

Performance Analysis of RR and FQ Algorithms in Reconfigurable Routers

Daniele Ludovici and Stephan Wong

Computer Engineering Laboratory

Electrical Engineering, Mathematics and Computer Science Department

Delft University of Technology

Makelweg 4, 2628 CD, Delft, The Netherlands

{ludan, stephan}@dutepp0.et.tudelft.nl

Abstract—Currently, we are witnessing a trend in network routers to include FPGA structures to provide flexibility in scheduler implementations at improved performance levels when compared to software-only implementations. This permits run-time reconfiguration of the scheduling algorithm utilized to adapt to changing network scenarios. In this paper, we introduce a reconfiguration model in the NS-2 (Network Simulator-2), to investigate the possibility to improve upon end-to-end delay, jitter, and throughput by exploiting the availability of a flexible hardware structure such as an FPGA. The reconfiguration model is created by modifying the existing implementation of the Nortel Diffserv module in the simulator. Our approach entails modeling different realistic network scenarios in conjunction with several scheduling algorithms to determine their performance (in terms of delay, jitter, and throughput). In addition, the implementation of certain scheduling algorithms do not require as much hardware area as other, possibly leading to the support of additional streams. The obtained results confirm that performance gains can be achieved when reconfigurable hardware is used to implement scheduling algorithms.

Keywords—Reconfigurable Computing, NS-2 Network Simulator, Router, FPGA

I. INTRODUCTION

Internet and computer networks are becoming part of our daily life, a change that affects universities, companies, commercial organizations, and also the normal end-user. These networks support a huge amount of different applications like e-mail, web, audio, and video services, including television on demand and file transfer. Furthermore, the demand for these these applications are growing quickly and probably this fact will continue in the near future. Recently, the provision of service guarantees has become an important issue, which was caused by both the heterogeneity of application requirements and the growing commercialization of the Internet. Selling products as television-on-demand or audio streaming with a network infrastructure that does not give guaran-

tees on the service level desired, is becoming difficult. Currently, packet traffic on the Internet is mostly still based on the best-effort model, which means “as much as possible as soon as possible”. Consequently, each packet is treated equally, i.e., no priorities are assigned to packets, while being transmitted over the Internet. This model suited well the more traditional Internet applications like web, e-mail, newsgroup, etc. On the other hand, new multimedia-related applications demand a differentiation in the levels of quality of service in packet traffic. The main reason is that, the traffic characteristics and its requirements for network transport functionality may vary. More specifically, different applications have varying requirements in terms of throughput, delay, and jitter. For example, telnet sessions generate low traffic but have high interactivity requirements. FTP sessions send bursts of kilobytes or even megabytes and the throughput represents a fundamental aspect compared with the interactivity. HTTP transactions open a transport connection to transmit a handful of packets but do not need an high bandwidth utilization. Multimedia applications, like audio and video streams, require a good bandwidth utilization, short delays as well as low jitter. If the Internet handles all data in the same way, then it can result in unacceptable, if not completely unusable, system. Therefore, differentiation of quality of service for particular packet traffic is needed. There is the necessity to handle this kind of situations at router level. Routers process the incoming packets according to the rules specified in protocols and must use hardware resources to do so. Given the different ways of packet processing, flexibility has become a key design issue leading to the introduction of reconfigurable hardware in routers next to the network processors. The presence of reconfigurable hardware allows the hardware to adapt itself to the incoming packet traffic, e.g., change to the scheduling algorithm

best suited for a particular mix of packet traffic.

In this paper, we investigate whether end-to-end delays, jitter, and throughput can be improved by exploiting the availability of a flexible hardware structure such as an FPGA. The main idea is to use reconfigurable hardware to obtain a more flexible scheduling discipline to do so. The main results show how it is possible to improve the performance of a certain kind of streams by changing the scheduling algorithm accordingly to some conditions within the network as well as how to find a good compromise between the performance desired for a kind of traffic stream and the overall number of the streams that the router is able to handle.

This paper is organized as follows. Section II describes the background on reconfigurable hardware and scheduling algorithms. Section III presents the detailed implementation of the reconfiguration mechanism discussing two different approaches to utilize it. Section IV presents the results of the simulations. Section V summarizes the conclusions and gives some future directions.

II. BACKGROUND

This section presents the background of this paper from two aspects: reconfigurable hardware and scheduling algorithms.

A. Reconfigurable Hardware

High-speed links have brought a strong impetus on the need for *fast* routers. These devices have to be fast enough to switch packets from incoming links into one of the outgoing link at a speed that matches the available link speed [4]. Nowadays, there is also need for *flexibility* because different kind of traffics, with different performance requirements, flow into the same network mixed together. Currently, reconfigurable hardware is a good solution to address both these problems. Reconfigurable hardware allows its own functionality to change in response to the demands placed upon the system while it is running. This gives us both flexibility and the performance of specialized hardware.

B. Scheduling Disciplines

The selection of an appropriate scheduling algorithm is a key point to build a network environment with QoS capabilities. Consider a simple case: a service provider has to share the output bandwidth among all data flows such that each flow obtains a fair portion of the bandwidth resources. Therefore,

we are assuming that each single flow has the same requirements. However, due to the diversity of the existent applications, a data flow may require a certain minimum amount of the output bandwidth, and this can lead to unequal bandwidth allocation between flows. Therefore, a service provider has to choose the correct disciplines to ensure that the requirements of all the data flows are met. A service discipline can be classified as either *work-conserving* or *non-work-conserving* [8]. With a work-conserving discipline, a scheduler is never idle when there is a packet to send. With a non-work-conserving discipline, each packet is assigned an eligible time, this parameter represents the time for the packet departure and if no packet is eligible, none will be transmitted even when the scheduler is idle; with such kind of disciplines, a scheduler can be idle at any time, in an effort to smooth out the traffic pattern [1]. At the moment, most manufactures of the telecommunication equipment rely upon work-conserving disciplines. The main reason is that they allow to utilize bandwidth resources more efficiently. Furthermore, two QoS frameworks proposed by the IETF rely upon the work-conserving schedulers [7]. Hence, we will focus on this class of scheduling disciplines and in particular on weighted round robin and weighted fair queueing.

B.1 Weighted Round Robin (WRR)

The WRR scheduler works in a cyclic manner, serving consecutively the input queues. The weight is a variable that indicates how many packets have to be sent in each cycle from each queue. If a queue has fewer packets than the value of the weight, the WRR scheduler outputs the existent number of packets and begins to serve the next queue. The WRR scheduler does not take the size of the transmitted packets into account. As a result, it is difficult to predict the actual bandwidth that each queue obtains. In other words, it is difficult to use only the weight values as the means to specify the amount of the output bandwidth. Suppose that w_i is the value of the weight associated with the i th queue. If L_i is the mean packet size of the i th input queue, then $w_i L_i$ bytes of data are sent during each cycle on average. If there are m input queues, then it is easy to show that the average amount of data transmitted from all queues during one cycle can be approximated by:

$$\sum_{i=1}^m w_i L_i \quad (1)$$

Expression 1 is referred to as the *frame size*. Tak-

ing the mean packet size and weights of all queues into account, it is possible to approximate the output bandwidth for the given k th queue:

$$\frac{w_k L_k}{\sum_i w_i L_i} B, \quad k \in (1, m) \quad (2)$$

where B specifies the output bandwidth of an interface, on which a router implements WRR. By approximating the average bandwidth of each queue, it is possible to provide the QoS guarantees [7].

B.2 Weighted Fair Queuing (WFQ)

Investigation into fair network resource allocation has led to the development of a class of algorithms that provide tight end-to-end delay bounds and efficient resource utilization. These algorithms attempt to approximate the ideal behavior of the Generalized Processor Sharing algorithm (GPS). GPS [11] has been proposed by Parekh and Gallager in 1993, it is based on a fluid model, so it assumes that the input traffic is infinitely divisible and that all sessions can be served at the same time. This is a work-conserving server and it guarantees each session to receive a *service rate* g_i of at least:

$$g_i = \frac{\theta_i}{\sum_{j=1}^N \theta_j} r \quad (3)$$

where r is the *server rate* and θ_i is the *weight* for the i -th session. GPS treats each packet as infinitely divisible, therefore, the scheduler picks a small piece of data from each session and transmits it to the output link. GPS is based on a fluid model, that is, it can transmit packets from different queues simultaneously instead of sending one packet per time separately. For example, having two sessions m and n , each of which with weight 1, an algorithm like *Bit by Bit Round Robin* transmits one bit of the first session and a bit of the second session. Instead, GPS always uses half bandwidth transmitting session m data and the other half transmitting session n data. GPS is a theoretical model and cannot be implemented in real. One of the first scheduling algorithm proposed to approximate GPS was WFQ, it schedules packets according to their arrival time, size, and the associated weight. Upon the arrival of a new packet, a “virtual finish time” is calculated and the packet is scheduled for departure in the right order with respect to the other packets. This “virtual finish time” represents the time at which the same packet would finish to be served in the GPS system. Subsequently, WFQ outputs packets in the ascending order of the virtual finish time. Such

an approach enables the sharing of resources between service classes in a fair and predictable way. Furthermore, it is possible to estimate the bandwidth allocation and the worst-case delay performance, which makes the use of the WFQ discipline very attractive for the provision of QoS, and especially for the provision of the end-to-end guarantees. Suppose that B is the total throughput of an output link on which a router implements WFQ. When all sessions of the WFQ scheduler are active, then each class receives a portion of the total bandwidth, which is determined by its weight w_i and is equal to $w_i B$, (B is the available bandwidth). Hence, to simplify the expression, we assume that it holds for all weights w_i that

$$\sum_i w_i = 1, \quad w_i \in (0, 1) \quad (4)$$

By knowing the QoS requirements of all data flows, we can find values for w_i such that all the QoS guarantees are ensured.

III. RECONFIGURATION MECHANISM

Our investigation entails the exploration into the viability of using reconfigurable hardware to improve the functionality and in turn performance of the scheduler when facing different incoming packet traffic. From literature, implementation of different scheduling algorithms have different area requirements. Consequently, reconfigurable hardware area can be saved when not needed and can then be used for other purposes. In addition, it can be used to improve the performance of the current scheduling algorithm by supporting more streams. Within many of different approaches proposed in literature, Weighted Round Robin (WRR) and Weighted Fair Queuing (WFQ) are perhaps the two most adopted disciplines [10]. These algorithms can be implemented in hardware to speed up some functions as well as to give the possibility of scheduler reconfiguration at run-time. Data about the required area in FPGA are taken from the implementation on a Virtex II device. A simple comparison from [10] and [9] in terms of CLBs shows us that it is possible to save almost 70% of FPGA area depending on the chosen algorithm. This leads to the following challenges. Is it possible to devise to a more dynamic handling of the traffic, to handle more stream sessions with a different scheduling algorithm? Is it possible to find a tradeoff between the utilized FPGAs area and the performance desired?

In our simulations, a multimedia stream traffic source (i.e., MPEG4 [6]) was utilized. In the NS-2

implementation, every kind of traffic is identified by a different packet type. During the queuing phase, the packet type is recognized and therefore the scheduling algorithm is changed.

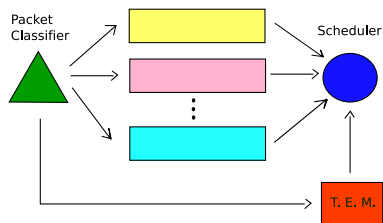


Fig. 1. Reconfiguration Idea

This operation is performed in the T.E.M (Traffic Evaluation Module) component, depicted in Figure 1 that we introduced. It is possible to do this mainly in two different ways. First, a static threshold can be set and the scheduling algorithm is changed in hardware when it is exceeded. Second, a ratio between different packet streams within a specific time period can be set to determine whether to switch the scheduling algorithm or not. The second approach is more accurate, but choosing the size of the interval and the desired threshold represents a challenge for further investigation.

In this section, a comparison between the two methodologies is presented. To add reconfiguration support in the simulator, the Diffserv module has been modified. In particular, the enqueue function has been altered to monitor, keep track of the kind of traffic passing into the network, and change the scheduling algorithm accordingly. Two main approaches have been adopted:

1. *static threshold*: the reconfiguration happens after a fixed number of video packets has been received and forwarded by the core router (dropped packets are not taken in account). This approach lacks in flexibility and it is not suitable for a complex traffic situation, because there could be a long period without multimedia transmission inside the backbone, but it may be possible to have a big variation between activity and inactivity interval of the sender. Of course, this approach presents a very easy implementation and does not introduce computational overhead for the router. The major problem is that, whether the threshold is passed, the system has no way to recognize if the MPEG4 traffic is continuing to flow. It would be possible to set a timer and try to understand if such kind of traffic is not passing anymore, but it would be another challenge to choice the value of this deadline.
2. *average load in a period*: this approach is more ac-

curate as a counter for each kind of packet is kept. I.e., every N seconds a check on the ratio between the number of video packet, and the number of the other packet, is calculated and if this ratio goes beyond a certain threshold, then the scheduling algorithm is changed. Dimensioning the threshold is also a challenge, but it is easier than the first approach and it is foreseeable to devise a solution with a movable threshold under specific conditions.

To validate this modification, Table I presents the results of a set of simulations in which the use of Round Robin, Weighted Round Robin, and Round Robin changing into Weighted Round Robin are compared. In the first simulation, every queue has been served by a Round Robin scheduler without the modification. In the second, the modification has been applied and during the simulation run changing the scheduling algorithm from RR to WRR. In the last simulation, the scheduler is set on WRR. For the second and the last one, the queue weights are set as presented in Table I.

| Delay(s) | RR | RR→WRR | WRR | Weight |
|----------|--------|--------|--------|--------|
| Poisson1 | 0.1881 | 0.2329 | 0.3076 | 3 |
| Poisson2 | 0.1885 | 0.1386 | 0.0952 | 7 |
| MPEG4 | 0.1717 | 0.1050 | 0.0520 | 10 |

TABLE I
SET OF SIMULATIONS WITH RECONFIGURATION

| Packet Drop | TotPkts | TxPkts | link-drops |
|-------------|---------|--------|------------|
| All | 110451 | 88655 | 21796 |
| Poisson1 | 38174 | 29749 | 8425 |
| Poisson2 | 37809 | 29746 | 8063 |
| MPEG4 | 34468 | 29160 | 5308 |

TABLE II
PACKET DROP FOR ROUND ROBIN

| Packet Drop | TotPkts | TxPkts | link-drops |
|-------------|---------|--------|------------|
| All | 110451 | 88779 | 21672 |
| Poisson1 | 38174 | 23435 | 14739 |
| Poisson2 | 37809 | 33468 | 4341 |
| MPEG4 | 34468 | 31876 | 2592 |

TABLE III
PACKET DROP FOR RR → WRR

| Packet Drop | TotPkts | TxPkts | link-drops |
|-------------|---------|--------|------------|
| All | 110451 | 88958 | 21493 |
| Poisson1 | 38174 | 17412 | 20762 |
| Poisson2 | 37809 | 37078 | 731 |
| MPEG4 | 34468 | 34468 | 0 |

TABLE IV
PACKET DROP FOR WRR

In Tables II, III, and IV, the drop statistics are gathered for the three queues. Of course, the queue weight does not make any sense for the simulation using RR, because every queue has the same weight.

From these results, it can be observed how the re-configuration from RR to WRR is possible to recover the bad trend for the MPEG4 stream. Our idea is to use reconfiguration and give priority to multimedia traffic only when this kind of traffic is present in the network. Doing this, it is possible to save area in FPGA and also handle more streams. This will be further described in the next section.

IV. SIMULATION RESULTS

Two main simulation sessions have been set up to study the behavior of the two schedulers and how some factors, like traffic load and queue weight, influence their performance:

- *Variable Traffic Load:* the effect of changing the traffic load in the link between the core router and the second edge has been studied. The impact of this change on the metrics of interest has been evaluated.
- *Variable Queue Weight:* the effect of changing the queue size of the MPEG4 class has been studied. The impact of this change on the metrics has been evaluated as in the previous case.

In both these sets, the simulations are carried out several times changing the random seed in the simulator each time. The average value for every metric has been calculated for every run. Subsequently, the mean over all the runs has been calculated and its 95% confidence interval determined. The WRR implementation is included in the simulator while the WFQ has been taken from [7].

A. Variable Traffic Load

A set of simulations to test the performance of WFQ and WRR has been carried out. These simulations compare the two scheduling algorithms varying the traffic load on the link of the core router. The scenario with two sending nodes is the following: a MPEG4

source and another source, actually we alternate it with a Constant Bit Rate source and a Poisson source. There are also a couple of edge routers and a core router in the middle. Each link has the same capacity (10 Mbps) and the outgoing link for the core router has been set up at 6 Mbps creating a bottleneck for the transfer. The transmission delay is 5ms. Each source starts to send at the same time and they send the traffic to a unique sink. Figure 2 illustrates the network scenario. This simulation runs in nine main sessions, each of which consists of eight simulations to calculate confidence interval; in every one of the eight main sessions the sending rate of the MPEG4 node has been changed to vary the load on the core router link. The duration is 1000 seconds for each simulation.

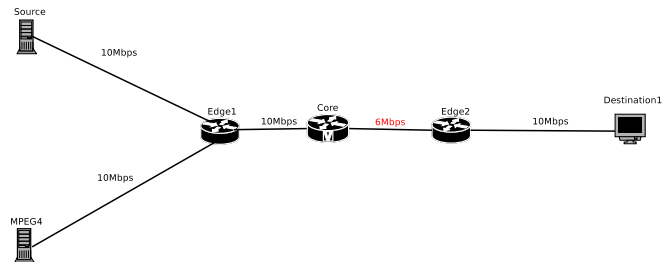


Fig. 2. Network Scenario with load variation

Figure 3 depicts the average delay experienced by the MPEG4 traffic under different load condition in the network. It should be noted that the WFQ scheduler treats MPEG4 stream better when the traffic load reaches a full load (100%). The other kind of traffic in this scenario is a constant bitrate. The advantage of using sorted priority algorithms like WFQ instead WRR is that, in the former, the delay (maximum but also the average) is proportional to the allocated rate, in the latter, the proportionality is not so clear, because in the delay computation there a fix term (the round period) dominating the term depending by the rate.

B. Variable Queue Weight

An alternative simulation scenario has been created to evaluate the impact of changing the queue size of the multimedia stream, keeping the weights for the other queues fixed. This scenario is presented in Figure 4 and the simulations have been repeated with two different kind of additional traffics, CBR and Poisson. There is a bottleneck on the link between the core router and the second edge node, due to the fact that the sending rate of all the sources is calculated to fully

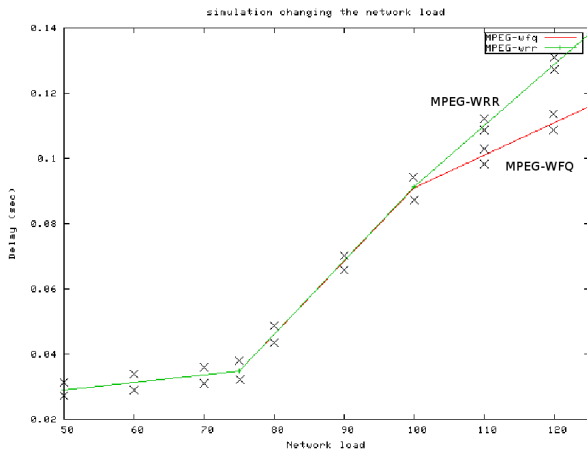


Fig. 3. MPEG4 delay variation for different traffic load with CBR

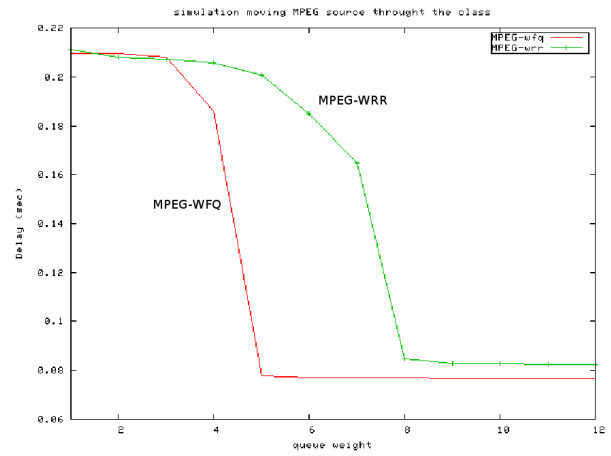


Fig. 5. MPEG4 delay variation for different class queue weight with CBR

utilize that link.

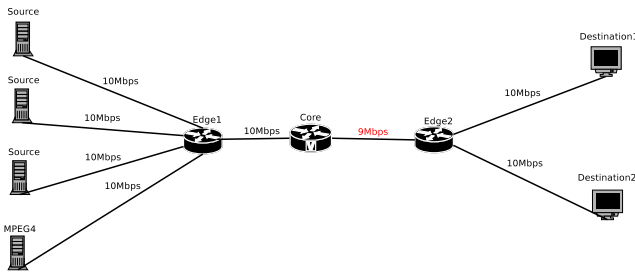


Fig. 4. Simulation Scenario with Queue Weight Variation

This set of simulations compares the impact of the choice of the scheduler on the performance of the MPEG4 stream, in fact the same analysis is carried out with WFQ and WRR. The queue weight of the multimedia traffic has been varied starting from a condition of disadvantage, with respect to the other 3 streams, arriving to a condition of very big advantage. A queue weight represents the class that a flow belongs to. For the traffic source competing against the multimedia node, the class has a fixed value, instead for the MPEG4, it has been varied starting from 1 to 12

The result of this set of simulations has been plotted in Figure 5, in this case the competing flows are CBR. This figure plots the trend of the delay for the multimedia stream each time that its class is changed. At a qualitative level, the trend between the two simulations is the same apart the region in which the weight's value is 1 and 4. This behavior happens because the drop percentage is very high when the MPEG4 class is low so the delay behavior could be very random.

This figure shows that the zone in which the performance gain is maximum is $[4 - 8]$. This investiga-

tion helps to understand some criterias to address the problem of the correct choice of the reconfiguration time. This means that the area of reconfigurability in terms of queue weights can be characterized. Having the necessity of to handle more streams, the scheduler algorithm could be switched from WFQ to WRR when the multimedia stream does not pose too strict delay requirements. For example, unless the choice of the weight (for MPEG4) is within the area $[4 - 8]$, could be more suitable to utilize WRR than WFQ because the gain in terms of delay is not so high compared to the number of streams that would be possible to handle adopting the other solution.

V. CONCLUSIONS

In this paper, we first argued that FPGAs can be a good solution to address the problem of the efficient choice of a scheduling algorithm in a QoS-aware network router, and briefly introduced the reconfigurable hardware as well as the investigated scheduling algorithms. Subsequently, we described the reconfiguration mechanism utilized to evaluate our idea, modifying the NS-2 Network Simulator. Finally, we discussed the simulation results. These results can be utilized to tune a reconfigurable network router configuration to meet the desired requirements for a certain kind of streams. The results confirm the idea on the behavior of the two investigated scheduling algorithm: WFQ outperforms WRR in terms of end-to-end delay, jitter and throughput but it is more expensive than it at a computational level. Nonetheless, it is possible to find a tradeoff between the required area in FPGA and the level of performance desired for a kind of stream. Moreover, in situations in which the level

of performance desired for a stream (i.e., MPEG4) is obtainable with a cheaper algorithm in terms of hardware implementation area, the reconfiguration is useful and permits the handling of a larger number of flows. Future investigations should study the reconfiguration mechanism implementing it in a real hardware device.

REFERENCES

- [1] Kevin Fall, "Scheduling Best-Effort and Guaranteed Connections". 1999. URL reference: <http://www.cs.berkeley.edu/~kfall/EE122/lec27/>
- [2] F. Shallwani, J. Ethridge, P. Piedad, M. Baines, "A Network Simulator Differentiated Services Implementation". Open IP. Nortel Networks, 2000.
- [3] UCB/LBNL/VINT Network simulator - NS-2 <http://www.isi.edu/nsnam/ns/>
- [4] P. Vellore, R. Venkatesan, "Performance Analysis of Scheduling Disciplines in Hardware" in the *Proceedings of the Canadian Conference, Electrical and Computer Engineering*, pp. 715-718, on May 2005.
- [5] Meina Song, Junde Song, Hongwen Li, "Implementing a High Performance Scheduling Discipline WF2Q+ in FPGA", in the *Proceedings of the Canadian Conference, Electrical and Computer Engineering, IEEE CCECE 2003*, on May 2003.
- [6] A. Matrawy, I. Lambadaris, C. Huang, "MPEG-4 Traffic Modeling Using the Transform Expand Sample Methodology" in the *Proceedings of the IEEE 4th International Workshop in Gaithersburg, Networked Appliances*, pp. 249-256, 2002.
- [7] A. Sayenko, "Adaptive Scheduling for the QoS Supported Networks", MSc. Thesis, University of Jyvaskyla, 2005.
- [8] Sungwon Yi, Xidong Deng, G. Kesidis, C. R. Das, "Providing Fairness in Diffserv Architecture" in the *Proceedings of IEEE Global Telecommunications Conference*, pp. 1435-1439, November 2002.
- [9] L. Rohan and D. Taube, "Weighted Round Robin Scheduling Module", URL reference: <http://www.arl.wustl.edu/~lockwood/class/cs535/project/fairqueue/index.html>, 2002.
- [10] Meina Song, Junde Song, Hongwen Li, "Implementing a High Performance Scheduling Discipline WF2Q+ in FPGA" in the *Proceedings of IEEE CCECE Canadian Conference*, Vol. 1, pp. 187-190, on May 2003.
- [11] Parekh, Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The SingleNode Case" in *ACM/IEEE Transactions on Networking*, pp. 344-357, on June 1993.