EFFICIENT AND FAIR HIERARCHICAL PACKET SCHEDULING USING DYNAMIC DEFICIT ROUND ROBIN

Chin-Chi Wu[1,2] Chiou Moh[2] Hsien-Ming Wu[1,3] Ding-Jyh Tsaur[1,4] Woei Lin[1]
[1]Institute of Computer Science National Chung Hsing University
[2] Department of Information Management Nan Kai Institute of Technology
[3]Computer and Information Network Center National Chung Hsing University
[4]Department of Information Management Chin Min Institute of Technology
No. 250, Kuo Kuang Road, Taichung, Taiwan R.O.C
wcc007@nkc.edu.tw dianam@nkc.edu.tw woody@nchu.edu.tw djc@ms.chinmin.edu.tw wlin@nchu.edu.tw

ABSTRACT

This paper proposes a new hierarchical packet scheduling algorithm, HDDRR, enhanced from the existing dynamic deficit round-robin (DDRR) to provide relative differentiated service that enables support of delay-sensitive applications over the Internet. The level of service differentiation can be adjusted with parameter. The HDDRR scheduler fully utilizes the property of DDRR, therefore it can achieve high throughput efficiently and simultaneously provide smaller delay for short packets of each class. Simulation results showing the effectiveness of HDDRR are also presented.

KEY WORDS

Quality of Service, packet scheduling, dynamic deficit round-robin, relative differentiated service

1. Introduction

Traditionally, the Internet provides only the best effort service. However, with the introduction of ADSL, wireless WAN and LAN, and great enhancement in implementation of IP telephony, the volume of Internet traffic has rapidly increased. The provision of quality of service (QoS) has become an important issue.

In recent years, much literature [1-7] proposed mechanisms to provide QoS through packet scheduling. These approaches are mostly based on the InterServ [8] or DiffServ [9] model. Although many methods based on the InterServ model can provide absolute QoS guarantee but few of them are deployed commercially due to the limitation of scalability. Due to simplicity and scalability, the other type of approaches based on the DiffServ model received more attention.

In [3], Li stated that relative differentiated service (RDS) is a possible option to upgrade the current Internet when there is still a gap between the best-effort service and the DiffServ. The other reason for deployment of RDS is that the price of the RDS is cheaper than that of the DiffServ.

In the RDS model, there is no admission control and resource reservation support. Therefore, the RDS model does not provide absolute service guarantee. However, packets with high priority will receive better service than that of low priority. Users can choose the priority for their applications according to their priorities with regard to service requirements, cost or policies.

Some former methods such as the waiting time priority (WTP), the proportional average delays (PAD) and the hybrid proportional delay (HPD) proposed in [1] are representatives supporting the RDS model. The key characteristic of these three approaches is that they determine the scheduling priority by time-stamping each arriving packet and computing the waiting time of the head-of-line (HOL) packets.

Li et. al. [3] stated that choosing a packet with the smallest time stamp can cause a bottleneck. Therefore, some other measurement-based approaches emphasizing accurate control, small overhead or simplicity are proposed in [3-6]. These approaches update the priority control parameters or service rates periodically according to the continuous monitoring of the packet arrival rates and/or queue lengths.

Although these measurement-based approaches can also achieve a good performance in delay differentiation, the traffic monitoring and periodical computation for service rate control will introduce bookkeeping overhead. In [2], Lai et. al. stated that another shortcoming of some measurement-based approaches is that a small change in the class load distribution is likely to significantly affect the delay differentiation. Therefore, these approaches must tradeoff between the overhead for service rate adaptation and the accuracy for quick response to changes of traffic loads.

In the aforementioned research of RDS, most of the examples assume that the arriving packets are classified and aggregated according to packet priorities. These approaches seldom consider the packet length and give different treatment in scheduling except the schedulers [2] proposed by Lai. et. al. In [2], Lai and Li described that when the packet length of a packet is considered and the scheduling follows the shortest job first (SJF) rule, the scheduler can yield an optimal queueing delay. So a scheduler such as the AWTP [2] which obeys this rule can reduce the overall average waiting time.

The dynamic deficit round-robin (DDRR) scheduler [10] is another type of scheduler for the conventional best-effort service which provides delay differentiation according to packet length. The DDRR scheduler is enhanced from the deficit round-robin (DRR) [11] scheduling. In addition to low complexity, DDRR also provides max-min fair share, high throughput and small delay for short packets.

The property of small delay for short packets in DDRR can avoid degradation in QoS for some delay-sensitive applications. Because a lot of short packets are generated by delay-sensitive applications such as VoIP, the degradation in short-packet delay is likely to significantly affect the quality of service [10].

While AWTP considers the HOL packet lengths among classes, this does not include the situation where packets of the same priority but from different subscribers or links arrive simultaneously to provide better service for short packets. We assume that, in most applications, the packets of the same application have the same priority. Therefore, designing a scheduler which provides RDS between classes and considers the SJF rule for both reducing the queueing delay and ensuring small delay for the short packets of the same class is a challenge.

In this paper, we propose a new hierarchical dynamic deficit round-robin (HDDRR) scheduler which is an enhancement of DDRR. We introduce a new queueing system to DDRR so that the new scheduler, HDDRR, does not only support RDS but also ensure that the short packets of each class experience relatively small delay.

Even though the traffic load of each class varies as time, the HDDRR scheduler can still provide relative delay differentiation between classes. The HDDRR scheduler also allows the network administrator to adjust the RDS between classes based on pricing or policy requirements.

The structure of this paper is as follows. Section 2 reviews the background relevant to the DDRR scheduler. Section 3 proposes the HDDRR scheduler. Section 4 examines the performance and overhead of the HDDRR scheduling algorithm via simulation. Finally, we conclude in Section 5.

2. Background

The DDRR scheduler [10] is originally designed for a 5-Tb/s switch and is responsible for arbitrating the emission of packets to the switch element at the next stage. In order to achieve small delay for short packets, max-min fair share and also provision of high throughput, Yamakoshi el. al. devised the DDRR scheduler.

The DDRR is derived from the DRR scheduling algorithm. Both the two packet scheduling algorithms rely on the use of deficit counters. The main difference is that the DDRR scheduler dynamically changes the granularity for the deficit counters instead of using a fixed granularity as in the DRR.

We describe the DDRR algorithm and define the used notations as follows. Assume that there are N queues composed of variable-length packets. The ranges of the following subscripts i,j are $0 \le i \le N-1$ and $j>0$, respectively.

- $L_{i,j}$: the HOL packet length of queue i in the round j
- $D_{i,j}$: the value of deficit counter for queue i in the round j. The initial value $D_{i,0}$ is 0.
- $M_{i,j}$: difference of $L_{i,j}$ and $D_{i,j-1}$
- $M_{min,j}$: Suppose that the minimum value among $M_{i,j}$ is with an index min (i.e., of the queue min) in the round j.
- ∞ : a very large value

```
Algorithm DDRR
Step1. Calculate $M_{i,j}$
        (a) $M_{i,j} = L_{i,j} - D_{i,j-1}$ for nonempty queue $i$, or
        (b) $M_{i,j} = \infty$    for empty queue $i$
Step2. Select the minimum $M_{i,j}$ (i.e., $M_{min,j}$)
        $M_{min,j} = min\ \{ M_{i,j},\ 0 \le i \le N-1 \}$
Step3. Update $D_{i,j}$
        (a) $D_{i,j} = D_{i,j-1} + M_{min,j}$ for nonempty queue $i$ and
            $i \ne min$, or
        (b) $D_{i,j} = 0$   for empty queue $i$ or $i=min$
```

The $M_{min,j}$ determined in step 2 is the scheduling granularity of the round j. The packet at the head of queue min is selected to dispatch and the granularity is changed as the HOL packet length. Figure 1 shows the schematic architecture of the DDRR scheduler working on three queues with variable-length packets. Table 1 shows the scheduling by DDRR.

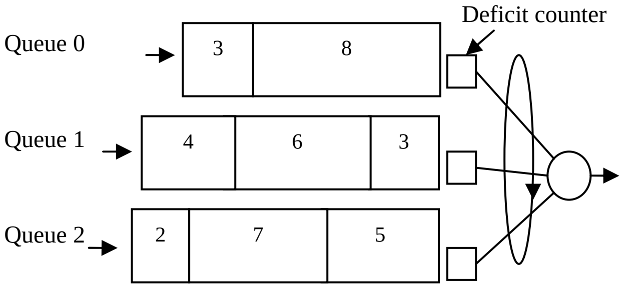

Figure 1. The schematic architecture of the DDRR scheduler

Table 1. Operations of DDRR scheduling

Step	Sequence	1	2	3	4
1	$M_{0,j}$	8=8-0	5=8-3	3=8-5	3=3-0
	$M_{1,j}$	3=3-0	6=6-0	4=6-2	1=6-5
	$M_{2,j}$	5=5-0	2=5-3	7=7-0	4=7-3
2	min/ $M_{min,j}$	1/3	2/2	0/3	1/1
3	$D_{0,j}$	3=0+3	5=3+2	0	1=0+1
	$D_{1,j}$	0	2=0+2	5=2+3	0
	$D_{2,j}$	3=0+3	0	3=0+3	4=3+1

3. HDDRR Scheduler

The HDDRR scheduler is supposed to operate under the following conditions. The first is that the arriving packets from each link are not aggregated together according to their priority. For each link, the router creates one packet queue for each priority. The second is that the packets generated by the same application must be with the same priority. Figure 2 shows the schematic architecture of the HDDRR scheduler.

Assume there are *n* links (from *0* to *n-1*) offering traffic and the traffic can be classified into *m* classes (from *0* to *m-1*). For simplicity, we illustrate the architecture and operations of HDDRR with *2* links and *2* classes of traffic. The classes *0* and *1* stand for the highest priority and the lowest priority (best-effort), respectively. The HDDRR is primarily composed of two DDRR schedulers, a packet queueing system and a token queueing system. We describe the functions of each component as follows.

(1) Packet queue (PQ)
PQs store the arrived packets which can not be transmitted immediately. The router sets up *m* PQs for each link. The PQ identifier *Q(i,j)* represents the queue which stores the arriving packet of priority *i* and coming from link *j*. We define *j* as *source link id.* For example, the PQs *Q(0,0)* and *Q(1,0)* store the packets of class 0 and class 1 from source link 0, respectively.

(2) Token queue (TQ)
TQs store tokens which are generated for each arriving packet. A token includes the information, *packet weight* and *source link id*, for DDRR-2 scheduler to determine which packet should be served next. Five bytes memory space is enough for a token to save the information. The packet weight is a function of packet length and service differentiation parameter. Through proper definition of packet weight, the HDDRR scheduler can simultaneously achieve RDS and small delay for short packets. We discuss this in later section. The token may come either from the packet classifier or DDRR-1 scheduler. The router sets up *n* class-0 TQs (TQ 0 and TQ 1 herein) for the class-0 traffic of each link, and *1* class-1 TQ (TQ 2) for the class-1 traffic from all links.

(3) Deficit counter (DC)
The router sets up a DC for every TQ. In addition, the router also sets up DCs for the PQs which store the best-effort (class-1) packets.

(4) Packet classifier (PC)
Whenever any packet arrives, the PC identifies the packet priority and enqueues the packet according to packet priority and source link id. If the packet belongs to the class-0 traffic, the PC generates a corresponding token immediately and enqueues the token to the class-0 TQ according to the source link id.

If the packet is of the class 1, the PC then examines whether any token representing the best-effort traffic exists in TQ 2. If there is no token in TQ 2, the PC then generates and enqueues a token for the arriving packet. Otherwise, the PC just enqueues the arriving packet.

(5) DDRR-2 scheduler
The DDRR-2 scheduler is not only responsible for ensuring short packets of high-class traffic experience small delay but also for achieving delay differentiation between classes. The packet scheduling by DDRR-2 includes two steps. The first step is to execute the DDRR algorithm on the TQs to choose the token which has the minimum difference between the packet weight and the DC. The second is to really dispatch a packet from the PQ according to the source link id of the selected token.

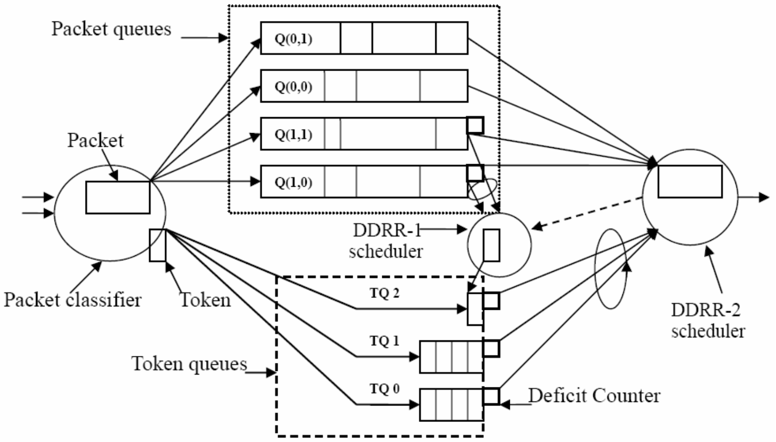

Figure 2. The schematic architecture of the HDDRR scheduler

If the token is selected from the class-0 TQs (TQ 0 or TQ 1), the DDRR-2 scheduler dispatches the HOL packet from the corresponding class-0 PQ (with the same linkid). Otherwise, if the token is selected from class-1 TQ (TQ 2), the DDRR-2 scheduler dispatches the HOL packet from the class-1 PQ specified by the source link id of the chosen token. Finally, the selected token is removed. However, after DDRR-2 dispatches the class-1 packet, the DDRR-1 scheduler is invoked by DDRR-2 to determine which HOL packet of class-1 PQs should be served next and enqueue a corresponding token to TQ 2.

(6) DDRR-1 scheduler

The DDRR-1 scheduler is responsible for achieving small delay for short packets of best-effort traffic. DDRR-1 scheduler works on the class-1 PQs and selects packets according to the HOL packet lengths and DCs. The detailed operations have been described in Section 2. But DDRR-1 scheduler does not dispatch a packet, it only generates and enqueues a token for the selected packet.

4. Simulations

In the Section 3, we have stated the assumptions and described the operations of HDDRR. There are two important factors which influence the performance of the HDDRR scheduler. The first is the token queueing system. When the number of source links and/or service classes increases, how the token queues influence the performance of the HDDRR scheduler will be left as a future work. The second factor is the packet weight defined for each packet. In this paper, we only examine the effect of packet weights.

In [10], the DDRR scheduler has demonstrated that the packet delay decreases linearly as the packet length decreases. Therefore, the packet weight for HDDRR is defined as a function of packet length and differentiated service parameter. We describe the notations and the packet weight function W_C as follows.

- C : the class of traffic
- L : packet length
- DSR_C : differentiated service parameter for class C

$$W_C(L,DSR_C) = L/DSR_C \qquad (1)$$

A. Traffic Model and Parameters

For simplicity, we only define 2 classes of traffic and assume 4 source links offering traffic. The traffic offered among the four links is with uniform distribution. The packet length is normalized as [10] to an integer number of fixed-size cell times and its range is from 1 to 32. The lengths of packets are assumed to have an exponential distribution with the mean length being 10 cells. The interarrival time between packets is also with exponential distribution. The DSR_1 is defined with 1 and DSR_0 (abbreviated as DSR) is defined with 1, 2, 4 and 8. We also define three cases of class load distribution (30% v.s. 70%, 50% v.s. 50% and 70% v.s. 30 % for class 0 and class 1 respectively) to examine the performance under different load conditions.

B. Simulation Results

Figures 3, 4 and 5 show the average packet delay ratios for class 1 relative to class 0 with various *DSR*. The HDDRR scheduler always achieves the average packet delay of high priority (class 0) smaller than that of low priority (class 1) and the packet delay ratios also increase as the *DSR* increases, especially under a heavy load.

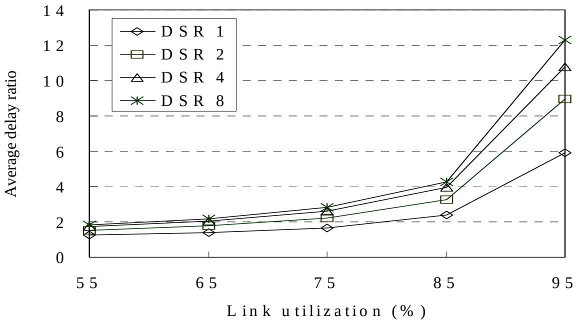

Figure 3. Average packet delay ratio under class 0/class1 load ratio 30/70

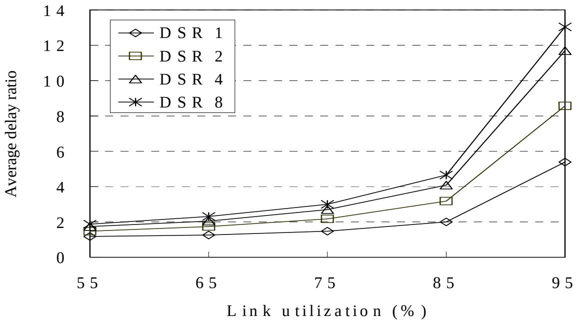

Figure 4. Average packet delay ratio under class 0/class1 load ratio 50/50

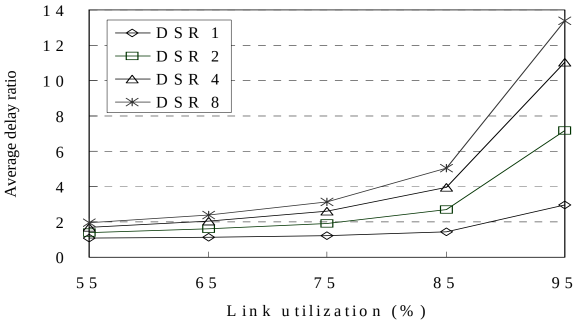

Figure 5. Average packet delay ratio under class 0/class1 load ratio 70/30

Figures 6, 7 and 8 show the average packet delays varying with the packet lengths for the case of two classes of traffic, high link utilization (95%) and various class load distributions. The average delay of short packets for each class is relatively smaller than that of long packets. This is a very important feature of HDDRR scheduling. This feature can ensure providing better service for the short packets of each class. Reviewing Figures 3-8, it reveals that the HDDRR scheduler achieves the goals of supporting the RDS model and providing relatively small delay for short packets of the same class.

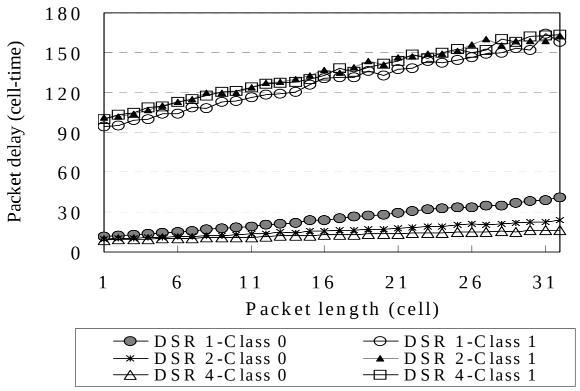

Figure 6. Packet delay versus packet length under class 0/class1 load ratio 30/70

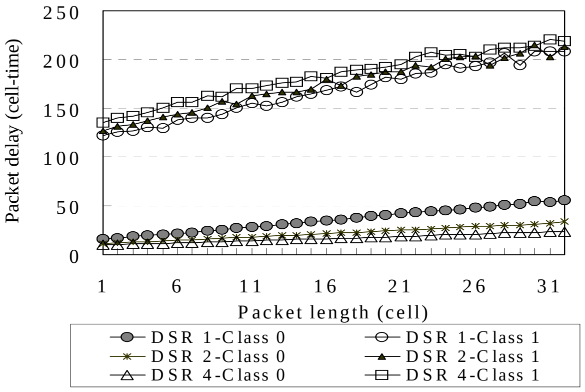

Figure 7. Packet delay versus packet length under class 0/class1 load ratio 50/50

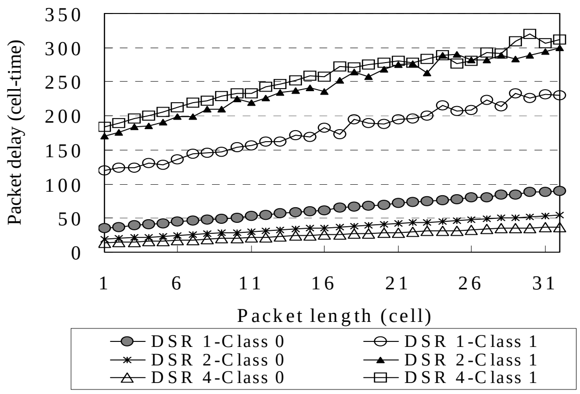

Figure 8. Packet delay versus packet length under class 0/class1 load ratio 70/30

Due to space limitations and the traffic offered from 4 source links being with uniform distribution, we only show the maximum packet queue lengths of source link 0 under different load conditions and with various DSR in Figures 9, 10 and 11. Figures 9, 10 and 11 indicate that the maximum packet queue lengths of class 0 are smaller than that of class 1 under various *DSR* and link utilization. In the case of *DSR* greater than 1, the packet weight of class 0 is smaller than that of class 1, the queues of class 0 have better chance to send out a packet.

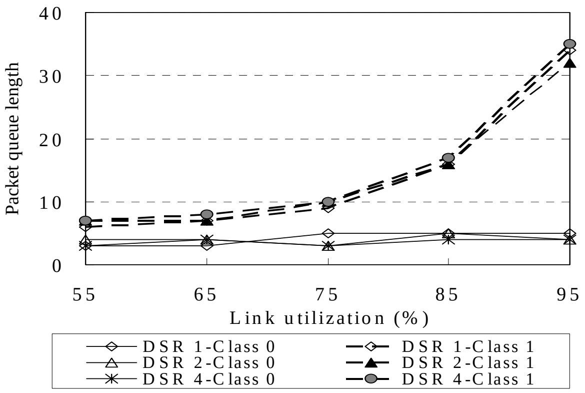

Figure 9. Packet queue lengths of source link 0 under class 0/class 1 load ratio 30/70

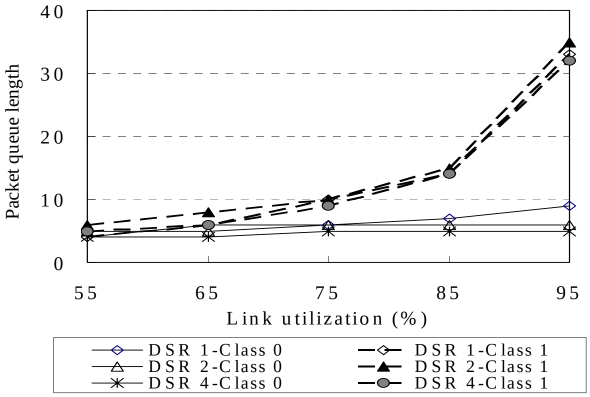

Figure 10. Packet queue lengths of source link 0 under class 0/class 1 load ratio 50/50

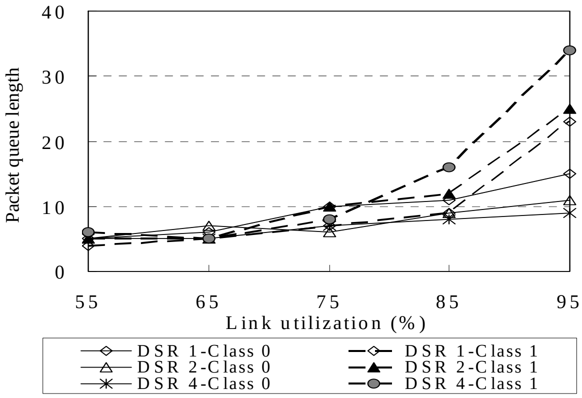

Figure 11. Packet queue lengths of source link 0 under class 0/class 1 load ratio 70/30

Even though the *DSR* is 1, each class-0 packet queue has a dedicated token queue in scheduling while the class-1 packet queues of all links must share a token queue. Therefore, the packet queues of class 0 usually have a higher probability to send out a packet.

Figure 12 shows that the extra buffer space requirement for the token queues is very small even under high link utilization. Therefore, the token queues can be implemented with the highest-speed cache to reduce the access time.

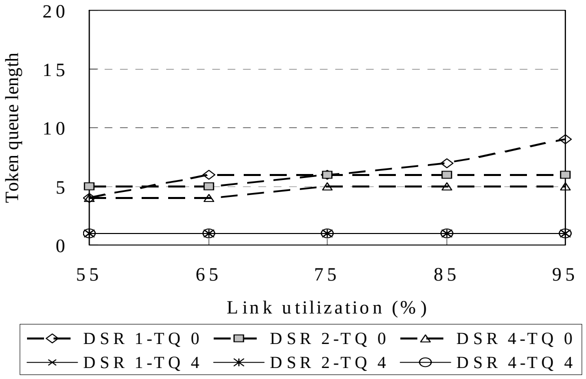

Figure 12. Maximum token queue lengths under class 0/class 1 load ratio 50/50

C. Upper Bound of Class-1 Packet Waiting for Scheduling

Based on the packet weight function of (1) and the above assumptions, we calculate the upper bound, U_w, of the waiting time for the class-1 packet when it has been selected by the DDRR-1 scheduler and is ready to be dispatched.

In the worst case, all the packets from class 0 and class 1 are assumed with the smallest, L_{min}, and largest, L_{max}, packet length, respectively and all the class-0 packet queues always have packets awaiting dispatch. Assume that R is the ratio of DSR_0/DSR_1. The DC value of TQ N increase L_{min}/R every N rounds of scheduling by DDRR-2 scheduler, where N is the number of source links. It is easily to derive that the upper bound U_w as follows.

$$U_w = R*L_{max}/L_{min} * N \qquad (2)$$

Therefore, based on (2), we list the worst-case upper bounds for various DSR_0 in Table 2. Table 2 reveals that, we can adjust the RDS via the differentiated service parameter *DSR*.

Table 2. The worst-case upper bounds of waiting time for class-1 packets $N = 4, L_{min} =1, L_{max} =32$

DSR_0	1	2	4	8
U_w (cell-time)	128	256	512	1024

5. Conclusions

In this paper, in order to support relative differentiated service and provide small delay for short packets of the same class, we develop a hierarchical scheduler HDDRR. Simulation results show that the HDDRR scheduler performs very well under various load conditions and the service differentiation can be adjusted with parameter. We also show that the extra buffer requirement for the token queues is very small. In addition, the time complexity of the HDDRR is *O(N)*, where *N* is the number of links, which is minor for each packet scheduling. In addition, due to the nature of max-min fair share of DDRR, HDDRR can also achieve fair share among queues of the same priority. Therefore, the HDDRR is an efficient and fair scheduler. This hierarchical scheduling model can be extended to include more classes and links, and we leave this investigation as a future work.

References

[1] C. Dovrolis, D. Stiliadis, P. Ramanathan, Proportional Differentiated Services: Delay Differentiation and Packet Scheduling, *IEEE/ACM Transactions on Networking, 10,* Feb. 2002, 12-26.

[2] Lai, Y.-C.; Li, W.-H., High-performance scheduler to achieve proportional delay differentiation, *Proc. IEE Communications, 150* , Issue: 3 , June 2003, 153-158.

[3] Li, Z.G.; Chen, C.; Soh, Y.C., Relative differentiated delay service: time varying deficit round robin, *Proc. 5th World Congress on Intelligent Control and Automation, 6* , 15-19 June 2004, 5608-5612.

[4] Angelos Michalas, Paraskevi Fafali, Malamati Louta, Vassilios Loumos, Proportional Delay Differentiation Employing the CBQ Service Discipline, *Proc. 7th International Conference on Telecommunications,* 11-13 June 2003, 483-489.

[5] Y. Moret and S. Fdida, A Proportional Queue Control Mechanism to Provide Differentiated Services, *Proc. International Symposium on Computer and Information Systems (ISCIS)*, Oct. 1998.

[6] Wei, J.; Li, Q.; Xu, C.-Z., VirtualLength: a new packet scheduling algorithm for proportional delay differentiation, *Proc. 12th International Conference on Computer Communications and Networks,* 20-22 Oct. 2003, 331-336.

[7] Chin-Chi Wu, Hsien-Ming Wu, Chia-Lung Liu, Woei Lin, A DDRR-Based Scheduler to Achieve Proportional Delay Differentiation in Terabit Network, *Proc. 19th International Conference on Advanced Information Networking and Applications, 2,* 28-30 March 2005, 347-350.

[8] R. Braden, D. Clark, and S. Shenker, Integrated Services in the Internet Architecture : An Overview, *RFC 1633*, July 1994.

[9] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, An Architecture for Differentiated Services, *RFC 2475,* Dec. 1998.

[10] K. Yamakoshi, E. Oki, and N. Yamanaka, Dynamic deficit round-robin scheduler for 5-Tb/s switch using wavelength routing, *Proc. 2002 Merging Optical and IP Technologies Workshop on High Performance Switching and Routing*, 26-29 May 2002, 204-208.

[11] M. Shreedhar, G. Varghese, Efficient fair queuing using deficit round-robin, *IEEE/ACM Transactions on Networking, 4* , Issue 3 , June 1996, 375-385.