originated from and the address it is destined to. Routers use the destination addresses contained in the packets to make routing decisions and to forward them.

However, IP is a best effort protocol. It does not guarantee that the packet will eventually be delivered to the destination. Thus, TCP was designed to cooperate with IP to provide guaranteed transfer. TCP establishes a virtual connection between source and destination before transferring data between them. TCP acknowledges the reception and retransmits the lost data to make sure that the data reach their destination. TCP also provides some congestion control algorithms to prevent the source from sending too many data which will exceed the network capacity.

A router is a network device that interconnects several networks. Topology information is exchanged among routers. Routing decisions are made based on the topology information

according to certain algorithms, such as the Bellman-Ford Algorithm and Shortest Path First Algorithm. Routers calculate the best path to the destination and forward the arriving packets to the next hop on the best path.

1.1.3. IP Routing Protocols

Routing information can be manually set, which is referred to as static routing. However, this method is inefficient and practically impossible in a large scale network. Most networks run dynamic routing protocols to automatically collect routing information and make routing decisions, which are referred to as dynamic routing. All the routing mentioned later refer to dynamic routing.

Currently, several commonly used IP routing protocols are Routing Information Protocol (RIP) version 1[17], RIP version 2[18], Open Shortest Path First (OSPF)[16], Integrated Intermediate System-to-Intermediate System (IS-IS)[21], Interior Gateway Routing Protocol (IGRP)[19], Enhanced IGRP (EIGRP)[20], and the Border Gateway Protocol (BGP)[22]. These routing protocols can be classified from different points of view: intradomain routing versus interdomain routing, and distance-vector protocols versus link-state protocols. The different breeds of routing protocols have different capabilities related to both architectural design and embedded functionality.

1.1.3.1. Intradomain and Interdomain

A network domain, or an autonomous system (AS), refers to a network of interconnected routers and related systems managed and maintained together by a common administration. All these globally spanned interconnected network domains constitute the Internet. Routing protocols are designed to acquire the optimized routes within a domain or among domains, which are referred to as intradomain routing and interdomain routing, respectively.

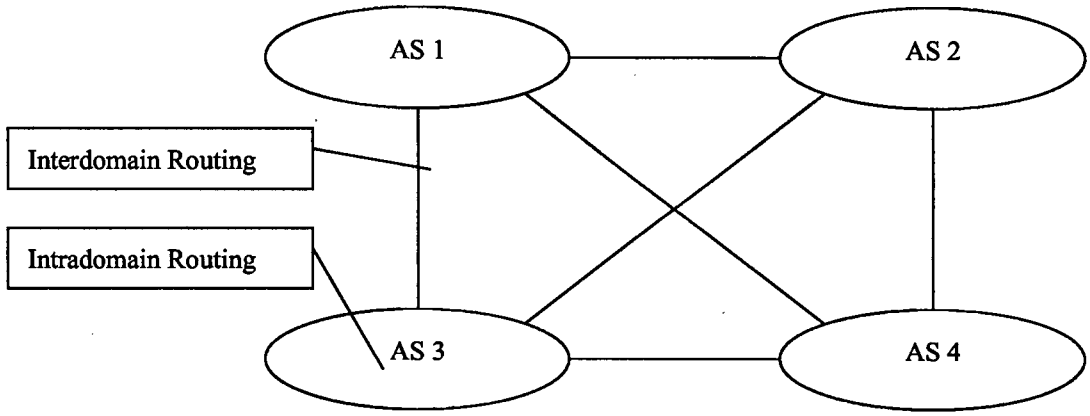


Figure 1.3 Intradomain and Interdomain routing

Figure 1.3 depicts the instances of intradomain and interdomain routing in a routing system comprising of four interconnected network domains, AS1 AS2, AS3 and AS4.

All the routing protocols mentioned in Section 1.1.3 are intradomain routing protocols, except BGP. Different domains are owned and administrated by different owners. Interdomain routing is more complicated than intradomain routing, because it is not only a technical problem, but also related to the domain runners' policies. BGP is the most commonly used dynamic interdomain routing protocol. Instead of finding the best route from source to a destination, it focuses mainly on the aspect of policy implementation among the different ASs. The BGP routing decision is based on hop counting, which is not efficient. It does not have a congestion control mechanism, either, because the BGP router only changes the connectivity information. BGP is also unstable [11]. Because of the deficiencies of BGP, the performance of current Internet routing is far from optimized.

The optimized solution of interdomain routing problems is not a purely technological problem. In this paper, we will only focus on intradomain routing optimization. There are still some technical improvements that can be done on intradomain routing.

1.1.3.2. Distance-Vector and Link-State Protocols

Currently surviving intradomain routing protocols can be classified by architectural design: distance vector protocols and link-state protocols. The classification of those intradomain routing protocols is shown in Table 1.3. The major difference between these two types is the way they discover and calculate routes to destinations

Table 1.1 Distance Vector and Link-State Protocols

Protocol	Category	Metric	Algorithm
RIP v1 and v2	Distance Vector	Hop Count	Bellman-Ford
OSPF	Link State	Bandwidth-based cost	Shortest Path First
IS-IS	Link State	Manual Cost	Shortest Path First
IGRP	Distance Vector	Composite	Bellman-Ford
EIGRP	Distance Vector	Composite	Diffusing Update

The Bellman-Ford algorithm is used in distance-vector routing. A router running distance vector protocol periodically propagates a copy of its routing table to its immediate neighbors. Each recipient router updates its own distance vectors in its routing table according to the information received and forwards the routing table to its immediate neighbors. This process occurs in an omni directional manner among immediately neighboring routers. Each router in the algorithm only has the knowledge of the distances to networked resources of its immediate neighbors. It does not know anything specific about other routers, or the network's actual topology.

Each router running link-state routing protocols maintains a database containing the complete network topology. The link-state routers have full knowledge of network topology. They know the existence of all other routers and how these routers are interconnected. Link-state protocol achieves this by broadcasting link-state advertisements (LASs) within the network. Routers running link-state routing protocols originate LSA to describe the link states, such as the link cost, of all the links to which they directly connect. They also relay the received LSAs to their directly connected neighbors except those from which the LSAs are received. Eventually, all the routers in the network will acquire the topology of the network. After the router gets the network topology, it calculates the route using the Shortest Path First (SPF) algorithm. Currently, the most popular SPF algorithm is the Dijkstra Algorithm, which is implemented in OSPF and IS-IS.

Distance-vector protocols are simpler to design. However, their periodic routing table update consumes a lot of link bandwidth. Furthermore, several other problems are inherent in the distance-vector protocols, such as transient routing loops, count to infinity, and slow convergence. On the other hand, link-state protocols only send incremental updates for any network changes. Generally speaking, link-state protocols have better performance in finding the optimized routers and converge faster than distance-vector protocols. Although link-state protocols consume more of the router's processing and memory resources than distance-vector protocols, these minor shortcomings have not prevented link-state protocols from being the dominant routing protocols running currently on the Internet.

1.2. Motivation and Objective

With the development of the Internet, traffic pattern changes so quickly that static weight routing is not efficient anymore. The objective of this thesis is to find a scheme to stabilize the adaptive routing while keeping the benefit of adaptation.

1.2.1. Limitations of Current Link-State Routing Protocols

Currently, OSPFs are widely used in most Internet Service Provider (ISP) backbone networks. IS-IS exists in some ISP because of historical reasons and often acts as OSPF's backup. However, now OSPF is the de facto industrial standard. The weights, or the lengths, of the links in an OSPF/IS-IS network are set by the network administrator and usually are not changed during the network operation. For example, in most OSPF implementations, the default link weight W is set to $1*10^8 / BW$, where BW is the bandwidth of the corresponding link in terms of bit per second (bps), and in IS-IS, the weights of all links are set to the same default value of 10. The SPF algorithm calculates the routes based on the fixed links weight. Since the link weights are fixed, the path for any source and destination pair is fixed, too. OSPF and IS-IS route the traffic regardless of the load of the link. This feature makes it difficult to do traffic engineering in OSPF and IS-IS. Although OSPF and IS-IS can cope with some network changes such as link up and down, they do not have the ability to adapt to the dynamic traffic running over the network.

1.2.2. Related Work

Multi-Protocol Label Switching (MPLS)[25] has a flexible frame and can use any routing protocol and any traffic engineering scheme. MPLS establishes a virtual connection between any source destination pair and tries to turn the connectionless IP network into a connect oriented network, which by nature are easy to implement traffic engineering. The key point in MPLS is the Label Distribution Protocol (LDP)[26]. LDP acts as the routing protocol in an IP network. Currently, to distribute labels effectively, MPLS needs to know too many network details which are neither compatible with current IP network nor economically efficient. MPLS is developing, but the deployment is limited, let alone tested. Some other protocols/technologies with a similar core concept, such as RSVP[27], Traffic Engineering (TE)[28], also have the same difficulties.

Another way of traffic engineering is to optimize the link weights in OSPF/IS-IS networks[1][6][9]. Given a network and the traffic on it, properly setting the link weights will significantly improve the throughput and performance of the network compared to the default protocol weight settings mentioned in 1.2.1. The limitation of this method is that it is an offline algorithm. It needs to collect the network topology and traffic information before calculating the optimized weight setting, i.e., the resulting weight setting is only optimized for a particular topology and particular traffic pattern. However, in real networks, the traffic load is very dynamic. This method is difficult to implement in real networks.

1.2.3. Objective

The objective of this paper is to design a real-time scheme to adaptively set the weight mapping function of each link in OSPF/IS-IS network which is running adaptive SPF routing protocols. The performance of this scheme will be evaluated by simulation and compared with both non-adaptive SPF routing and fixed weight mapping function adaptive routing.

1.3. Thesis Outline

Chapter 2 gives a brief historical review about adaptive routing. Chapter 3 proposes Dynamic Weight Mapping Adaptive Routing (DWMAR), including analysis, algorithm and implementation details. Chapter 4 compares the simulation performance of DWMAR with other two routing algorithms: one is non-adaptive routing, the other is adaptive routing. Finally, Chapter 5 draw conclusions and Chapter 6 discusses future work.

2. A REVIEW OF ADAPTIVE ROUTING

Adaptive routing is routing that is automatically adjusted to compensate for network changes such as traffic patterns, channel availability, or equipment failures.

2.1. The Commencement of Adaptive Routing

Although almost all the current routing protocols use static weight settings, adaptive routing commenced even earlier than those protocols. As early as the 1960's, the ARPANET routing was truly adaptive. The delay of link was calculated and the link weight was associated with the delay. First the Bellman-Ford, and then the SPF algorithm was used to calculate the path of each source destination pair. Since the link weight was associated with the link delay, the early adaptive routing in APRANET was Minimal Delay Adaptive Routing (MDAR).

2.2. Minimal Delay Adaptive Routing

However, MDAR was prone to severe routing oscillations under a high volume of traffic. Bertsekas gave some theoretical analysis on the instability of MDAR[4]. Ramakrishnan and Rodrigues also illustrate the unstable nature of MDAR[1]. Since a similar analysis method is used in the proposed scheme, a brief review of Ramakrishnan and Rodrigues' research is given below.

For the sake of simplicity, assume that there is only one link adapting its weight in a MDAR network while the weights of the remaining links are kept constant. Also, assume the network traffic flow is static. The dynamic behavior of this adaptive link under a relatively heavy traffic load is shown in Figure 2.1.

Plot the link delay curve and the traffic load curve of the adaptive link on the same coordinate system. The horizontal coordinate is link utilization, and the vertical

coordinates are link delay and traffic load, respectively. Since the link weight is directly associated with link delay, the link delay curve equals the weight mapping curve. When the link utilization increases, the queuing delay increases, i.e. the weight of the link increases. At the same time, as the link weight increases, the routing algorithm will move some network flow from the link, which causes the link utilization to decrease. This forms an iteration. For example, suppose the initial weight of the adaptive link is 1.5. From the load curve it can be deduced that the link utilization will be 0.75. Therefore, according to the link delay curve, the link delay under such a traffic load is 4.3. MDAR will set the link weight to 4.3 in the next update. Weight 4.3 will cause a link utilization of 0.15, and the corresponding delay is 1.2. Then, MDAR will set the link weight to 1.2 in the next update, resulting in a link utilization of 0.8 and a link delay of 5. From then on, the link utilization will oscillate between 0.15 and 0.8, which means route flipping for many of the source destination pairs in the network.

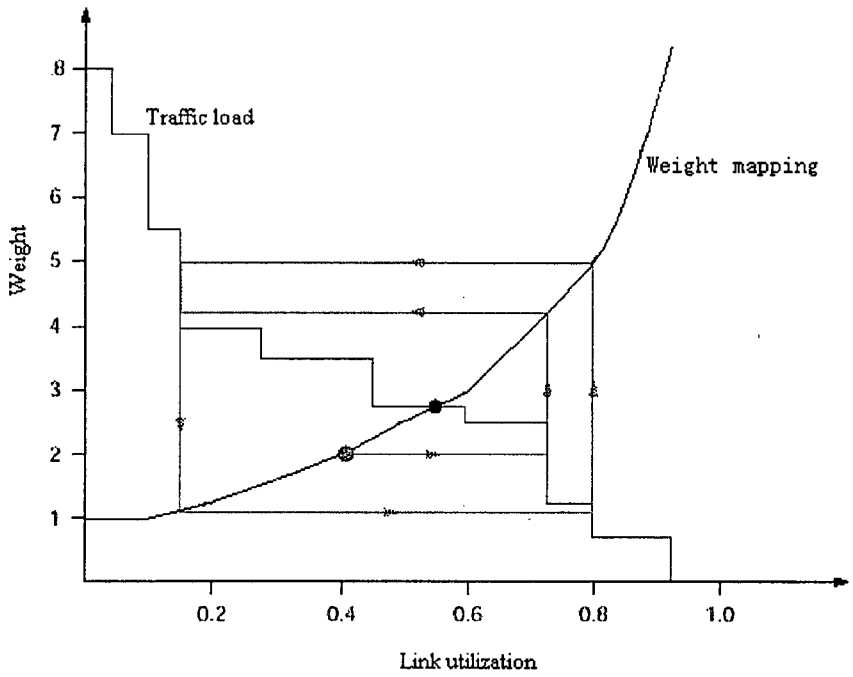


Figure 2.1 Instability of Minimal Delay Adaptive Routing

From the viewpoint of a control system, the adaptive routing can be described as the system shown in Figure 2.2. MDAR is prone to be unstable because the feedback gain is too large when the link works under relatively heavy load. Let ***W(u)*** be the weight mapping function and ***L(u)*** be the traffic load curve. In MDAR, ***W(u)*** has the same shape as a queuing delay curve, and the derivative of ***W(u)*** increases dramatically when link utilization approaches 1. This large derivative leads to a high feedback gain and makes the system prone to be unstable.

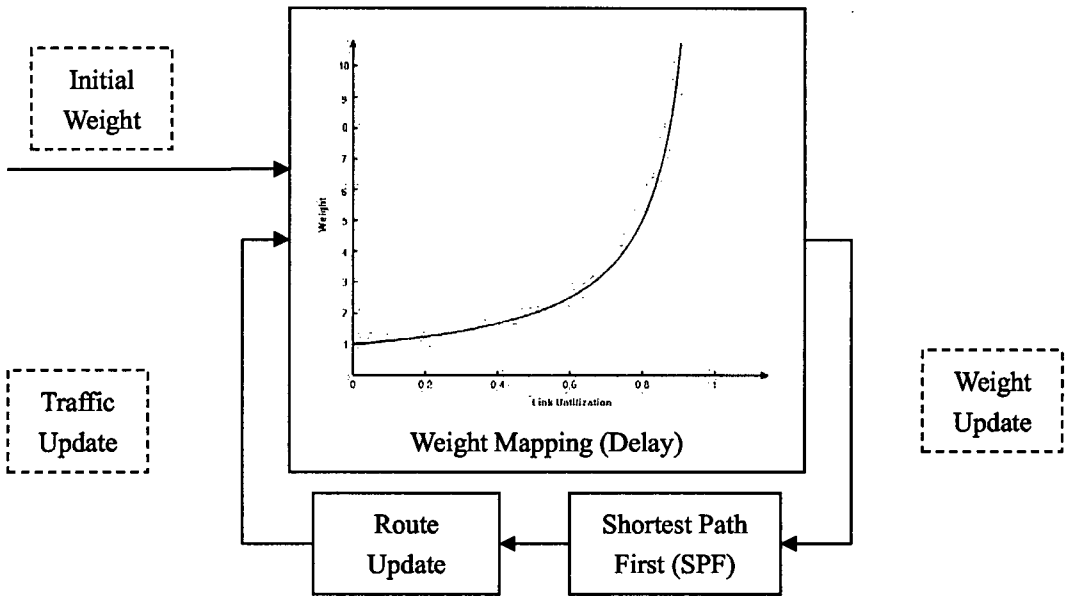
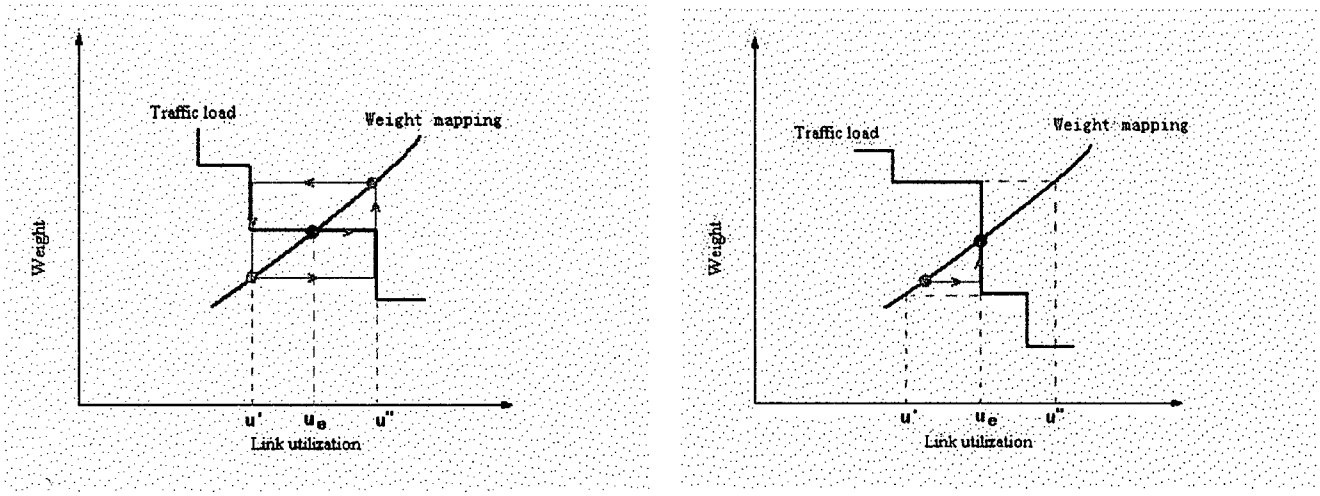


Figure 2.2 The System Block Diagram of Minimal Delay Adaptive Routing.

However, no matter what kind of shape ***W(u)*** is, ***W(u)*** and ***L(u)*** have an intersection point. This intersection point is the equilibrium point (EP) of the system. If the system remains on this point, it is stable and no further state changes will occur. There are two kinds of EP, stable EP and unstable EP. When the system EP is stable, even if the system does not work on the EP, the feedback process will drive the system state closer and closer to the EP, and finally converge at the EP. When the system EP is unstable, the system will not

converge at the EP and finally a loop will be formed, as in the example listed above. Figure 2.3 shows the examples of two types of EP.



(a) Unstable equilibrium point (b) Stable equilibrium point

Figure 2.3 Two Types of Equilibrium Points of Adaptive Routing.

2.3. Load Sensitive Adaptive Routing

Later researchers found that it is not necessary to set the link weight to a very large value. The amount of traffic on a link depends on its weight relative to the ambient weights. Generally speaking, in a network rich with alternate paths, a normalized weight of 4 is enough to shed off all the network traffic from the link[3]. Based on this finding, Load Sensitive Adaptive Routing (LSAR) [2] is raised.

Instead of non-linear weight mapping function in MDAR, LSAR adopts a sectioned linear weight mapping function (Figure 2.4). The non-adaptive section reduces the routing message overhead when the link works under light load. The linear adaptive section has a static derivative when the link works under relatively heavy load. LSAR prevents the high feedback gain problem when the link utilization is approaching 1, and the system has more chances to achieve stability. However, according to the analysis in Section 2.2, LSAR still cannot guarantee the system stability.

Since the traffic load function can be arbitrary;

a. It is impossible to find a fixed global weight mapping function which is optimized for every link.
b. No matter what kind of weight mapping function is used, it is possible that the link is unstable if the link weight is always updated according to the weight mapping function. Some supplement methods should be provide to ensure the system stability when purely weight mapping does not work.

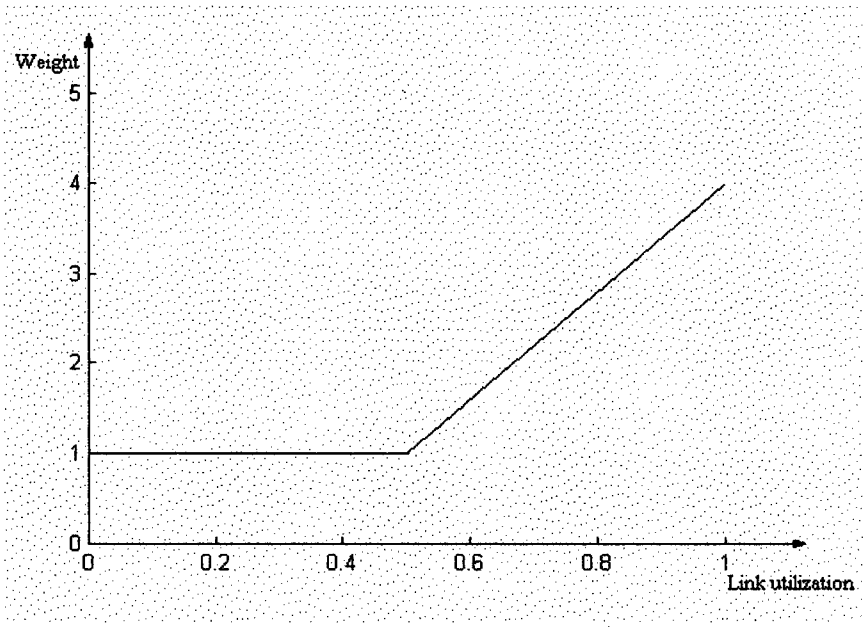


Figure 2.4 The Weight Mapping Function of LSAR.

2.4. Summary

The above analysis shows that traditional adaptive routing is unstable in nature. In a large network, ***L(u)*** on each link can be arbitrary decreasing function. Even if the ***L(u)*** of each link is known, it is very difficult to find a ***W(u)*** to make all the links have a stable EP. Furthermore, in a real network, the ***L(u)*** is often unknown.

3. DYNAMIC WEIGHT MAPPING ADAPTIVE ROUTING

There is no universal weight mapping function which can achieve system stability. However, for a given link, i.e. a given $L(u)$, there are several weight mapping functions which can make the link stable. Since the weight does not have to have a physical meaning any more, it is not necessary to have a universal weight mapping function. A customized weight mapping function for each link will solve the problem. Hence, we come to the Dynamic Weight Mapping Adaptive Routing (DWMAR).

3.1. Assumptions in DWMAR

In DWMAR, it is assumed that every stream only contributes a small portion of overall network flow, and every step of the $L(u)$ is very small. The Traffic Load Function can be looked at as a smooth curve instead of a step-shaped curve. Nowadays, as the scale of the network becomes larger and larger, so this assumption is true in most of the scenarios on the Internet.

3.2. Stability Analysis in DWMAR

Under the new assumptions, the condition of the stable equilibrium point should be re-analyzed.

For the sake of simplicity, just think about two straight lines, i.e. the traffic load curve $L(u)$ and the weight mapping curve $W(u)$ are both straight lines (*Figure3.1*):

Let h_1 and h_2 be their slopes, respectively. It is obvious that the traffic load function is a decreasing function and the weight mapping function is an increasing function, so $h_1<0$ and $h_2>0$.

If their intersection point is a stable EP, then $h_1+h_2<0$, i.e. $|h_1|>|h_2|$

If the traffic load function is given, reducing the slope of the weight mapping function helps to increase the routing stability. From the viewpoint of control theory, the same conclusion can be drawn: reducing the gain of feedback increases the system stability.

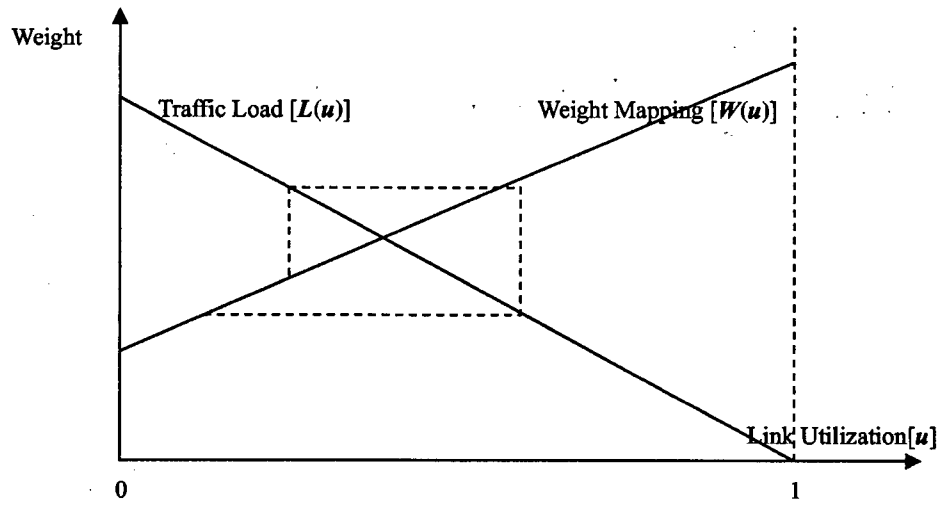


Figure 3.1 A Simplified Scenario of Adaptive Routing in Large Scale Network

3.3. Sectioned Linear Weight Mapping Function

The weight mapping function can be manually assigned, while the traffic load function can not. Furthermore, it is difficult to find the traffic load function in a large network. However, now it is known that reducing the slope of $W(u)$ near the EP helps to increase the stability of the system. A 3-section weight mapping function can therefore be designed.

1. In the first section, the link is lightly loaded, and the link delay is relatively small. In this section, the link weight keeps within a small value to attract more traffic.
2. In the second section, the link is medium loaded. The link is considered to be efficiently used while link delay is within a tolerable boundary. We want to keep the link working within this section, which means the traffic load function and weight mapping function are expected to intersect in this section. To approximate the

minimum end to end delay, it is reasonable to make the link weight proportional to link delay in this section.

3. In the third section, the link is heavily loaded. Some of traffic is to be removed from this link to alleviate the burden of this link. In this section, the weight should have a relatively large value to shed the traffic from the link.

Theoretically, the shape of each section can be arbitrary, as long as the sections meet the above definition and the weight mapping function is a continuously increasing function. However, the weight mapping function should be a trade off between minimum delay adaptive routing and system stability. To make the weight reflect the link delay to some extent, the weight mapping function should have roughly the shape of the queuing delay function, while having a smaller derivative in section 2. A linear 3-section weight mapping function is designed as shown in Figure 3.2. The weight mapping function in each section does not have to be linear, but linear function is the simplest function and easy to analyze.

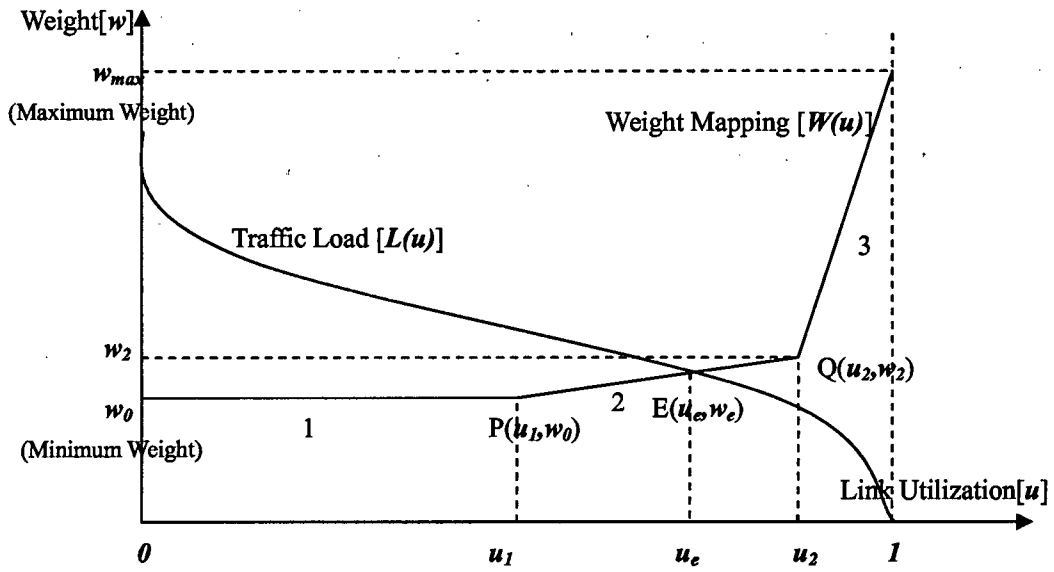


Figure 3.2 Weight Mapping Function in DWMAR

1. Section 1 is set as a non adaptive section. In this section, the link weight is frozen at the minimum weight to attract more traffic. In DWMAR network, the routers send link state messages periodically only if weight updates occur. The purpose for setting it as a non-adaptive section is to save routing message exchanges.
2. Section 2. The traffic load function and weight mapping function are expected to intersect in section 2. In this section, the slope of the weight mapping function ***W(u)*** has a small value to increase the stability of the system.
3. Section 3. In section 3, link suffers from heavy load and intolerable link delay. ***W(u)*** rockets to the maximum weight to give a large feed back so that the traffic will move from this link quickly. In this section, the system is prone to be unstable, but we will use a different policy to achieve system stability, which will be described in the algorithm.

3.4. Dynamic Weight Mapping Functions

To guarantee that the two curves intersect in section 2, a dynamic weight mapping function must be used. During the adaptive routing process, the positions of ***P*** and ***Q*** can be adjusted to achieve the purpose that

a. ***W(u)*** and ***L(u)*** intersect in section 2, and
b. The EP is stable.

It is not easy to precisely decide the position of ***P*** and ***Q***, because there is no precise definition of a 'light load link', 'medium load link' or 'heavy load link'. It depends on the traffic pattern and the delay requirement of traffic itself. Given the arrival rate, compared to an M/M/1 queue, if the traffic has a more even distribution, the expected delay will be smaller, otherwise, the delay will be longer. At the same time, given an end to end delay, it may be acceptable to a bunch file transfer application while unacceptable to a real-time application. It is difficult to establish a mathematical model to analyze the traffic in a

complex network. The most basic and commonly used model, M/M/1 queue, is referred to to decide the boundary of ***P*** and ***Q***. The following discussion is qualitative more than quantitative.

Assuming that the minimum weight and the link propagation delay are all normalized and both are 1, from the queuing theory, the expected link delay is $1 / (1 - u)$, where u is the link utilization. When the delay is less than 2, the link is considered to be lightly loaded. When the delay is greater than 5, the link is considered to be heavily loaded. From these assumptions, the parameter settings are:

1. $u_1 \in [0, 0.5]$ and $u_2 \in [0.7, 0.85]$
2. Since in networks that are rich with alternate paths, the normalized weight of 4 is big enough to shed all the traffic off a link [3], the maximum weight w_{max} should not exceed 4. In the simulation, w_{max} is fixed at 4, which is discussed and supported by simulations at Section 4.1.1.
3. The slope of section 2 should be less than section 3.

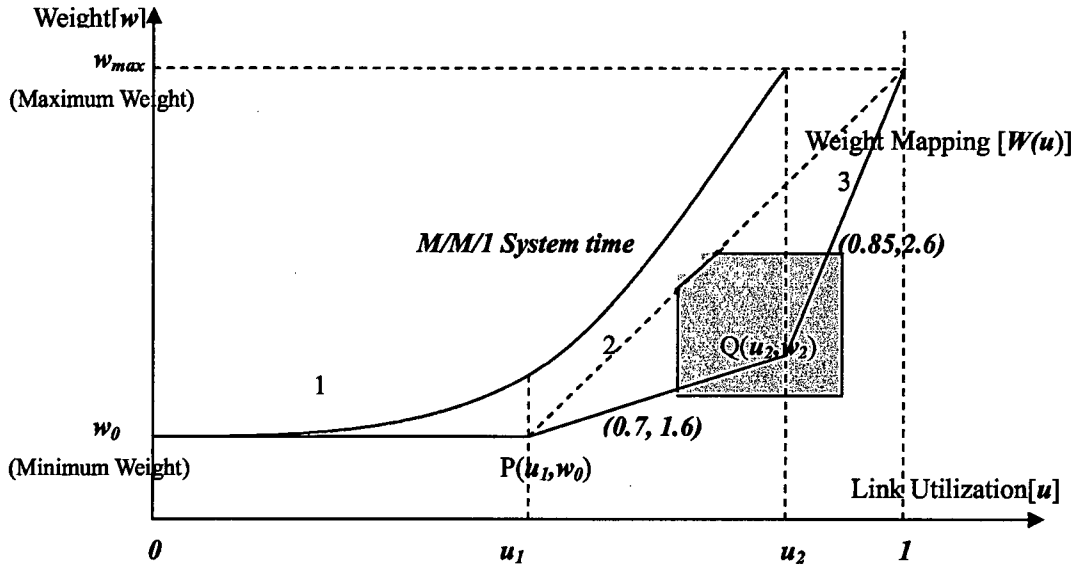


Figure 3.3 Decide the Adaptive Range of Weight Mapping Function

Summing up all these conditions, the position of P is on line $w = 1$ and adjustable from $(0, 1)$ to $(0.5, 1)$. The position of Q is adjustable in the shaded area in Figure 3.3.

Compared to LASR, the 3-sectioned weight mapping function of DWMAR tends to make links work under "medium load". Fewer links are working under heavy load, which means less packet drop ratio is expected. Low packet drop ratio also leads to larger network capacity and less link utilization variation. At the same time, the dynamic weight mapping function increases the system stability. When accompanied by the various adaptive schemes introduced in 3.5, DMWAR provides guaranteed stability of system.

3.5. DWMAR Algorithm

Here the basic idea is given for how to adjust the weight mapping function to achieve the system stability. There are 2 major steps in the algorithm, one is to find the position of equilibrium point, the other is to detect an unstable equilibrium point and adjust the weight mapping function to make it stable.

The status machine of DWMAR is shown in Figure 3.4.

1. If $L(u)$ and $W(u)$ intersect in section 1, there is no need to do any adjusting, either weight or weight mapping function
2. If $L(u)$ and $W(u)$ intersect in section 2, and

    1) the EP is stable.

       This is an ideal result, no further adjusting is needed.

    2) the EP is not stable.

       Reduce u_1 and/or increase u_2 to increase the possibility of stability. If it does not work, do not follow the weight mapping function. Search the section to find the EP (e.g binary search).

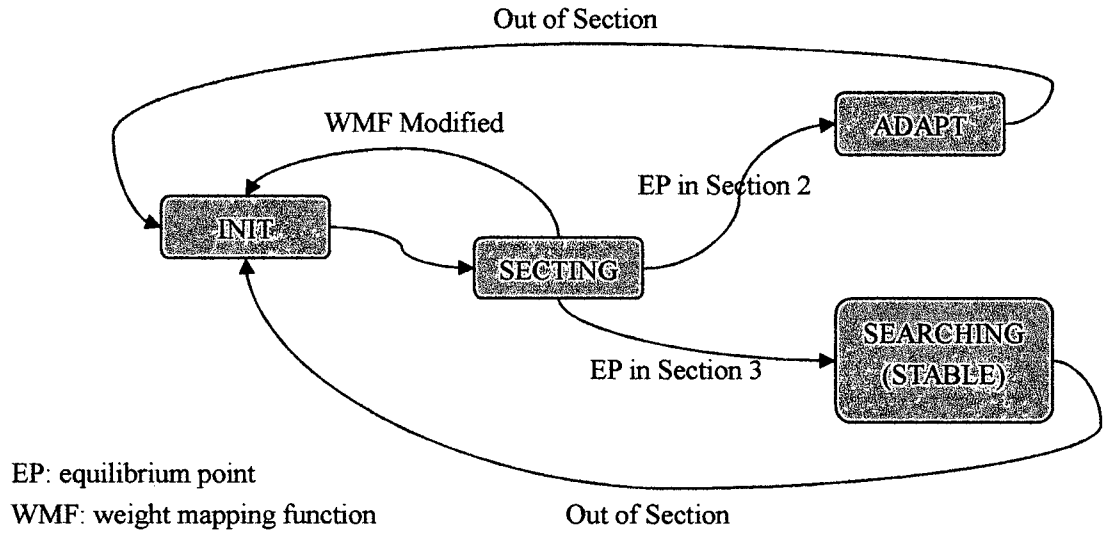


Figure 3.4 Status Machine of DWMAR

3. If ***L(u)*** and ***W(u)*** intersect in section ***3***

   Firstly, increase u_2 and w_2 so that ***W(u)*** and ***L(u)*** can intersect in section ***2***. If it does not work, do not follow the weight mapping function, search in section ***3*** to find the equilibrium point. If it does, follow step 2.

In DWMAR algorithm, it is unnecessary to construct ***L(u)***. Since ***W(u)*** and ***L(u)*** are increasing and decreasing function, respectively, they must have one and only one intersecting point, the equilibrium point (EP). The key point of DWMAR algorithm is to find the EP, adjust EP to be in section 2, and make it stable.

Now let us discuss how to distinguish situation ***2*** and ***3*** in practical terms.

a. Assume that current link weight is in section ***1*** or section ***2***, and at the next update interval, the link utilization goes into section ***3***.

b. Set link weight to w_2, and wait to the next update interval.

c. If the link utilization drops within section *2*, then it is situation 2, otherwise, it is situation 3.

In situation 2, the routing oscillation should be detected, since the oscillation means an unstable equilibrium point. It can be done by analyzing the recent history data (2 or 3 update intervals before). If the difference between two neighboring weights is decreasing, the EP is considered to be stable. Otherwise, if the difference is either increasing or uncertain, then the EP is unstable. The minimum weight update should also be set to save network traffic, as well as maximum update.

The algorithm above is based on constant network flow. Under the constant network flow, this algorithm is guaranteed to converge. If the network flow changes, the algorithm should be restarted. The following situation means that the network flow changes:

1. The EP is expected in section *3*, the current link weight is in section *3*, but at the next update interval, the link utilization is not in section *3*.
2. The EP is expected in section *2*, the current link weight is in section *2*, but at the next update interval, the link utilization is in section *3*.

When the network traffic is dynamic, DWMAR will update link weights according to algorithm. This process will not stop unless the network traffic is constant. But this does not mean that DWMAR does not converge, they are two different concepts.

4. SIMULATION

The simulation software is ns2.27 [23]. DWMAR is implemented by modifying the LS protocol included in ns2. The topology in the simulation is generated by the GT-ITM Internet topology generator [24]. The duplex link class in ns2 is also modified to make it a real duplex link, i.e. traffic from both directions share the link bandwidth.

4.1. Simplification of Parameter Settings

There are several dynamic parameters in DWMAR, although some of them only have a small impact on routing performance. Fixing these parameters will not significantly affect the simulation results. These parameters include

a. the maximum weight w_{max}, and
b. the horizontal coordinate of P u_1

Fixing these two parameters helps to simplify the adaptive routing logic and saves CPU resources of the routers. Furthermore, maximizing the non-adaptive section will save the route information exchange between routers, if this maximizing will not significantly reduce the benefit of adaptive routing. For the sake of simplicity, assuming the traffic load function $L(u)$ is a straight line, a qualitative analysis is given below on the impacts of fixing these 2 parameters.

4.1.1. Fixing w_{max}

According to the analysis in 3.2, decreasing w_{max} will increase the stability of the system in section 3, while increasing w_{max} will reduce the stability. However, increasing w_{max} increases the sensitivity of the feedback system. Since one of the aims of DWMAR is to prevent links working in section 3 and DWMAR uses a different adaptive policy to achieve system stability in section 3, the key point in this section is not the stability

problem, but the reflection speed of the system. It is reasonable to set the w_{max} to the maximum necessary value. According to [3], w_{max} can be fixed at ***4***. Since the actual $L(u)$ is not easy to know, this hypothesis is to be proved by the following simulation.

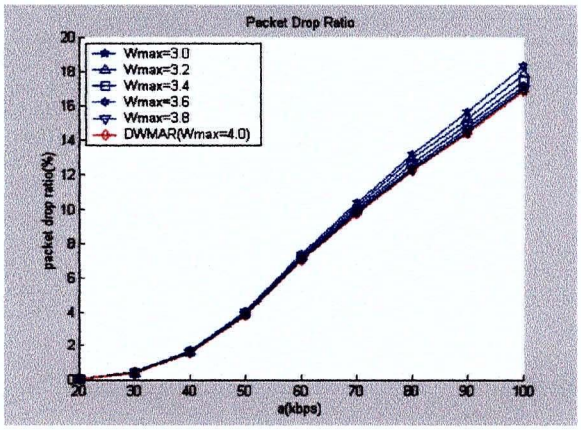
Static traffic is used in this simulation. This traffic is generated using a method similar to the one used in [9]. Two random numbers, O_x, $D_x \in [0, 1]$, are assigned to each node x in the network, and another random number $C_{(x,y)} \in [0, 1]$ is assigned to each pair of nodes (x, y). The flow between each origin destination pair is implemented by a CBR UDP flow. The traffic rate from origin node x to destination node y is calculated as

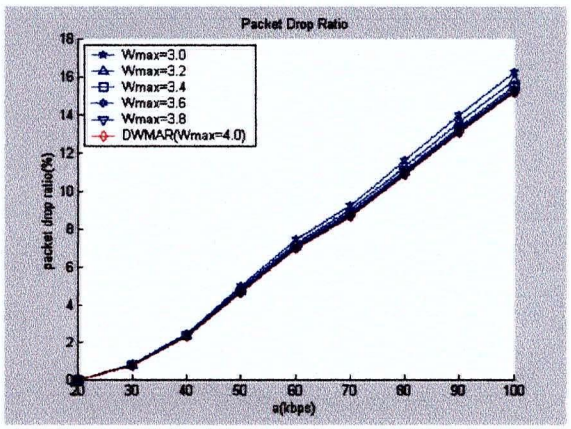
$$R_{(x,y)} = \alpha * O_x * D_x * C_{(x,y)}$$

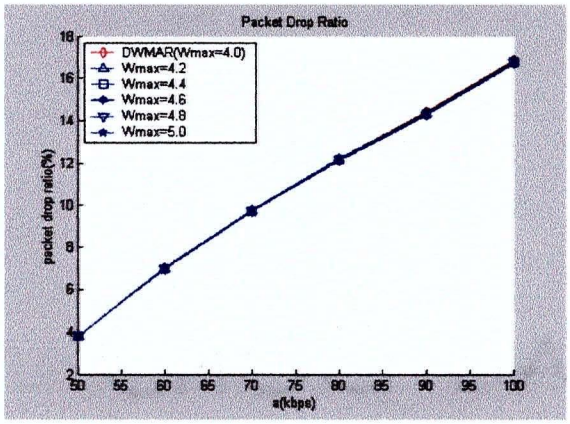
Where $R_{(x,y)}$ is the traffic rate, and α is the traffic load factor used to adjust the traffic flows. Different values of O_x and D_x can make node x a more or less active sender or receiver, and this gives us the ability to simulate the hot spots on the network.

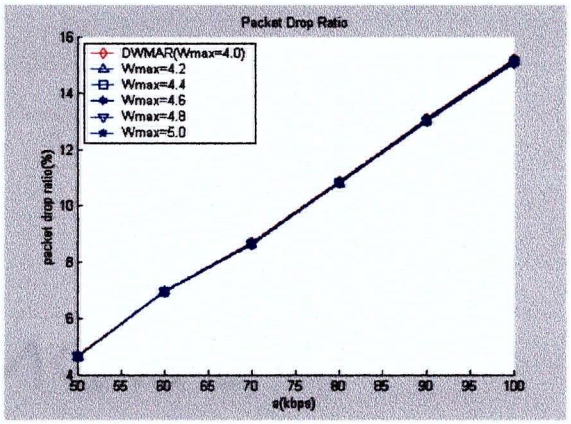
By controlling the seed of the random number generator, the traffic flows on the compared algorithms are exactly the same. The traffic flow patterns for a set of simulations performed on the same network topology are also the same, except for the traffic load factor α.

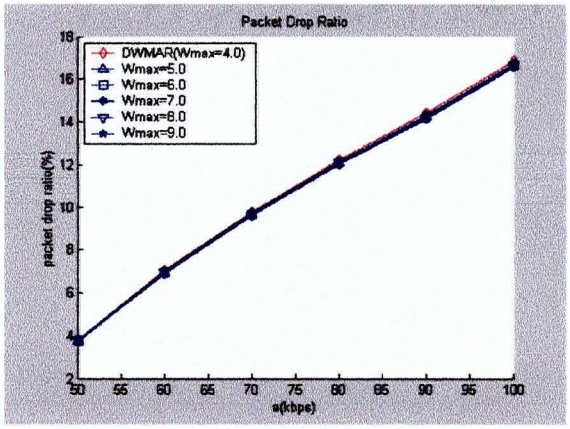
In this simulation, w_{max} is varied from 3.0 to 9.0 to test the impact on DWMAR. During a given simulation round, w_{max} is still static. The packet drop ratio is used to evaluate the performance. Simulations are run on 2 different network topologies, which are generated by GT-ITM[24] in transit-stub mode.

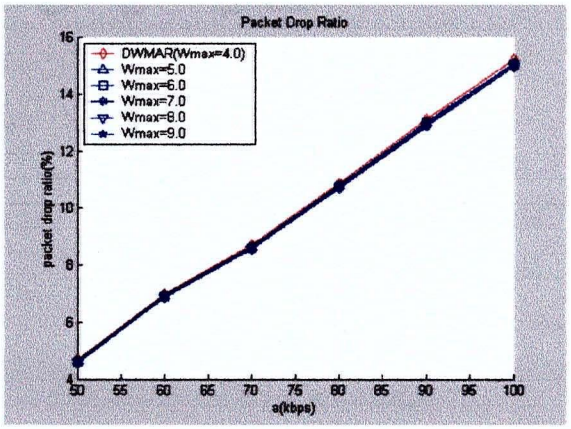












63 nodes, 155 links

63 nodes, 157 links

Figure 4.1 Impacts of Varying w_{max}

According to the simulation results, reducing w_{max} from 4 to 3 does not increase the performance of DWMAR. On the contrary, it reduces the performance. One possible reason is that links will work under relatively higher load, which increases the possibility of packet dropping.

At the same time, increasing w_{max} from 4 does not increase the performance significantly either. According to the simulation results, varying w_{max} from 4 to 9 has almost no impact on the performance. This is so because 4 is large enough to shed most traffic from a link[3]. Setting w_{max} to a larger value than 4 will not help to remove traffic from a heavily loaded link. When w_{max} is set to 4 and the network reaches stability, if a link is still working on section 3, this means that this link is a bottle neck of the network. A further increment of the link weight will not help to relieve the load of the link. On the other hand, when w_{max} increases, the routing takes more update rounds to reach stability. For example, when w_{max} is between 3 and 4, the network takes fewer than 15 update rounds to achieve stability for both of the 2 topologies. However, it takes more than 70 rounds when w_{max} is set to 9.0. In a real network, the exchanging routing information will cost a lot of network bandwidth. During the adaptation routing process, more packet dropping will happen than in the stable routing stage.

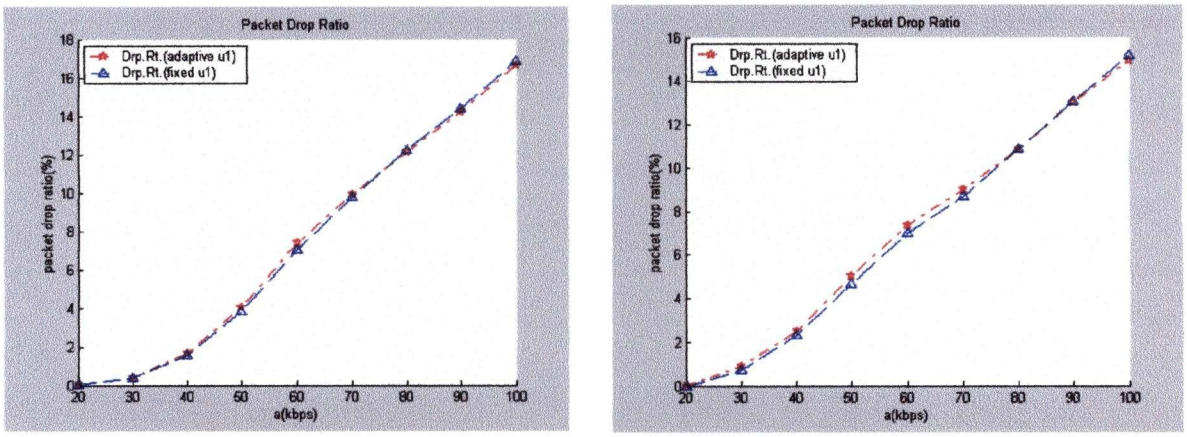
Summing up all of the above analysis, fixing w_{max} at ***4*** is better than making it adaptive. This is a trade off between stability and effectiveness.

4.1.2. Fixing u_1

Moving u_1 from (0.5, 1) to (0, 1) only slightly decreases the slope of section 2, which means only a small contribution to system stability. At the same time, fixing the u_1 can save lots of routing information exchanged among routers. Furthermore, the less route update occurs, the more stable the network is, and the faster the algorithm converges. It is reasonable to fix u_1 at ***0.5*** and this will not affect the performance of DWMAR

significantly. This parameter setting is also supported by simulation.

A small modification is done to the algorithm in 3.5. In this simulation, DMWAR takes an extra effort to make a link stable in section 2. After adjusting the position of Q, DMWAR will try to reduce u_1 until $u_1=0$ to increase the stability of the system. This simulation can test the impact of adjusting u_1. The simulation results are compared with the fixed u_1 algorithm and also evaluated by packet drop ratio.



63 nodes, 155 links 63 nodes, 157 links

Figure 4.2 Impacts of Varying u_1

According to the simulation results, making u_1 adjustable does not decrease the packet drop ratio at all. The results are almost the same as the fixed u_1 algorithm. The possible reasons are

a. the improvement is so slight that it cannot be observed by simulation
b. the improvement is counteracted by the overhead of routing information exchanges.

4.1.3. Parameter Settings in Simulation

After the simplification, the parameters used in the DWMAR simulation are listed below:

Table 4.1 Parameters in DWMAR Simulation

Parameters	Meanings	Values
MAX_WEIGHT	Maximum weight	***4.0***
U1	Bound of non-adaptive section	***0.5***
u2max	The boundary of Q (see Figure. 3.3)	***0.85***
w2max		***2.6***
u2min		***0.7***
w2min		***1.6***
wt_aver_len	Number of historical weights used in averaging	***4***
Min_w_updt	The minimum weight changes between two consecutive updates	***0.1***
updt_itvl	The interval of sending routing update message	***2s***

The results of DWMAR are compared with the results of non-adaptive routing (SPF) and fixed weight mapping adaptive routing (LSAR). LSAR is implemented simply by setting

$u2max = u2min = 1$ *and*

$w2max = w2min = MAX\_WEIGHT = 3$

The performance of DWMAR is compared with the other two routing protocols on both static traffic loads and dynamic traffic loads.

4.2. Static Traffic

The static network traffic is generated using the method described in Section 4.1.1

The performance of DWMAR is compared with SPF and LSAR based on the 6 network topologies shown in Table 4.2, which are generated by GT-ITM[24] in transit-stub mode except topology 1, which is manually set according to the transit-stub mode. Topology 1 is shown in Figure 4.3. The generating files of other topologies are listed in Table 4.3. All the links has the bandwidth of 1Mbps. The buffer size of each router port is 20.

Table 4.2 Network Topologies Used in Static Traffic Simulation

Topology Number	Number of Nodes	Number of Links
1	16	27
2	42	91
3	63	155
4	63	157
5	90	194
6	90	211

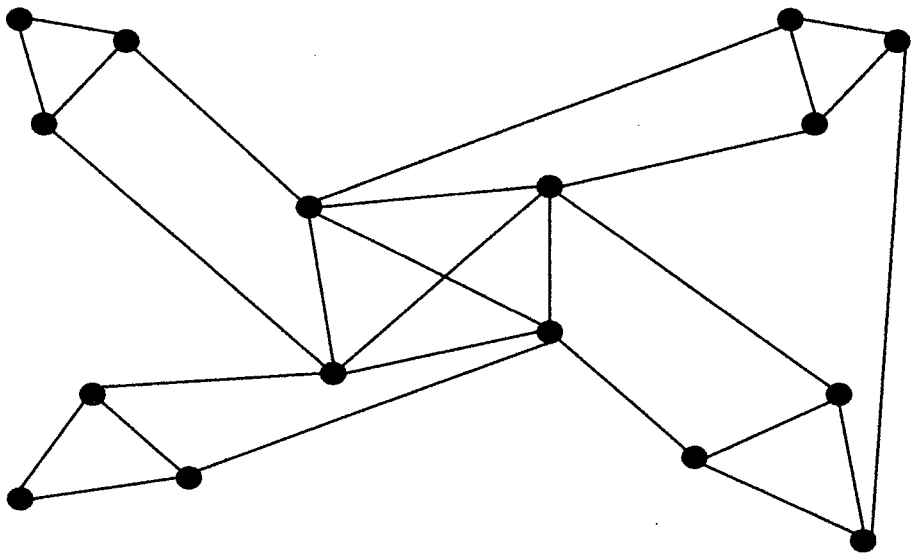


Figure 4.3 Topology 1 (16 Nodes, 27 Links)

Table 4.3 Generating Files of Topologies Used in Static Traffic Simulation

Topology Number	2	3 and 4	5 and 6
Generating File	ts 1 67 3 30 20 1 20 3 1.0 6 20 3 1.0 2 10 3 0.9	ts 2 47 2 20 20 1 20 3 1.0 7 20 3 0.95 4 10 3 0.9	ts 2 47 3 30 30 1 20 3 1.0 9 20 3 0.95 3 10 3 0.9

4.2.1. The Performance of DWMAR

The simulation results are listed from Figure 4.4 to Figure 4.9. All the data are collected after the algorithm has converged.

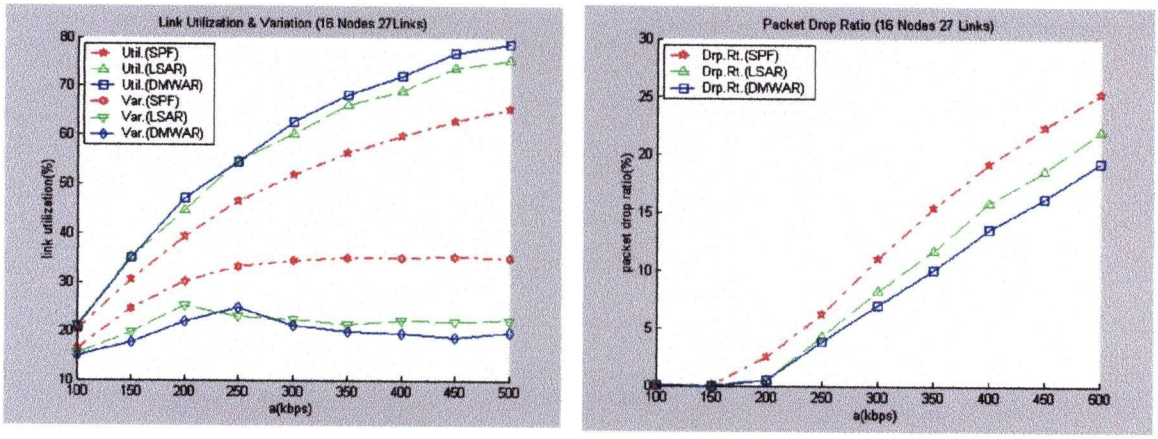


Figure 4.4 Simulation Results of Static Traffic on Topology 1 (16 Nodes, 27 Links)

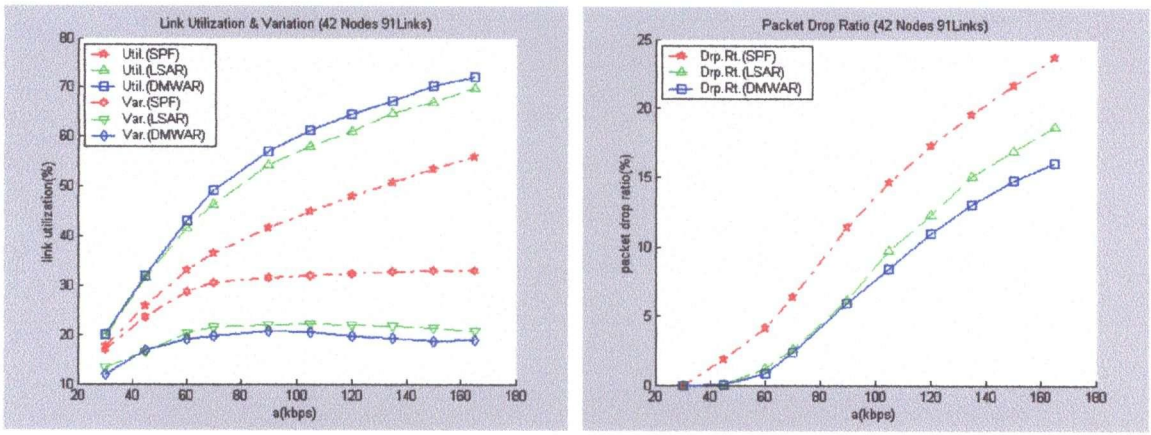


Figure 4.5 Simulation Results of Static Traffic on Topology 2 (42 Nodes, 91 Links)

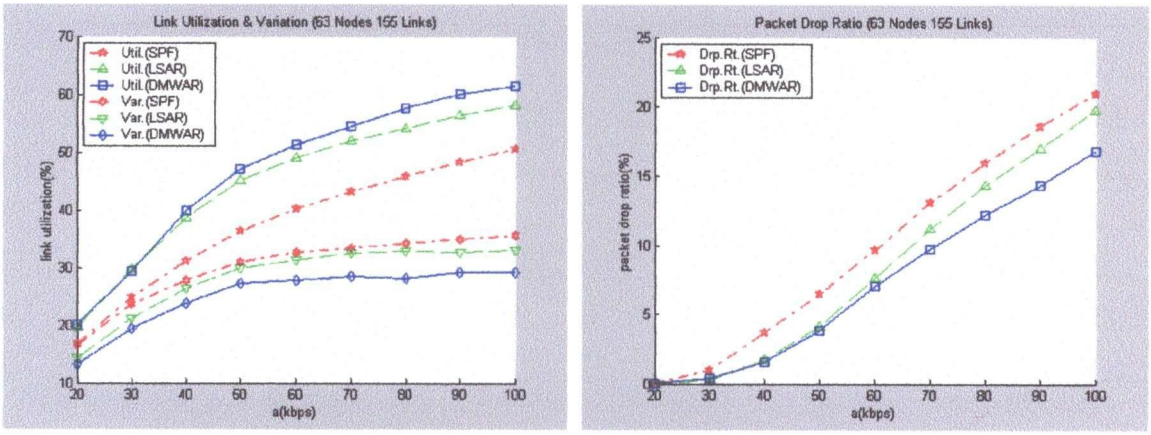


Figure 4.6 Simulation Results of Static Traffic on Topology 3 (63 Nodes, 155 Links)

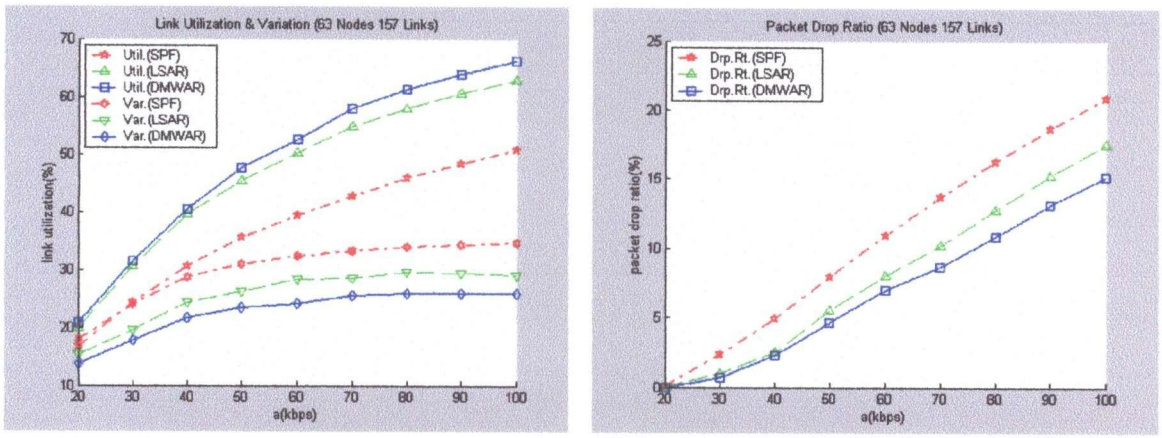


Figure 4.7 Simulation Results of Static Traffic on Topology 4 (63 Nodes, 157 Links)

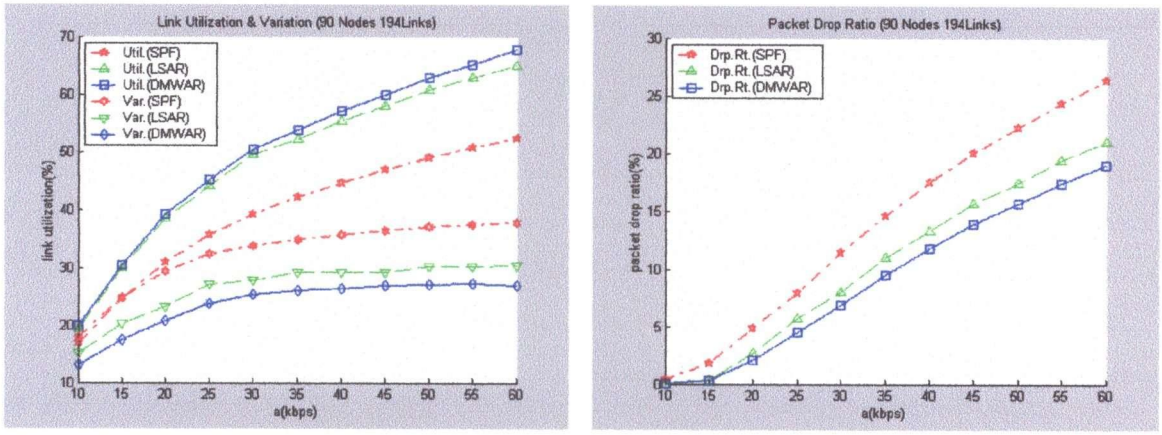


Figure 4.8 Simulation Results of Static Traffic on Topology 5 (90 Nodes, 194 Links)

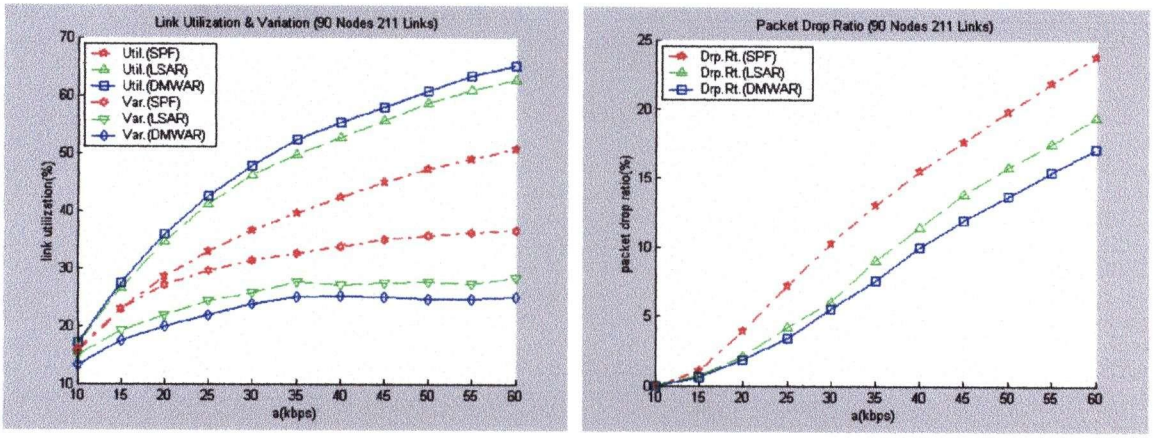


Figure 4.9 Simulation Results of Static Traffic on Topology 6 (90 Nodes, 211 Links)

According to simulation results, both DWMAR and LSAR show much better performance than SPF: lower packet drop ratio, higher link utilization and lower link variation. DWMAR has a small advantage over LSAR, which matches our expectation before simulation.

In the static traffic simulation, the overall data sent by sources are fixed in a scenario, no matter what kind of routing algorithm is chosen. Lower packet drop ratio means more network capacity. On the other hand, the increase of link utilization is not proportional to the decrease of packet drop ratio. This is so because the increase of link utilization not only comes from the increase of network capacity, but also comes from the roundabout route selected by adaptive routing. In the static traffic simulation, the network capacity increase contributes to about 40% of the link utilization increase. The adaptive routing also makes the traffic flow more evenly distributed on network links, resulting in lower link variation.

DWMAR has a small advantage over LSAR because of the DWMAR trend to make the link loads concentrate on the 'medium high' section. For DWMAR, although the average link utilization is higher, fewer links work under heavy load than in LSAR. Thus, better performance is observed on packet drop ratio and link variation.

4.2.2. The Stability of DWMAR

For each traffic pattern and each topology, the number of update rounds elapsed before system reached the stable state was recorded. The results are shown from Figure 4.10 to Figure 4.15. In this simulation, the stability of MDAR is also tested as a reference. MDAR is implemented by set the weight mapping function ***W(u) = 1/(1-u)*** which is fixed during the routing progress. MDAR also has the same parameters of average damping and minimum weight update as the LSAR and DWMAR's. Simulation was run for 500 update rounds for each traffic pattern and each topology.

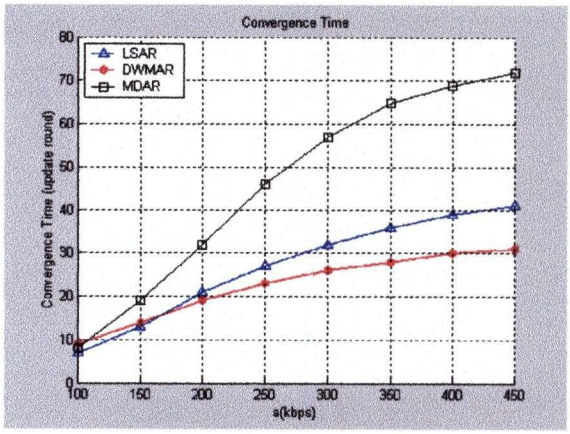


Figure 4.10 Convergence Time for Topology 1

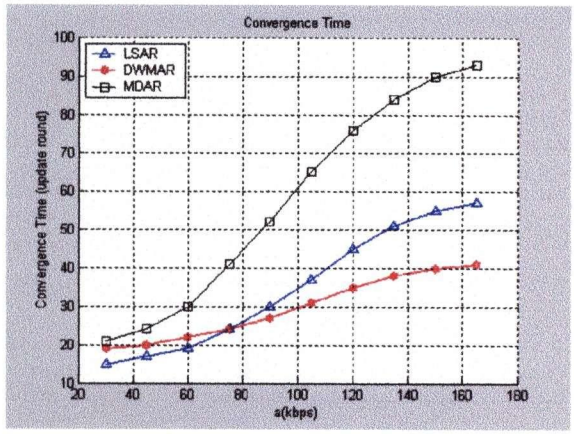


Figure 4.11 Convergence Time for Topology 2

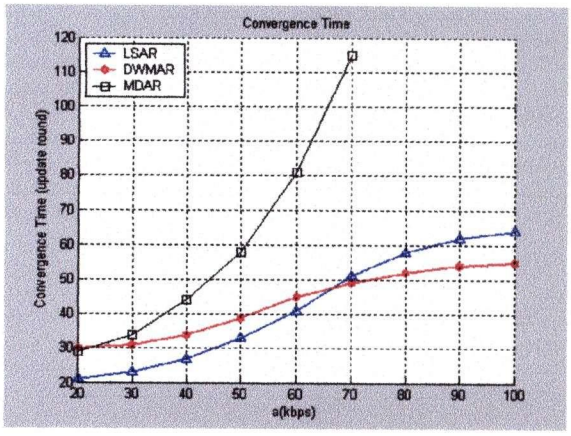


Figure 4.12 Convergence Time for Topology 3

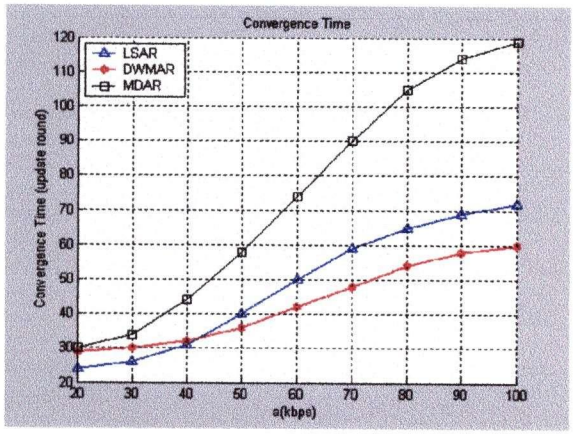


Figure 4.13 Convergence Time for Topology 4

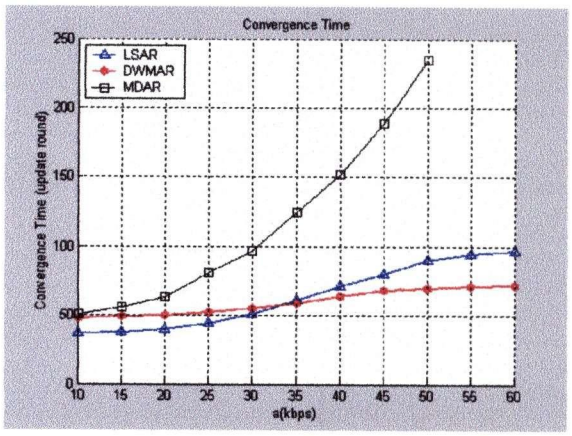


Figure 4.14 Convergence Time for Topology 5

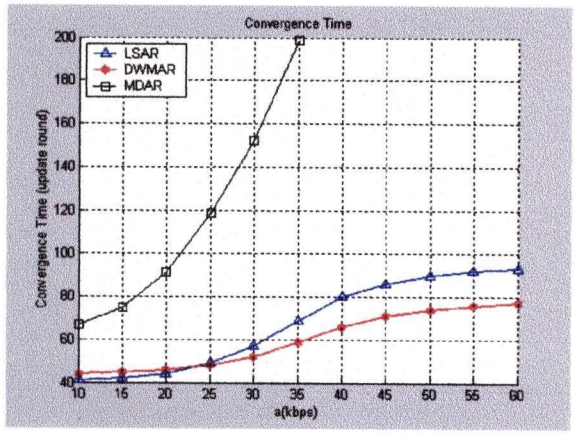


Figure 4.15 Convergence Time for Topology 6

Some of the simulation results of MDAR are not shown in the figures, which means the algorithm did not converge before 500 update rounds.

In the static traffic simulations, packet drop ratio is a criterion to evaluate the network performance, and convergence time is the one to evaluate the network stability. According to the simulation results above, the relationship between network stability and network performance can be obtained. For different network topologies and different

traffic patterns, comparing the absolute values of packet drop ratio and convergence time does not reflect the relationship between stability and performance. Hence, the relative values are used to show this relationship. The data of LSAR are used as the reference. The data of DWMAR is normalized by their correspondent data of LSAR under the same topologies and same traffic patterns. To make the data more representative, the data obtained form light traffic and heavy traffic are removed. When the traffic is too light, most of the links are working on the non-adaptive section. And when the traffic is too heavy, the traffic is beyond the adaptive capability of the AR algorithm and approaching the theoretical maximum capacity of the network. Neither of the situations will correctly reflect the relationship between stability and performance within the adaptive capability of AR algorithm. The refined result, which is a collection of discrete point, is shown in Figure 4.16. To further clarify the relationship between the stability and performance, least-squares fitting is used to fit all these data in a line. The line has a positive slope and shows that the packet drop ratio tends to increase when the convergence time increases, i.e. the network performance is proportional to network stability.

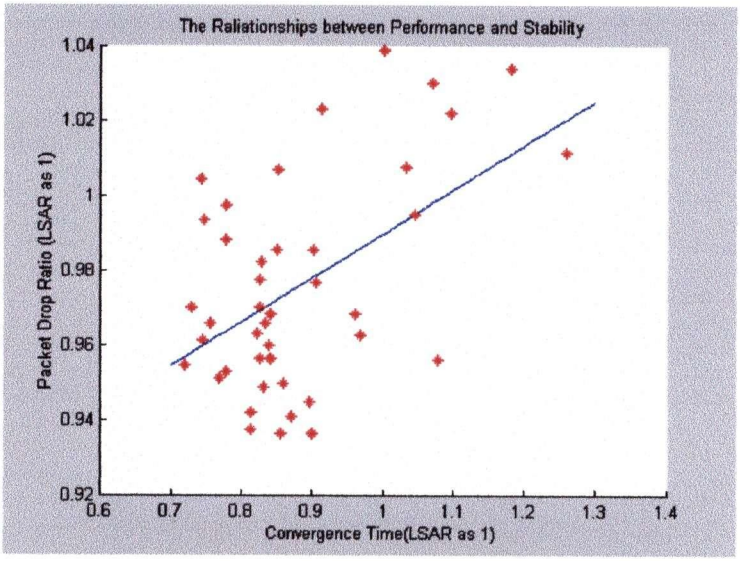


Figure 4.16 The Relationships between Performance and Stability

According to the results, both LSAR and DWMAR are more stable than MDAR. This is because that the feedback gain of MDAR is too large when the link utilization is high. If the feedback gain is beyond the impact of averaging damping, a routing oscillation forms and the algorithm diverges. For LSAR and DWMAR, unnecessary high feedback gain is prevented by setting the maximum weight to a relatively small value. They are able to reach a stable status much easier than MDAR. Furthermore, DWMAR detects routing oscillation and uses a different policy to find the EP, which guarantees the stability of the routing.

Compared with LSAR, DWMAR does not shorten the convergence time when the traffic load is light. On the contrary, it increases the convergence time slightly. However, when the traffic load becomes heavy, DWMAR increases the system stability significantly. Overall, the convergence time of DWMAR has a more even pattern, while the convergence time of LSAR tends to increase dramatically with the increase of network traffic load.

When the network traffic load is light, in DWMAR algorithm, most of the links initially work on section 2 or 1. Compared with the adaptive section in LSAR, weight mapping function in section 2 has a less slope. A less slope means a less feedback gain. According to the control system theory, a less feedback gain makes system more stable, but reacts more slowly. That is why DWMAR takes more time to reach stable state.

When the network traffic load is heavy, most of the links work initially on section 3. The slope of section 3 is greater than the slope of adaptive section in LSAR. A greater feedback gain makes DWMAR converge faster than LSAR. Furthermore, for links whose weight cannot be adjusted to reach a stable EP, DWMAR uses a different policy to avoid weight oscillation, saving the convergence time. So when the network traffic is heavy, DWMAR has a much shorter convergence time than LSAR.

4.3. Dynamic Traffic

In real life, the network traffic is dynamic rather than static. Furthermore, most of the traffic on the Internet comprises TCP streams, which have their own congestion control mechanism. The performance cannot be judged purely by packet drop ratio. The network performance should be evaluated by checking the overall valid data received by all destination nodes.

The method introduced in [10] is used to generate dynamic traffic. The traffic pattern for each protocol is listed in Table 4.4. A similar method as used in the static traffic simulation which is described in Section 4.1.1, is used to generate the average rate of each application (including the rates of CBR UDP streams).

Table 4.4 Traffic Patterns for Simulated Applications used in Dynamic Traffic Simulation

Application	Distribution		
	Inter-arrival	Duration	Packet Size
HTTP	Exponential	Log-normal	Self-Similar
FTP	Exponential	Log-normal	Pareto
SMTP	Exponential	Log-normal	Log-normal
TELNET	Exponential	Log-normal	Pareto

The percentage of each kind of session is listed in Table 4.5.

Table 4.5 Session Percentage in Dynamic Traffic Simulation

Protocol	Application	Percentage
TCP(Reno)	HTTP	50%
	FTP	15%
	SMTP	15%
	TELNET	10%
UDP	(CBR)	10%

Thirty minutes of dynamic traffic load are randomly generated using this method. Overall TCP data received by all TCP sinks are used as the major parameter to evaluate the performance of each routing protocol, because this parameter represents the effective network capacity. As mentioned before, packet drop ratio cannot be the major parameter because of TCP's congestion control mechanism. If links congest and some TCP packets are dropped, less data will be sent from the TCP source in the next short period. Three different routing protocols are run on 30 different topologies, which are also created by GT-ITM. The simulation results are listed in Table 4.6 and 4.7.

Table 4.6 Simulation Results of Dynamic Traffic for SPF and DWMAR

	Nodes	Links	SPF				DMWAR						
			Util. Rt. %	Var. %	TCP Data (Bytes)	Drp Rt. %P	Util. Rt. %	Impr.	Var. %	TCP Data (Bytes)	Impr.	Drp Rt. (Packet)	Impr.
1	42	98	23	26	1.8E+09	2.7E-04	31	0.36	21	2.1E+09	0.19	2.6E-04	0.05
2	42	95	25	28	1.7E+09	3.7E-04	30	0.20	19	2.0E+09	0.21	3.0E-04	0.17
3	42	87	29	29	1.4E+09	5.9E-04	33	0.13	21	1.7E+09	0.18	5.4E-04	0.09
4	42	92	25	27	1.4E+09	2.1E-04	30	0.19	22	1.7E+09	0.16	2.1E-04	0.04
5	42	91	26	27	1.7E+09	3.7E-04	35	0.35	21	1.9E+09	0.15	1.5E-04	0.58
6	42	99	24	27	1.9E+09	1.0E-04	30	0.24	20	2.2E+09	0.19	3.8E-05	0.63
7	42	87	28	28	1.8E+09	4.8E-05	34	0.20	23	2.2E+09	0.20	3.8E-05	0.20
8	42	100	24	26	2.0E+09	3.6E-05	30	0.24	20	2.2E+09	0.12	1.5E-05	0.57
9	42	99	25	28	2.0E+09	1.2E-05	28	0.13	20	2.3E+09	0.13	1.3E-05	-0.11
10	42	98	24	26	1.7E+09	1.1E-04	29	0.21	21	2.1E+09	0.21	8.1E-05	0.27
11	63	156	24	27	2.1E+09	4.2E-04	29	0.23	20	2.6E+09	0.24	3.5E-04	0.15
12	63	155	25	27	2.2E+09	5.1E-04	30	0.21	21	2.6E+09	0.18	3.0E-04	0.40
13	63	157	26	25	2.2E+09	3.9E-05	29	0.12	20	2.7E+09	0.20	4.1E-05	-0.05
14	63	154	25	28	2.4E+09	3.1E-04	31	0.27	23	2.8E+09	0.17	2.1E-04	0.31
15	63	152	23	26	2.0E+09	2.9E-04	29	0.30	22	2.4E+09	0.21	2.0E-04	0.32
16	63	161	26	27	2.4E+09	6.6E-05	29	0.12	20	2.7E+09	0.13	6.6E-05	0.01
17	63	159	22	25	2.1E+09	2.1E-04	28	0.24	21	2.4E+09	0.17	1.6E-04	0.26
18	63	168	24	27	2.1E+09	1.5E-05	25	0.04	21	2.4E+09	0.13	1.5E-05	-0.03
19	63	159	23	27	2.3E+09	4.2E-04	27	0.16	22	2.6E+09	0.17	3.5E-04	0.16
20	63	163	23	27	2.3E+09	3.6E-04	27	0.17	22	2.7E+09	0.16	3.0E-04	0.17
21	80	164	25	27	2.1E+09	2.4E-04	28	0.12	23	2.4E+09	0.19	1.9E-04	0.21
22	80	162	23	26	2.0E+09	5.0E-04	27	0.19	23	2.4E+09	0.24	2.4E-04	0.52
23	80	162	25	27	2.0E+09	2.8E-04	29	0.14	22	2.5E+09	0.25	9.4E-05	0.66
24	80	176	21	25	2.0E+09	7.8E-04	25	0.20	23	2.5E+09	0.27	3.2E-04	0.59
25	80	147	28	27	2.0E+09	3.2E-04	34	0.25	23	2.7E+09	0.32	1.1E-04	0.65
26	80	152	24	28	2.0E+09	3.5E-04	29	0.21	24	2.3E+09	0.16	2.2E-04	0.37
27	80	158	23	26	2.2E+09	6.0E-04	29	0.25	22	2.7E+09	0.24	3.0E-04	0.50
28	80	160	26	27	2.3E+09	3.3E-04	31	0.19	24	2.6E+09	0.12	2.7E-04	0.17
29	80	162	23	26	2.0E+09	4.1E-04	28	0.23	24	2.5E+09	0.21	3.1E-04	0.24
30	80	160	24	27	2.0E+09	1.7E-04	27	0.13	24	2.5E+09	0.25	1.3E-04	0.28
AV.	-	-	-	27	-	-	-	0.20	22	-	0.19	-	0.28

Table 4.7 Simulation Results of Dynamic Traffic for SPF and LSAR

	Nodes	Links	SPF				LSAR						
			Util. Rt. %	Var. %	TCP Data (Bytes)	Drp Rt. %P	Util. Rt. %	Impr.	Var. %	TCP Data (Bytes)	Impr.	Drp Rt. (Packet)	Impr.
1	42	98	23	26	1.8E+09	2.7E-04	29	0.25	20	2.0E+09	0.12	2.3E-04	0.17
2	42	95	25	28	1.7E+09	3.7E-04	28	0.13	19	1.9E+09	0.13	3.2E-04	0.14
3	42	87	29	29	1.4E+09	5.9E-04	30	0.05	22	1.6E+09	0.10	5.7E-04	0.03
4	42	92	25	27	1.4E+09	2.1E-04	29	0.14	22	1.5E+09	0.07	2.1E-04	0.00
5	42	91	26	27	1.7E+09	3.7E-04	33	0.26	22	1.8E+09	0.08	1.5E-04	0.60
6	42	99	24	27	1.9E+09	1.0E-04	28	0.18	19	2.1E+09	0.13	5.6E-05	0.46
7	42	87	28	28	1.8E+09	4.8E-05	31	0.12	23	2.1E+09	0.14	3.9E-05	0.20
8	42	100	24	26	2.0E+09	3.6E-05	29	0.21	20	2.1E+09	0.09	2.3E-05	0.37
9	42	99	25	28	2.0E+09	1.2E-05	26	0.06	20	2.2E+09	0.09	1.3E-05	-0.06
10	42	98	24	26	1.7E+09	1.1E-04	28	0.16	20	2.0E+09	0.16	1.1E-04	0.01
11	63	156	24	27	2.1E+09	4.2E-04	27	0.14	19	2.4E+09	0.14	3.8E-04	0.08
12	63	155	25	27	2.2E+09	5.1E-04	29	0.15	20	2.5E+09	0.11	3.2E-04	0.37
13	63	157	26	25	2.2E+09	3.9E-05	29	0.10	20	2.6E+09	0.16	4.0E-05	-0.03
14	63	154	25	28	2.4E+09	3.1E-04	29	0.17	23	2.7E+09	0.13	2.3E-04	0.26
15	63	152	23	26	2.0E+09	2.9E-04	27	0.21	21	2.3E+09	0.14	2.0E-04	0.30
16	63	161	26	27	2.4E+09	6.6E-05	27	0.07	20	2.6E+09	0.08	6.6E-05	0.00
17	63	159	22	25	2.1E+09	2.1E-04	27	0.18	20	2.3E+09	0.11	1.9E-04	0.11
18	63	168	24	27	2.1E+09	1.5E-05	24	0.01	21	2.2E+09	0.07	1.7E-05	-0.12
19	63	159	23	27	2.3E+09	4.2E-04	26	0.12	22	2.5E+09	0.11	3.6E-04	0.14
20	63	163	23	27	2.3E+09	3.6E-04	26	0.11	22	2.6E+09	0.11	2.9E-04	0.18
21	80	164	25	27	2.1E+09	2.4E-04	27	0.07	23	2.3E+09	0.13	1.9E-04	0.19
22	80	162	23	26	2.0E+09	5.0E-04	26	0.12	23	2.3E+09	0.16	3.0E-04	0.41
23	80	162	25	27	2.0E+09	2.8E-04	27	0.07	22	2.4E+09	0.18	1.3E-04	0.54
24	80	176	21	25	2.0E+09	7.8E-04	23	0.10	23	2.3E+09	0.18	4.3E-04	0.45
25	80	147	28	27	2.0E+09	3.2E-04	31	0.14	22	2.5E+09	0.22	3.1E-04	0.04
26	80	152	24	28	2.0E+09	3.5E-04	28	0.13	24	2.2E+09	0.08	2.9E-04	0.18
27	80	158	23	26	2.2E+09	6.0E-04	27	0.16	22	2.5E+09	0.14	3.8E-04	0.37
28	80	160	26	27	2.3E+09	3.3E-04	29	0.11	24	2.5E+09	0.08	2.7E-04	0.18
29	80	162	23	26	2.0E+09	4.1E-04	26	0.16	24	2.4E+09	0.16	4.0E-04	0.03
30	80	160	24	27	2.0E+09	1.7E-04	26	0.06	23	2.3E+09	0.17	1.6E-04	0.10
AV.	-	-	-	27	-	-	-	0.13	21	-	0.13	-	0.19

To make it easier to compare, the simulation result of the non-adaptive SPF algorithm is repeated in both tables. The bottom line of each table is the average data. The improvement is calculated based on the simulation results of the non-adaptive SPF algorithm.

Improvement of Link Utilization for LSAR = (LSAR-SPF)/SPF

Improvement of Link Utilization for DWMAR = (DWMAR-SPF)/SPF

Improvement of Packet Drop Ratio for LSAR = (SPF-LSAR)/SPF

Improvement of Packet. Drop Ratio for LSAR = (SPF-DWMAR)/SPF

When compared with non-adaptive SPF, DWMAR and LSAR increase the average network capacity by 19% and 13% respectively. The average packet drop ratio is reduced by 28% and 19% respectively. The average link utilization is increased by 20% and 13% respectively. The link utilization variation is almost the same for DWMAR and LSAR, while both are lower than non-adaptive SPF. Using DWMAR can achieve about 50% more network capacity increase than fixed weight mapping adaptive routing (LSAR).

The simulation result of dynamic traffic almost repeats the results for static traffic. Furthermore, it substantially shows the increase of network capacity achieved by DWMAR, which proves our expectation.

4.4. Summary

The performance of DWMAR is tested by simulation and the simulation results are compared with non-adaptive SPF and LSAR. Both of the adaptive routing methods have an advantage over non-adaptive routing on network throughput, packet drop ratio, and routing convergence times. DWMAR has better performance than LSAR, which uses fixed weight mapping function.

Compared to non adaptive routing, adaptive routing increases the link utilization ratio.

the increase is consumed by the non-direct routes selected by adaptive routing.

The performance of DWMAR depends on the topology and the traffic pattern of the network. In some scenarios, adaptive routing even increases the packet drop ratio. However, most simulation results are positive. Furthermore, in the dynamic traffic simulation, DWMAR always has an advantage over LSAR on network throughput. So does LSAR over non-adaptive SPF. In general, the more nodes and links in the network, and the more alternative routes available from origins to destinations, the better the DWMAR performances.

Compared with LSAR, DWMAR is more stable when the traffic load is heavy. While it takes more time to converge when the traffic load is not heavy. DWMAR is more suitable to heavily loaded networks.

5. CONCLUSIONS

The main contribution of this thesis is to introduce the dynamic weight mapping function into adaptive routing and theoretically guarantee the stability of adaptive routing. The main differences between DWMAR and traditional adaptive routing are that

1. DWMAR abandons the universal weight mapping function and customizes weight mapping function for every link, which gives every link a greater chance to have a stable equilibrium point.
2. DMWAR purposely designs the weight mapping function and makes the link utilization concentrate on an effective section, thereby increasing the network capacity.

In this thesis, the dynamics of adaptive routing are qualitatively analyzed from the viewpoint of the feedback control system. Then the dynamic weight mapping adaptive routing (DWMAR) algorithm is proposed. DWMAR is a convergent adaptive routing algorithm which adjusts the weight mapping function of every link according to its utilization.

The performance of DWMAR is evaluated by simulation and compared with non-adaptive SPF and LSAR. The static traffic simulation tests the stable performance of routing protocol. DWMAR has a small advantage over LSAR in terms of packet drop ratio and network capacity, while both of the adaptive routing algorithms have a better performance than non-adaptive SPF. The dynamic traffic test simulates the traffic in the real Internet and has more realistic meanings. Compared to non-adaptive SPF, DWMAR increases the average network capacity by 19% and reduces the packet drop ratio by 28%, while the improvement for LSAR is 13% and 19%, respectively. DWMAR shows about 50% more increase in network capacity than LSAR. Compared with LSAR, DWMAR shows better stability when the network traffic is heavy. The convergence time of

DWMAR is impacted less by network traffic than LSAR.

The parameter setting of DWMAR is discussed in this thesis. There are many methods to alter weight mapping function responding to the link utilization change. DWMAR focuses on the major factors, and ignores the minor ones. Fixing certain minor parameters, such as maximum weight, upper bound of non-adaptive section, will simplify the algorithm without significant impact on the benefits of DWMAR. The parameter setting is also supported by simulation.

DWMAR is not designed to provide minimum end-to-end delay routing. The solution that DWMAR provides is not even optimal in link utilization and network throughput. However, it provides traffic balancing, which is lack in current dominant OSPF and IS-IS protocol. DWMAR is more stable on algorithm and compatible to Internet. Therefore, it shows better performance on Internet model than both non-adaptive routing and traditional adaptive routing. Compared with LSAR, it is possible that a particular traffic pattern, especially in a light loaded network, does not see better performance with DWMAR. This is because that DWMAR use more complicated adaptive logic. When the traffic load is heavy, the complicated adaptive logic helps to find a stable adaptive method. However, when the traffic load is light, the complicated logic takes more time to converge than the simple adaptive policy does in LSAR. DWMAR is designed to relieve the heavy traffic burden of the current Internet.

The implementation of DWMAR is relatively easy. It only requires a minor modification of the current link-state routing protocol. For example, as OSPF-TE, add an option in OSPF protocol will make DWMAR work. Furthermore, DWMAR does not have to be implemented on all the routers in the network. If only a few routers run DWMAR in the network, it will not ruin the routing function of the network, but would reduce some of the benefits compared with the pure DWMAR network. This feature makes it easy to transfer from non-adaptive SPF networks to DWMAR networks.

6. FUTURE WORK

To make the adaptive routing more flexible, it can work with DiffServ / IntServ. The routing algorithm can adjust the shape of the weight mapping function according to the packet type the interface received. For example, if most of the data on a link are real-time application data, the weight mapping function can be adjusted to avoid high link utilization, so that the data packets have a smaller possibility of experiencing a long queuing delay. Otherwise, if most of the data are delay insensitive, e.g. FTP data, the weight mapping function can be adjusted to achieve maximum link utilization.

To further study dynamic weight mapping adaptive routing, a mathematical model is required for the adaptive routing algorithm in packet switched data networks. It may be difficult to establish a pure mathematical model. A trade off is that of using Internet statistical data to establish an experiential model. Fuzzy logic will be helpful in building a more intelligent adaptive routing algorithm.

7. REFERENCES

[1] K. G. Ramakrishnan and Manoel A. Rodrigues, "Optimal Routing in Shortest-Path Data Networks", Bell Labs Technical Journal, Vol.6, No.1 .pp. 117-138, January – June 2001.

[2] Hao Wang, "Load-sensitive Adaptive Routing (LSAR) for Computer Networks, April 2003. Master Thesis of University of British Columbia

[3] Atul Khanna and John Zinky, "The Revised ARPANET Routing Metric", Proceedings of ACM SIGCOMM, pp. 45-56, 1989.

[4] Dimitri P. Bertsekas, "Dynamic Behavior of Shortest Path Routing Algorithms for Communication Networks", IEEE Transactions on Automatic Control, Vol. AC-27, pp. 60-74, 1982.

[5] Z. Wang and J. Crowcroft, "Analysis of Shortest-Path Routing Algorithms in a Dynamic Network Environment", ACM Computer Communication Review, Vol. 22, pp. 63-71, 1992.

[6] M. Ericsson, M. G. C. Resende and P. M. Pardalos, "A Genetic Algorithm for the Weight Setting Problem in OSPF Routing", Journal of Combinatorial Optimization, 2002.

[7] Eric J. Anderson and Thomas E. Anderson "On the Stability of Adaptive Routing in the Presence of Congestion Control", IEEE INFOCOM 2003, pp. 948-958, 2003

[8] Anwar Elwalid, Cheng Jin, Steven Low and Indra Widjaja, "MATE: multipath adaptive traffic engineering", Computer Networks 40, pp. 695–709, 2002

[9] Bernard Fortz and Mikkel Thorup, "Internet Traffic Engineering by Optimizing OSPF Weights", Proceedings of the INFOCOM 2000, pp. 519-528, 2000.

[10] M. Yuksel, B. Sikdar, K. S. Vastola and B. Szymanski, "Workload Generation for ns Simulations of Wide Area Networks and the Internet", Proc. Communication Networks and Distributed Systems Modeling and Simulation Conference, pp.

93-98, San Diego, CA, USA, 2000.

[11] Graig Labovitz, G. Robert Malan and Farnam Jahanian, "Internet Routing Instability", IEEE/ACM Transactions on Networking, Vol. 6, No. 5, pp. 515-527, October 1998.

[12] D. Gamarnik, "Stability of adaptive and nonadaptive packet routing policies in adversarial queuing networks", Society for Industrial and Applied Mathematics, Vol. 32, No. 2, pp. 371–385, 2003

[13] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: multipath adaptive traffic engineering", Vol. 40, Computer Networks, pp. 695–709, 2002

[14] V. Grout, J. Davies, M. Hughes and N. Houlden, "A New Distributed Link-State Routing Protocol with Enhanced Traffic Load Balancing", Centre for Applied Internet Research, University of Wales, 2004

[15] M. Kodialam, T. V. Lakshman, and S. Sengupta, "Online Multicast Routing With Bandwidth Guarantees: A New Approach Using Multicast Network Flow", Vol. 11, No. 4, IEEE/ACM Transactions on Networking, pp. 676-686, Aug. 2003

[16] J. Moy, "OSPF Version 2", Internet Engineering Task Force, RFC 2328, April 1998.
http://www.ietf.org/rfc/rfc2328.txt

[17] C.L. Hedrick, "Routing Information Protocol ", Internet Engineering Task Force , RFC 1058, Jun. 1988
http://www.ietf.org/rfc/rfc1058.txt

[18] G. Malkin, "RIP Version 2", Internet Engineering Task Force , RFC 2453, Nov. 1998
http://www.ietf.org/rfc/rfc2453.txt

[19] Cisco, "Interior Gateway Routing Protocol (IGRP)"
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/igrp.html

[20] Cisco, "Enhanced Interior Gateway Routing Protocol (EIGRP)"

http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/en_igrp.html

[21] Cisco, "Configuring of IS-IS for IP on Cisco Routers"
http://www.cisco.com/warp/public/97/is-is-ip-config.html

[22] Y. Rekhter, T. Li, "A Border Gateway Protocol 4 (BGP-4)", Internet Engineering Task Force , RFC 2453, Mar. 1995
http://www.ietf.org/rfc/rfc1771.txt

[23] Network Simulator 2 Manual
http://www.isi.edu/nsnam/ns/doc/.

[24] GT-ITM: Georgia Tech Internetwork Topology Models
http://www.cc.gatech.edu/fac/Ellen.Zegura/gt-itm/gi-itm.tar.gz

[25] E. Rosen, A. Viswanathan, R. Callon, "Multiprotocol Label Switching Architecture", Internet Engineering Task Force, RFC 3031, Jan. 2001.
http://www.ietf.org/rfc/rfc3031.txt

[26] B. Davie, J. Lawrence, K. McCloghrie, E. Rosen, G. Swallow, Y. Rekhter, P. Doolan, "MPLS using LDP and ATM VC Switching", Internet Engineering Task Force, RFC 3035, Jan. 2001.
http://www.ietf.org/rfc/rfc3035.txt

[27] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification" Internet Engineering Task Force, RFC 2205, Sep. 1997.
http://www.ietf.org/rfc/rfc2205.txt

[28] D. Katz, K. Kompella, and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2 ", Internet Engineering Task Force , RFC 3630, Sep. 2003
http://www.ietf.org/rfc/rfc3630.txt

[29] J. Postel, J. K. Reynolds, "File Transfer Protocol", Internet Engineering Task Force, RFC 959, Oct. 1985.
http://www.ietf.org/rfc/rfc0959.txt

[30] J. Postel, "Internet Protocol", Internet Engineering Task Force, RFC 791, Sep. 1981.
http://www.ietf.org/rfc/rfc0791.txt

[31] J. Postel, "Transmission Control Protocol", Internet Engineering Task Force, RFC 793, Sep. 1981.
http://www.ietf.org/rfc/rfc0793.txt

[32] J. Postel, "User Datagram Protocol", Internet Engineering Task Force, RFC 768, Aug. 1980.
http://www.ietf.org/rfc/rfc0768.txt

[33] J. Postel, "Internet Control Message Protocol", Internet Engineering Task Force, RFC 792, Sep. 1981.
http://www.ietf.org/rfc/rfc0792.txt

[34] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1 ", Internet Engineering Task Force, RFC 2616, Jun. 1999.
http://www.ietf.org/rfc/rfc2616.txt