Revista de Sistemas de Informação da FSMA
n. 11 (2013) pp. 27-32

SISTEMAS DE INFORMAÇÃO
http://www.fsma.edu.br/si/sistemas.html

WBWTP+: A Packet Scheduling Algorithm for Achieving Proportional Delay Differentiation in IP Networks

Alencar de Melo Júnior, Juan Manuel Adán-Coello

***Abstract*—The Proportional Differentiation Model (PDM) is currently one of the main service proposals for the Next Generation Internet. This paper presents a new packet scheduling algorithm for implementing the PDM model using measurement windows and a mechanism of dynamic adjustment of packet delay estimation. Window Based Waiting-Time Priority Plus (WBWTP+), the proposed algorithm, is an evolution of the WBWTP algorithm that adjusts dynamically the relative weights of transmitted and waiting for transmission packets according to the current state of the system. The WBWTP+ delay estimator makes possible to accelerate or to delay the transmission of backlogged packets. Simulations performed to asses the performance of the WBWTP+ show that it presents significant improvement in the attendance of the PDM objective in relation to WBWTP in most scenarios, excepted when the link utilization rate is 100%. Even in that case the performance of WBWTP+ was better than that of WTP and PAD algorithms.**

***Index Terms*— QoS, Next Generation Internet (NGI), Proportional Differentiation Model, Packet Scheduling.**

I. INTRODUCTION

THE dissemination of multimedia applications has entailed computers to process continuous media (audio and video) which brought about a great increase in the volume of Internet traffic and great alterations in its nature, because continuous media presents temporal restrictions quite different from those of data oriented traffic. Multimedia applications when developed to run on the best-effort Internet have to cope with high delays and jitter (delay variation) to work properly. Therefore, the development of new Internet architectures that take into account the typical QoS parameters of these applications has been an active area of research and standardization.

The scalability of the Differentiated Services Architecture (DiffServ) [1] makes it the most promising proposal for the Next Generation Internet. In this architecture, individual reservations for microflows are not made; instead, packets of individual microflows are classified at the domain edge into flow aggregates, and a few levels of service differentiation are offered for the aggregates. Packet schedulers for this architecture are designed to meet less demanding QoS applications, i.e., applications that are able to adapt to the net conditions. In this context, the scheduler tries to offer relative QoS guarantees to flow aggregates while avoiding the starvation of any aggregate.

Among the several proposed service models in the scope of the DiffServ architecture, the Proportional Differentiation Model (PDM) [2] has received much attention lately [3][4], because of the clarity of its specification and the feasibility of its implementation through scalable mechanisms.

For any QoS oriented Internet architecture, the packet scheduler appears as a fundamental component to assure the fulfillment of the QoS requirements of the packet flows served by the routers. The main purpose of this paper is to present a new packet scheduling algorithms for the Proportional Differentiation Model: the Window Based Waiting-Time Priority Plus algorithm (WBWTP+).

The paper is organized as follows. In section II the main aspects of the Proportional Differentiation Model are discussed and a QoS metric for the model is presented. A revision of the main proposals for packet scheduling in the PDM context is presented in section III. The main contribution of this paper, the WBWTP+ algorithm, is described in section IV. The simulation model and experiments used to evaluate the algorithms are discussed in section V. Finally, the conclusions are presented in section VI.

II. PROPORTIONAL DIFFERENTIATION MODEL

The Proportional Differentiation Model (PDM) aims to provide a small number of service classes, guaranteeing only a relative ordering of classes' performances considering the QoS parameters queuing delay and packet dropping. The PDM does not require resource provisioning and route pinning is not an important aspect [2].

The main characteristics of PDM are the controllability, from the net operator point of view, and the predictability, from the user point of view. The controllability of PDM allows the operator to adjust the QoS spacing among service classes

Manuscript received March 28, 2013.

Alencar de Melo Jr. is with the Instituto Federal de São Paulo, Hortolândia, SP, Brazil (e-mail: alencar@ifsp.edu.br).

Juan Manuel Adán-Coello is with Pontifícia Universidade Católica de Campinas, Campinas, SP, Brazil (e-mail: juan@puc-campinas.edu.br).

choosing a set of differentiation parameters. Predictability in PDM is related to the maintenance of a consistent ordering among QoS classes in conformity with the parameters specified by the net operator. Ideally, controllability and predictability should be observed regardless of the load distribution over the classes, which is variable, as well as the time scale.

In PDM, the net should assure that service class C_i will receive better service, or at least not worse service than service class C_{i-1} in terms of per-hop metrics for queuing delay and packet dropping. Considering that $\delta_0 < \delta_1 < \delta_2 \ldots$, with $\delta_0 = 1$, are the QoS differentiation parameters specified by the net operator and that md_i is the average delay of queue C_i, the PDM objective for n service classes may be expressed as shown in (1).

$$\delta_0 md_0 = \delta_1 md_1 = \ldots = \delta_{n-1} md_{n-1} \tag{1}$$

Higher classes offer better performance to the users whereas the performance quantification depends on the current load in each class. It is not always feasible to achieve the above objective because, as it can be intuitively perceived, the delay of each class has a minimum value related to its load [5]. In [6], Dovrolis extends PDM considering packet drop, treating packet delay and dropping in a coupled fashion, so that higher classes account for smaller delays and drop rates than lower classes.

The eight classes selector defined by IETF in [7] is in conformity with PDM. When using PDM services, applications and users can dynamically adapt, choosing the service class that best meets their needs. Packet classification can be made either by the application, the source host or the routers located at the PDM domain edge. A restrictive policy, based or not on billing, should be implemented to prevent all users from choosing the same service class to their traffic.

A. QoSLevel: A Metric for the Proportional Differentiation Model

The PDM objective expressed in (1) can be rewritten as follows:

$$\left|\delta_i md_i - \delta_j md_j\right| = 0, \forall i, j \tag{2}$$

Hence, a metric for measuring the level of QoS attendance in PDM for n service classes can be defined as:

$$QoSLevel = \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \left|\delta_i md_i - \delta_j md_j\right| \tag{3}$$

The QoSLevel metric measures the deviation from the PDM objective for the each service classes in relation to each other. Values of QoSLevel close to zero correspond to a better attendance of the PDM objective expressed in (1) along the monitoring interval.

III. PACKET SCHEDULING IN THE PROPORTIONAL DIFFERENTIATION MODEL

The Waiting-Time Priority algorithm (WTP) [2], proposed initially by Kleinrock as Time Dependent Priority Queuing [8], was the first to be studied in the context of the PDM. In WTP, the priority of a packet increases proportionally to its queue waiting time, whereas the priorities of higher classes increase with a larger factor. The packet with the highest priority is served first, on a non-preemptive basis. The priority of a packet at the head of queue *j*, at time *t*, is given by:

$$p_j(t) = w_j(t) s_j \tag{4}$$

where $w_j(t)$ is the queuing waiting time of the packet at the head of queue *j*, and s_j is the differentiation parameter, with higher classes having higher differentiation parameters. When the link utilization rate approaches 100%, WTP can meet the PDM objective [2][5], even in short time intervals, assuming:

$$s_j = \delta_j \tag{5}$$

Therefore, every time a packet is going to be transmitted, the WTP scheduler selects a non-empty class *k*, as shown in (6), where $B(t)$ is the set of queues that have packets to be transmitted at time *t*.

$$k = \arg\max_{j \in B(t)} w_j(t) \delta_j \tag{6}$$

Simulations results presented in [2] and [5] show that the WTP algorithm can accurately meet the objective expressed in (1) only when link utilization is very high, typically above 90%. For that reason, new scheduling algorithms that can accurately meet the PDM objective even when link utilization rate is not so high are needed.

Several algorithms try to obtain better characterization for the delay of each class. WTP considers only the packets that are at the heads of queues in order to make its decision, not taking into account, among other aspects, the number of backlogged packets in each queue and their transmission times.

The Advanced WTP algorithm (AWTP) [9] extends the WTP algorithm considering for priority calculation the transmission times of the packets that are at the head of each class as well as their waiting times. However, like WTP, the AWTP algorithm does not consider all the packets that are waiting for transmission in the queues.

To select the next packet to be sent, the Proportional Average Delay algorithm (PAD) [5] computes the average queuing delay of all packets that have already been transmitted from each class. With the PAD scheduler, a class with higher importance can present larger delays than a class with smaller importance in short time intervals. This happens when many packets arrive at a queue but no packet is transmitted from it. In this case, the average queuing delay will not be updated, but the queue will have accumulated large delays, endangering the ordering of the service classes in a near future. In short time intervals, the PAD algorithm attains the PDM objective expressed in (1) only partially, because it tries to equalize the long term normalized average delay for service classes and not the normalized average delay for the last transmitted packets.

The Hybrid Proportional Delay scheduler (HPD) [5] results from the combination of the WTP and the PAD algorithms. It meets the PDM objective better than WTP under low load conditions and presents higher predictability than PAD in short timescales.

A. Packet Scheduling in PDM based on Measurement Windows

Our claim is that a more precise characterization of the queue delays experienced by service classes can be achieved using measurement windows. A packet scheduler based on measurement windows can compute the priority of a packet at the head of queue *j* at time *t* as follows:

$$p_j(t) = WW_j(t)\delta_j \tag{7}$$

where $WW_j(t)$ is the average waiting time in queue *j*, estimated from a measurement window of several packets either to be transmitted or already transmitted, and δ_j is the differentiation parameter of queue *j*.

A good estimator for $WW_j(t)$ has the following characteristics: a) a measurement window of limited size, to become sensitive to recent alterations in class load distribution; b) be updated at each packet arrival and departure, to avoid that a more important service class suffers longer delays than a class of lower importance. A delay estimator that has these essential characteristics is proposed in [10].

When implanting such delay estimator, for each service class a measurement window is defined to store information related to recent packet arrivals and departures. The window is implemented as a circular list with two pointers: *hw*, the head of the window (the position where an arriving packet is stored) and pointer *hq*, that indicates the head of the queue (the position of the next packet to be transmitted). When the queue is empty the pointers *hw* and *hq* remain aligned. For each packet to be transmitted, the window stores a time stamp corresponding to the time at which the packet entered the queue; for each packet already transmitted, it stores its waiting time in the queue.

Considering a window W_j, of size $\|W_j\|$, the average queue waiting time for a service class (or window) at time *t* is given by [10]:

Packet arrival	**Packet departure**
$n_j \leftarrow n_j + 1$ $S_j \leftarrow S_j + ctime$ $D_j^{dep} \leftarrow D_j^{dep} - d_j^{hw}$ $hw \leftarrow (hw+1) \bmod \|W_j\|$	$n_j \leftarrow n_j - 1$ $S_j \leftarrow S_j - ts_j^{hq}$ $d_j^{hq} \leftarrow ctime - ts_j^{hq}$ $D_j^{dep} \leftarrow D_j^{dep} + d_j^{hq}$ $hq \leftarrow (hq+1) \bmod \|W_j\|$

Fig. 1. Procedures performed after a packet arrival or departure.

$$WW_j(t) = \frac{1}{\|W_j\|}\left(n_j.t - S_j + D_j^{dep}\right) \tag{8}$$

where *j = (0, …, n – 1)*, is the service class number, n_j is the number of packets from W_j waiting for transmission, S_j is the sum of the timestamps of packets from W_j waiting for transmission, D_j^{dep} is the sum of the queue waiting time of packets from W_j already transmitted and *t* is the current time. As shown in Figure 1, pointers and variables are updated whenever a packet either arrives or is transmitted from each class queue.

The Window Based Waiting-Time Priority (WBWTP) scheduler, initially presented in [11], uses the delay estimator given in (8); an important question for the WBWTP algorithm is how large should be a measurement window ||*W*||. It is easy to see that, for each service class, ||*W*|| should be at least as large as its buffer size so that the delays of all packets waiting for transmission can be considered by the delay estimator. For a monitoring interval consisting of *T* packets, with *T* larger than the largest buffer size of all classes, it was verified in [11] by simulation that a measurement window of size *T* gives the lowest *QoSLevel*, allowing a better attendance of the PDM objective.

Simulation results presented in [11] also show that WBWTP presents average values for the *QoSLevel* metric lower than those presented by both WTP and PAD in most of the simulated scenarios, mainly in shorter monitoring intervals, i.e., it provides a more consistent ordering among service classes and is more precise in meeting the PDM objective. In most of the cases, the WBWTP algorithm also presented lower values for *QoSLevel* standard deviation in relation to both WTP and PAD algorithms, which contributes to the reduction of the jitter. With high link utilization rates, the performance of WBWTP is even better compared to PAD and WTP.

The desirable characteristic of preserving the premium service class from high packet dropping rates was also observed in the WBWTP algorithm. Usually, the premium service class is used by applications that require low delays and although these applications are packet-loss tolerant, protecting them from high drop rates is very important, since that sort of applications do not usually retransmit dropped packets.

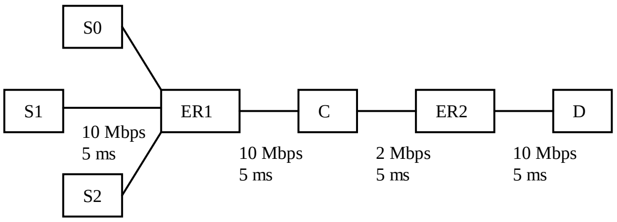

Fig. 2. Simulated DiffServ domain.

IV. THE WINDOW BASED WAITING-TIME PRIORITY PLUS ALGORITHM (WBWTP+)

Analyzing the delay estimator used by WBWTP defined in (8), it can be verified that the delays of transmitted packets (given by D_j^{dep}) and the delays of the packets waiting for transmission (given by $n_j.t - S_j$) have the same importance, that is, they contribute with equal weights for service class delay estimation.

The WBWTP+ (Window Based Waiting-Time Priority Plus) packet scheduling algorithm is an evolution of the WBWTP algorithm that adjusts dynamically the relative weights of transmitted and waiting for transmission packets according to the current state of the system. The WBWTP+ delay estimator makes possible to accelerate or to delay the transmission of backlogged packets, aiming to obtain better values for the *QoSLevel* metric. For this, the original delay estimator of the WBWTP algorithm is modified to:

$$WW_j(t) = \frac{1}{\left\|W_j\right\|}\left(\alpha_j(n_j.t - S_j) + D_j^{dep}\right) \qquad (9)$$

where α_j is the weight of packets waiting for transmission. Increasing α_j, increases the weight of the packets that are waiting for transmission at class *j* and, consequently, the weight of packets already transmitted is decreased, and vice-versa. Fundamentally, the WBWTP+ delay estimator acts accelerating or delaying the transmission of backlogged packets, looking for better values of the *QoSLevel* metric. The process of weight adjustment is detailed next.

For each service class *j*, *j = (0, ..., n - 1)*, the QoS of service class *j*, *QoSCj*, is defined by:

$$QoSC_j = \sum_{i=0}^{n-1}(\delta_j md_j - \delta_i md_i) \qquad (10)$$

QoSCj is an aggregated measure for the QoS of service class *j* in relation to the QoS of all other service classes. A positive value for a term of the above summation means that the weighted average delay of class *j* in relation to the class *i* that is being considered is above the ideal; when the value of the term is negative, the weighted average delay of class *j* is below the ideal, in relation to class *i*. The ideal, considering the *QoSLevel* metric, is that $QoSC_j$ values are equal to zero.

A positive value for $QoSC_j$ indicates that the weighted average delay of service class *j* is high in comparison with the weighted average delays of the others service classes and therefore service class *j* can receive an incentive to accelerate packet transmission. Negative values indicate that the weighted average delay of service class *j* is smaller than desired and, in this case, class *j* backlogged packets can have their transmission postponed.

Instead of using $QoSC_j$ directly it is often more convenient to use its normalized version, $NQoSC_j$, with values in the interval [-1.0, 1.0], computed as shown bellow:

$$NQoSC_j = \frac{QoSC_j}{\max\{\left|QoSC_j\right|\}} \qquad (11)$$

At each adjustment interval of length equal to *AI* packets, the α_j parameters for all service classes *j* are adjusted as follows:

$$\alpha_j = \alpha base + NQoSC_j \qquad (12)$$

V. SIMULATION

This section presents the setup and results of simulations done to evaluate the performance of the WBWTP+ algorithm. The simulations were performed using the LBNL NS simulator [12], modified to support the WTP, PAD, WBWTP and WBWTP+ scheduling algorithms.

Figure 2 shows the simulated DiffServ domain used to evaluate the per-hop performance of the proposed algorithm. In figure 2, *ER1* and *ER2* represent edge routers of a DiffServ domain, *C* is the core router, *S0*, *S1* and *S2* represent traffic sources and *D* the destination node for the generated traffic. The three links connecting *S0*, *S1* and *S2* to the edge router *ER1* have identical capacities and propagation delays, respectively 10 Mbps and 5 ms. *S0*, *S1* and *S2* generate Pareto-type traffic with the following characteristics:

- node S0: the generated traffic belongs to class C_0, burst time = 500 ms, idle time = 500 ms and α = 1.3;
- node S1: the generated traffic belongs to class C_1, burst time = 500 ms, idle time = 500 ms and α = 1.3;
- node S2: the generated traffic belongs to class C_2, burst time = 750 ms, idle time = 250 ms and α = 1.3.

The Pareto distribution is characterized by an extreme variability and it has been the most widely used to characterize Internet traffic; typical values for parameter α in the Web are in the range of 0.8 to 1.3 [13]. Traffic source activity, characterized by the burst and idle periods, were defined in such a way to emphasize the variability of the generated traffic. The initial load distribution for classes C_0, C_1 and C_2 is respectively 50%, 30% and 20% of the total traffic; at half of the simulation time, the load distribution is altered to one third of the total traffic in each of the three classes. The buffers have a maximum size of 50 packets and drop-tail policy. A packet size equal to 600 bytes was used; simulations with different packet sizes did not showed significant differences neither in this work nor in the literature [14] .

The parameters used in the simulations for the WBWTP+ algorithm are the following: $\alpha base = 1.1$ and $AI = T/5$. For this AI value, the α_j parameters are adjusted five times for each monitoring interval T. In several studied scenarios, not exhibited here, adopting $AI = T/5$ provides, most of the times, the best results for the WBWTP+ algorithm. A lower frequency for α_j adjustment do not improve significantly the performance and a higher frequency of adjustments produces high oscillation in the delay estimation process and reduces WBWTP+ performance.

A. *Predictability Analysis*

Tables I, II and III summarize the results obtained for the WTP, PAD, WBWTP and WBWTP+ algorithms in simulation experiments for link utilization rates of 75%, 85% and 100%. For each link utilization rate, three experiments were conducted with different monitoring intervals T. In each experiment, 50 packet series of T packets were produced. The average (AV) and standard deviation (SD) for the *QoSLevel* metric, computed for the 50 packet series, are presented for all the studied cases. As mentioned before, the WBWTP and WBWTP+ algorithms use a measurement window size $||W||$ equal to the monitoring interval T. On tables I, II and III, the best performance is underlined. The improvement obtained with the WBWTP+ algorithm compared to WBWTP, when positive, is shown in bold. The differentiation parameters used were $\delta_0 = 1$, $\delta_1 = 2$ and $\delta_2 = 4$.

It is possible to see in Tables I, II and III that the WBWTP+ algorithm accounts for significant improvement in the attendance of the PDM objective in relation to WBWTP in most scenarios, excepted when the link utilization rate is 100%. Even in that case the performance of WBWTP+ was better than that of WTP and PAD algorithms, considering the average and the standard deviation for QoSLevel.

TABLE I

QoSLevel for C-ER2 link utilization rate = 75%

Algorithm	*Monitoring Interval*					
	T = 10,000 packets		*T = 30,000 packets*		*T = 50,000 packets*	
	AV	*SD*	*AV*	*SD*	*AV*	*SD*
WTP	158.69	91.71	202.35	143.16	150.97	112.13
PAD	125.52	70.03	104.56	62.49	76.93	62.49
WBWTP	112.34	52.53	120.52	93.14	100.09	70.68
WBWTP+	103.91	52.35	103.65	77.70	92.99	62.61
Improv.	**7.50%**	**0.36%**	**14.00%**	**16.58%**	**7.09%**	**11,43%**

AV- average, SD- standard deviation; $\delta_0 = 1$, $\delta_1 = 2$ *and* $\delta_2 = 4$

TABLE II

QoSLevel for C-ER2 link utilization rate = 85%

Algorithm	*Monitoring Interval*					
	T = 10,000 packets		*T = 30,000 packets*		*T = 50,000 packets*	
	AV	*SD*	*AV*	*SD*	*AV*	*SD*
WTP	198.01	139.08	219.54	142.32	178.62	131.06
PAD	184.55	166.42	162.52	124.07	134.40	98.37
WBWTP	133.39	104.70	142.66	97.49	133.04	98.99
WBWTP+	120.35	75.39	130.47	97.59	126.82	93.85
Improv.	**9.78%**	**28.00%**	**8.55%**	-0.09%	**4.68%**	**5.19%**

AV- average, SD- standard deviation; $\delta_0 = 1$, $\delta_1 = 2$ *and* $\delta_2 = 4$

TABLE III

QoSLevel for C-ER2 link utilization rate = 100%

Algorithm	*Monitoring Interval*					
	T = 10,000 packets		*T = 30,000 packets*		*T = 50,000 packets*	
	AV	*SD*	*AV*	*SD*	*AV*	*SD*
WTP	234.63	139.13	280.75	162.24	240.20	123.12
PAD	278.53	368.28	224.45	139.51	231.65	253.12
WBWTP	148.36	99.0	196.22	139.17	179.60	122.12
WBWTP+	150.37	114.42	191.57	138.82	183.8	118.11
Improv.	-1.35%	-15.56%	**2.37%**	**0.25%**	-2.34%	**3.29%**

AV- average, SD- standard deviation; $\delta_0 = 1$, $\delta_1 = 2$ *and* $\delta_2 = 4$

The HPD scheduler combines the features of both PAD and WTP schedulers. As shown in [5], HPD satisfies the PDM objective better than WTP under low load and presents higher predictability than PAD in short timescales. However, HPD does surpass neither WTP nor PAD performances in other circumstances. In almost all the simulations described here, WBWTP and WBWTP+ showed a better performance than both WTP and PAD and, as a consequence, we can infer that they will also present better performance than HPD.

B. *Packet Drop Rate*

As shown in Table IV, WBWTP and WBWTP+ account for lower drop rates than PAD and slightly higher than WTP. Both WBWTP and WBWTP+ have the desirable characteristic of preserving the premium service class from high dropping rates. Results for monitoring intervals of 50,000 packets and link utilization rates of 85% and 100% are consistent to the presented in Table IV.

TABLE IV

Class Drop rate (in %) with C-ER2 link utilization rate = 75%

Algorithm	*Monitoring Interval*							
	T = 10,000 packets				*T = 30,000 packets*			
	C0	*C1*	*C2*	*Total*	*C0*	*C1*	*C2*	*Total*
WTP	8.94	0.55	0.00	4.08	10.97	1.10	0.00	5.00
PAD	6.71	4.22	1.52	4.55	8.12	4.64	2.25	5.50
WBWTP	8.67	2.08	0.20	4.44	10.52	2.59	0.52	5.42
WBWTP+	8.95	1.92	0.13	4.50	10.62	2.87	0.23	5.47

$\delta_0 = 1$, $\delta_1 = 2$ *and* $\delta_2 = 4$

C. *Effects of Different Adjusting Intervals*

As WBWTP+ did not present significant improvement in relation to WBWTP when the link utilization rate is 100%, simulations were carried out to better evaluate the influence of the size of the adjusting interval *AI* on the *QoSLevel* in this situation. Table V shows the obtained results, considering different adjusting and monitoring intervals. It can be observed that changing the *AI* interval when the link utilization rate is 100% does not bring consistent improvements to the WBWTP+ performance.

TABLE V

QOSLEVEL FOR DIFFERENT SIZES OF THE ADJUSTING INTERVAL AI, WITH C-ER2 LINK UTILIZATION RATE = 100%

Algorithm	*Monitoring Interval*					
	T = 10,000 packets		*T = 30,000 packets*		*T = 50,000 packets*	
	AV	*SD*	*AV*	*SD*	*AV*	*SD*
WBWTP	148.368	99.02	196.22	139.17	179.60	122.12
WBWTP+ (*AI = T*)	156.05	118.33	187.60	138.25	177.72	117.20
WBWTP+ (*AI = T/2*)	158.09	116.35	189.85	143.20	187.02	118.40
WBWTP+ (*AI = T/3*)	162.47	108.46	191.39	137.95	186.04	119.33
WBWTP+ (*AI = T/4*)	157.70	113.27	188.21	140.83	178.48	120.69
WBWTP+ (*AI = T/5*)	150.37	114.42	191.57	138.82	183.80	118.11
WBWTP+ (*AI = T/10*)	160.40	116.23	195.31	139.28	178.63	116.95

VI. CONCLUSION

The Proportional Differentiation Model (PDM) is one of the main service proposals for the Next Generation Internet due to the clarity of its specification and the feasibility of its implementation. In this context, as shown in our simulation experiments, the WBWTP+ and WBWTP algorithms can contribute for making the PDM viable and, consequently, for making viable to offer differentiated services to Internet users.

WBWTP+ uses a dynamic weighting method for estimating the delays of packets waiting for transmission, while keeping constant the size of the estimation window ||W||. The α_j parameters enable to increase or decrease the weight for the delays of packets awaiting transmission and consequently to decrease or increase the weights for the delays of the packets already transmitted. In an indirect way, the increase and decrease of the values of α_j decreases and increases, respectively, the estimation window size ||W||.

In a given application context, it can be determined the link utilization rate beyond which the use of WBWTP is more advantageous than WBWTP+, taking into account, among other aspects, the differentiation level and the adopted monitoring interval. Therefore, when the utilization rate reaches that limit, the value 1.0 can be adopted for all α_j values, transforming the WBWTP+ algorithm into WBWTP; when the utilization rate falls below the established limit, the α_j parameters can be calculated according to the procedure described in this paper, returning to WBWTP+. It is important to stand out that even when the performance of WBWTP+ is lower to that of WBWTP, it is still higher than the performance of both WTP and PAD.

REFERENCES

[1] Blake, S., Black, D., Carlson, M. , Davies, E., Wang, Z., Weiss, W. , An Architecture for Differentiated Services, RFC 2475, December 1998, http: http://www.ietf.org/rfc/rfc2475.txt

[2] Dovrolis, C. & Ramanathan, P. A Case for Relative Differentiated Services and the Proportional Differentiation Model, IEEE Network Magazine, vol. 13, pp. 26-34, Sept./Oct. 1999.

[3] Argibay-Losada, P. J., Suarez-Gonzalez, A., Lopez-Garcia, C. & Fernandez-Veiga, M. Flow Splitting for End-to-End Proportional QoS in OBS Networks, IEEE Transactions on Communications, vol. 58, pp. 257-269, Jan. 2010.

[4] Huang, S., Long, K., Yang, X., Chen, Q. & Li, Y. A New Proportional Differentiation Scheme Based on Batch Scheduling for Optical Burst Switching Networks, Photonic Network Communications, vol. 18, pp. 49–54, August 2009.

[5] Dovrolis, C., Stiliadis, D. & Ramanathan, P. Proportional Differentiated Services: Delay Differentiation and Packet Scheduling, IEEE/ACM Transactions on Networking, vol. 10, no. 1, pp. 12-26, February 2002.

[6] Dovrolis, C. & Ramanathan, P. Proportional Differentiated Services, Part II: Loss Rate Differentiation and Packet Dropping, IEEE/IFIP International Workshop Quality of Service (IWQoS), June 2000, pp.52-61.

[7] Nichols, K., Blake, S., Baker, F. & Black, D. L. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, IETF RFC 2474, Dec. 1998.

[8] Kleinrock, L. A Delay Dependent Queue Discipline, Journal of the ACM, vol. 14, no. 2, pp. 242-261, 1967.

[9] Lai, Y.C. & Li, W.H. A Novel Scheduler for Proportional Delay Differentiation by Considering Packet Transmission Time, IEEE Communications Letters, vol. 7, no. 4, April 2003.

[10] Salles, R.M. & Barria, J.A. Utility-Based Scheduling Disciplines for Adaptive Applications Over the Internet, IEEE Communications Letters, vol. 6, no. 5, May 2002.

[11] Melo Jr., A., Magalhães, M.F. & Adán Coello, J.M. Packet Scheduling for the Proportional Differentiation Model Based on Measurement Windows. Proceedings of The IEEE 21st International Conference on Advanced Information Networking and Applications (AINA-07), pp. 738-746, Niagara Falls, Canada, May 21-23, 2007.

[12] NS – Network Simulator Version 2, June 2009, http://nsnam.isi.edu/nsnam/index.php/Main_Page

[13] Crovella, M.E. & Bestavros, A. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes, ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, pp. 160-169, 1996.

[14] Dovrolis, K. Proportional Differentiated Services for the Internet. PHD Thesis, University of Wisconsin – Madison, 2000.