Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 49(63), Fascicola 1, 2004

End-to-end Proportional Services for TCP Flows

Csaba Simon [1], Botond Tordai [2] and Miranda Naforniţă[2]

Abstract – We proposed an end-to-end relative differentiation scheme, called network-wide proportional service adaptation for TCP traffic. Apart from the already available proposals for TCP traffic transfer that focus over the per-hop performances, this one is defined over an administrative domain. The flows are aggregated based on their ingress and egress points and the performance parameter of the classes is the goodput of these flows.

We also present an algorithm, which computes the flow-shares required to sustain the model and the considered architecture. We verify then the proposed algorithm by simulations, proposing a new shaping mechanism at the ingress nodes, the BLUE AQM, initially developed for congestion control.

Keywords: Quality of Service, Proportional Services, TCP

I. INTRODUCTION

The Quality of Service (QoS) issues in IP networks are actual and interesting research topics in network communications. These researches address the ever-growing need of applications for more network resources, delivered in a predictable manner. The main purpose of IP is to assure the same service for all applications. The technological deployments, the growing presence of multimedia applications in services and the convergence of telephony and informatics, request an IP architecture with QoS capabilities. The same-service-to-all model in the Internet makes it impossible to introduce service differentiation. Thus, in the last decade two major solutions, namely IntServ and DiffServ have been developed to empower IP networks with QoS. The Integrated Services (IntServ) [1] architecture provides service differentiation for each individual traffic flow. Because of per-flow handling, it is predictable that in the core network, where several thousands of flows have to be processed, the implementation of this model will have huge efficiency problems. The Differentiated Services (DiffServ) [2] model defines service classes for each flow aggregates. The major advantage of the DiffServ architecture is that provides good scalability, which is very important in large networks. Also this service model has drawbacks. One of the greatest drawbacks comes from the static allocation of the internal queuing parameters. Since the weights are pre-allocated, a change in the offered load among classes cannot be handled.

To eliminate this problem, the relative service differentiation [3] was introduced. In this approach, the traffic flows are grouped in a number of well-ordered service classes. In this context there is no admission control and resource reservation; it is up to the users and applications to select the class that best meets their requirements, cost and policy constraints. The relative differentiated services work on a Per Hop Basis [4], thus the achieved end-to-end service differentiation is hard to control. To maintain scalability, algorithmic complexity should be pushed to the edges of the network whenever possible.

To correct this inefficiency a network-wide service differentiation was proposed, called network-wide proportional service [5]. This model originally was proposed and developed just for UDP (User Datagram Protocol) traffic. UDP is a lightweight transport protocol. The other widely used transport protocol, the TCP (Transport Control Protocol), assures guaranteed packet delivery an complex congestion avoidance mechanisms. Due to these properties, the vast majority of communication flows in the Internet use TCP. In order to make the adaptation of proportional service model feasible, a solution for TCP support is required. Therefore we introduce a solution for a network-wide proportional service model supporting TCP data traffic.

II. PROPORTIONAL SERVICE MODEL

The proportional differentiation model 'spaces' certain class performance metrics proportionally to the differentiation parameters that network administrator determines [4]. For example, if we consider a differentiation between **m** different classes, and q_i is such a performance measure for class i, the Proportional Service (PropServ) model imposes constrains of the following from all pairs of classes:

$$q_i / q_j = c_i / c_j \quad , i, j = 1,2,\ldots,m, \qquad (1)$$

where $c_1 < c_2 < \ldots < c_m$ are the generic quality differentiation parameters (QDP). Thus even if the actual quality level of each class varies with the class

[1] Department of Telecommunication and Media Informatics, Budapest University of Technology and Economics, E-Mail: simon@tmit.bme.hu

[2] Electronics and Telecommunications Faculty, "Politehnica" University Timişoara, E-Mail: mona-naf@etc.utt.ro

loads, the quality ratio between classes remains fixed, and controllable by the network operator, independent of the class loads. Additionally, the relative ordering between classes is consistent and predictable from the users perspective.

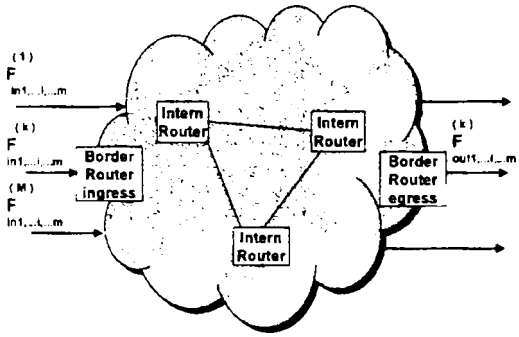

Figure 1. Network domain scenario

This new model approach, proposes a solution to obtain network level proportional differentiated service, based on the goodputs of the flows. The goodput, **G**, gives the fraction of the offered load (the data sent) that is actually transmitted through the network (achieved throughput). This model is defined for an administrative domain. Let us consider a network, which form an administrative domain with m different QoS classes. The flows with the same ingress and egress points are aggregated; according to the paths they are crossing the domain. For each ingress, **a**, and egress, **b**, there are **m** flows (flow is now a micro-flow aggregate), denoted F_i^{ab}, where the lower index, i = 1,2,...,m, identifies the QoS class the flow belongs to (see Fig. 1). For each flow, **F**, we have the offered load (input bandwidth, F_{in}) at the ingress and the achieved throughput (output bandwidth, F_{out}) at the egress. With these notations we can write the following equation:

$$G^{ab}_i = F^{ab}_{i,out} / F^{ab}_{i,in} \qquad (2)$$

Thus, based on equation **(1)** for each ingress-egress, a-b pair, the relation between the performances of different classes (on a given path) is:

$$G_i / G_j = (F_{i,out} / F_{i,in}) / (F_{j,out} / F_{j,in}) = c_i / c_j \quad i,j = 1,2,...,m, \qquad (3)$$

As one can see we omitted the upper index for each quantity in the equation for simplicity. In this approach is sufficient to define unique network-wide QoS classes, which yield equal c_j parameters for every ingress-egress pair. Using unique c_j parameters throughout the network makes the model simpler and easier to control. Costumer satisfaction may be reached by carefully choosing the pacing between the classes. Without loosing the generality of the model, in the following we consider only two classes, for the ease of presentation, the higher and a certain lower QoS class, indexed by m (the higher classes index) and i (certain classes index). To simplify the notation further, the flows between a given ingress-egress pair are indexed, thus $F^{(k)}$ denote the aggregate flow of a certain class on a single path across the domain, and we have **N** such flows (N micro-flow aggregates with the same QoS priorities). With these notations we can introduce the main (base) equation of the proportional service model:

$$G^{(k)}_m / G^{(k)}_i = (F^{(k)}_{m,out} / F^{(k)}_{m,in}) / (F^{(k)}_{i,out} / F^{(k)}_{i,in}) = c_m / c_i = \alpha_i, \quad \alpha_i > 1, \; k = 1,2,...,N, \qquad (4)$$

where, c_m and c_i are network wide differentiation parameters for the first (highest) and the second (a certain) QoS classes, and they ratio is represented by α_i.

This equation shows us that the goodput of the best quality class (which has the highest index, m) is α_i time bigger then the goodput of a certain **i** class. Is very important, that in the last formula, the two quantities, G_m and G_i, refers to one determined path, thus the paths reprezented by these two classes, **m** and [illegible] leave the network in the same egress node. Equation (4) defines the relation of the class-level flows sharing the same path in the network, but it does not adress the relation between the flows sharing the same link in the network.

A. Fairness consdierations

One drawback of today's Internet is the unfair bandwidth allocation. In the following, we present the fairness criteria proposed by this proportional service approach.

If we rewrite equation (4), we get the following relation:

$$F^{(k)}_{m,out} / F^{(k)}_{m,in} = \alpha_i (F^{(k)}_{i,out} / F^{(k)}_{i,in}), \quad \alpha_i > 1, \quad k = 1,2,...,N. \qquad (5)$$

This equation defines the relation between the highest and some certain lower classes within each flow path, and it says nothing about the case, when two or more flows share the same link. In today's networking, bandwidth is assigned to the flows (classes) in an arbitrary, inconvenient mode. According to this service differentiation approach, the competitive flows should share the common resources equally, proportional with their offered load. This desire is formulated in the next equation, which is called the *fairness condition*, and specifies the relation among concurrent flows, sharing the same link along their path, during they cross the network,

$$F^{(i)}_{out} / F^{(i)}_{in} = F^{(j)}_{out} / F^{(j)}_{in}, \qquad (6)$$

$$i,j = 1,2,...,N,$$

whenever these two paths, **i** and **j**, share the *same bottleneck link*. It has to be emphasized that this last equation, holds only for those flow pairs that experience the same loss rate at the same bottleneck link in the network.

B. UDP and Proportional Services

The proportional differentiated service model initially it was deployed to be implemented in the propagation (transmission) of UDP traffic [5]. Characteristic to the UDP data transfer is that this protocol does not verify the transmission speed with acknowledgements (does

not use feedback). This, it means, from the models point of view, that the traffic present in the edges (borders) of the network, is exactly the traffic transmitted by the application. This traffic is called the offered traffic, or the offered load, and we will denote this with **F**. Thus is ease to apply the presented model, which works with the offered loads of traffic generators.

C. TCP and Proportional Services

TCP flows use a specific end-to-end (host-to host) mechanism to control the traffic, and to avoid congestion collapse; this mechanism is called *feedback* [6, 7]. With the help of this specific mechanism, TCP flows guarantee that every sent packet will arrive to the destination. The cost of this guarantee, from the proportional services point of view is that the sender, the source does not have an offered load; the load has elastic adaptation to the state of the network. This represents a huge problem, because the presented PropServ model cannot predict the offered load of the flows, which are entering the network. As we could see in the model presented, and the selected performance parameter of the classes (goodput), the main problem is the approximation (prediction) of the offered load in the ingress nodes of the proportional service domain. Our proposal is to make this unpredictable offered load proportional with the numbers of the micro-flows within every class. In this approach the entering load of a certain class will be as follows:

$$F_{i,in} = n_i \bullet D \quad , \quad i = 1,2, \ldots ,m \qquad (7)$$

where n_i is the number of micro-flows in the certain flow-class,

D is a network wide constant,

$F_{i,in}$ is the offered load for a certain, **i**, class.

In the followings we will describe my proposed algorithm that computes the achievable bandwidths for the different flow-classes. Before the presentation of the algorithm, lets us see what represents exactly this **D** value.

D. The offered load in case of TCP flows

The proportional differentiated service handles the behavior of flows per-aggregate, not per-flow. For the ease of presentation, in the following we will call "flow" an aggregate of micro-flows, which they belong to the same priority class. As we mentioned before, in case of TCP flows, we cannot talk about a determined continue offered load. Thus, we need a theoretical approximation. Equation (7), that solves that inconvenient, shows that we equaled the bandwidth, corresponding to a flow-class, with the number of micro-flows, which are in the specified flow-class, multiplied with the biggest link capacity within the network. Thus, n_i represents the number of micro-flows that belong to the same QoS flow-class, on the same path through the network; and **D** = **C**. where **C** is the biggest link capacity in the network.

We denoted the bandwidth acquired by one flow-class (throughput), on a determined path, with $F^{x}_{i,out}$, where **x** denotes the path. and **i** denotes the flow-class. After $F^{x}_{i,out}$ is computed for all the paths, according to the classes that crosses them, we will shape (form) the flows, which are entering the network, in the ingress nodes, at the network borders, corresponding to the acquired $F^{x}_{i,out}$ values. One can see, that in the choosing of initial capacities for the entering flows is important just the fact, that even if just one flow is entering the network, this flow should occupy the highest bandwidth possible, otherwise, can appear inconveniences in the functionality of the TCP traffics data transfer, caused by the adaptability to the network state property of this kind of traffic. This condition is satisfied, because at the theoretical initialization every micro-flow gets the total bandwidth of the link with the biggest capacity. The shaping mechanism has to be implemented in the ingress nodes (border routers or boundary nodes) of every network domain, to avoid later congestion in the core network. The proposed shaping mechanism, is actually an Active Queue Management Mechanism, the BLUE AQM [8].

III. THE PROPOSED ALGORITHM

The promised algorithm will be presented in this subsection, the algorithm, which computes the distribution of the available bandwidth, the part that every single micro-flow will get within the shared (available) bandwidth. Based on the so called overload factor the allowable throughputs for the different flow-classes can be determined to assure the relative differentiation and the fairness criteria.

Assume, that in the network domain scenario presented in Fig. 1 we have **M** different links with different capacities (C_j, **j** = 1,...,**M**.), and there are **N** different routes for the flow aggregates. The tightest bottleneck along the path of a flow defines the *overload factor*, γ, for that given flow. The task in one step of my iterative method is to find the tightest bottleneck within the network and calculate the achievable bandwidths of all flows crossing the bottleneck. The bottleneck can be found by searching for the link with the highest utilization. The link utilization is computed by summing up the offered load for all flows that cross the link divided by the capacity of the link, as follows:

$$\rho_j = \Sigma_{i,k} (F^{(j)}_{i,k} / \alpha_i) / C_j \quad j = 1, \ldots, M \qquad (8)$$

where ρ_j – represents the utilization of link **j**;

$F^{(j)}_{i,k}$ – represents the offered traffic (load) of **k** flows, belonging to flow-class **i**, on link **j**;

C_j – is the capacity of link **j**, and

α_i – represents the proportional coefficient between the best flow-class, and flow-class **i**.

Note that a link is overloaded only if is ρ_j value is greater the one. The most heavily loaded link is chosen next, whose index is given by:

$\mathbf{b = arg\ max\ \rho_j}$ **(9)** The γ_j parameter for the link with the highest utilization is the following:

$$\gamma_j = 1 / \rho_j \quad (10)$$

If γ_j *is at least one*, all higher-class traffic demands will de satisfied since the links are under utilized (thus all micro-flows from flow-class **m** will get their resource requirements and they would not experience losses). The remaining task is to distribute the free remaining free bandwidth to satisfy as much lower flow-class requirements as possible.

In the other case, when γ_j *is less than one*, link **b** is a tight bottleneck. All flows that cross this link will suffer since the network cannot serve their initial offered load. In this case the achievable throughputs (the bandwidths) of flows sharing the same path, **x** (path which contains also the **b** link), are defined by the next equation:

$$F^x_{i,out} = (\gamma_j / \alpha_i) F^x_{i,in} \quad (11)$$

The excess traffic ($F^x_{i,in} - F^x_{i,out}$) would be lost at the bottleneck link anyway so it is desired to block this traffic at the ingress. By placing traffic shapers/policers at the edge (border) routers, the congestion collapse from undelivered packets can be avoided. Thus all the flows that are squeezed into the bottleneck link are considered only with this bottleneck capacity in the next iteration. A flow, which has already been shaped to the bottleneck link, is called *fixed*. As the algorithm iterates, it will 'fix' the flows one after the other. We repeat this procedure, every time when we find a bottleneck link. The algorithm ends, when every flow will be fixed, or when we would not have any more bottleneck links.

Theoretically, all the values should be recomputed, if a TCP flow enters or exits the network. Note that in normal networking condition, the number of flows presets in the network has a huge value. Thus the algorithm will not be resumed at every single TCP flow entering or exiting, because the modifications introduced by a determined number of flows can be tolerated for the good functionality of the model. In this condition, we will fix a limit for the number of micro-flows. In the moment when, after the last resume of the algorithm, the number of micro-flows entering or exiting the network reaches the determined limit, the algorithm is restarted.

We implemented in the ingress nodes the BLUE active queue management [8], to avoid network congestion, and to assure the calculated bandwidths (throughputs) for the different micro-flows.

IV. PROPORTIONAL SERVICE SIMULATION RESULTS WITH TCP TRAFFIC

In this section we present the simulation results of a ProPserv architecture with two QoS classes, applied to TCP flows. The monitored links were the ones binding the traffic sources with the network domain, and the links between the egress nodes and the destinations. More precisely we analyzed the link capacities of the links between the egress nodes and the receivers, and compared them with the entering link capacities. The basis of the proportional model is the proportional coefficient. That is why we traced its value over the links that are carrying the flow-classes after the exiting the network domain, and compared this value with the chosen value $\alpha_i = 2$. This way we could check whether the network domain is working according to a Proportional Service domain (which has to maintain this value constant) or not.

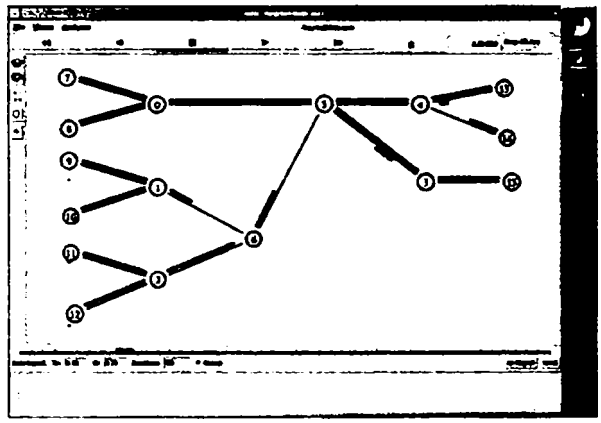

Figure 2. Simulation topology

Note that a correctly working architecture cuts the packets that would overload the core links at the ingress. That is, there should be no packet loss inside the network. This is the first criteria that should be verified. The way we implemented the simulator, the losses appear on the first, incoming links. The rest of the links should be loss free. We analyzed these links inside the network, and they had no losses.

Then we traced the achieved proportional coefficients over the three paths, as presented in Fig. 3. Let us take the links between egress 4 and Node 13. We mentioned earlier that this link is a part of the path for the flow-classes coming from traffic sources Node 7 and Node 8. These two aggregated flows will share the path between nodes 0 and 4, as specified in the Legend of the Figure. Similarly, we had two more paths. The total simulation time was 106 seconds, but only 100 seconds were in the stable region (that is, every source was sending with full rate).

As we can see in Fig. 3, the proportional coefficient is very close to the chosen value ($\alpha_i = 2$). The variance of the monitored value appears because the monitoring interval separates some packets, starting to arrive very close to the sampling period's end, and a part of it will arrive in the next second. From practical point of view we considered that the result shows that under static traffic conditions the architecture achieves the targeted behavior, thus the simulation validates the proposal.

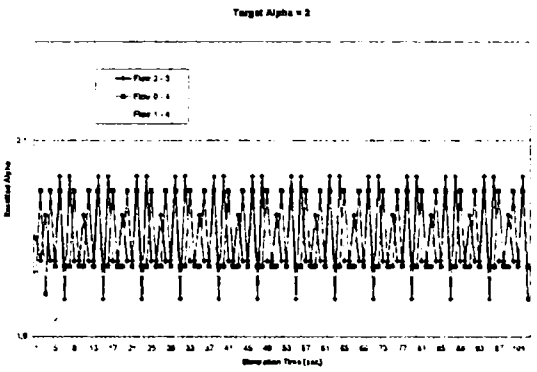

Figure 3. Simulated proportional coefficients

Even in the case of static traffic conditions the TCP flows might suffer due to asymmetric link delays. However, TCP throughput may vary based on the number of flows sharing the same link. More TCP flows might result in overload condition. downgrading the overall throughput. Therefore we tested my solution in a scenario, where every aggregated TCP flow consisted of ten times more TCP micro flows compared to the original scenario. This resulted in hundreds of TCP micro flows over the bottleneck links.

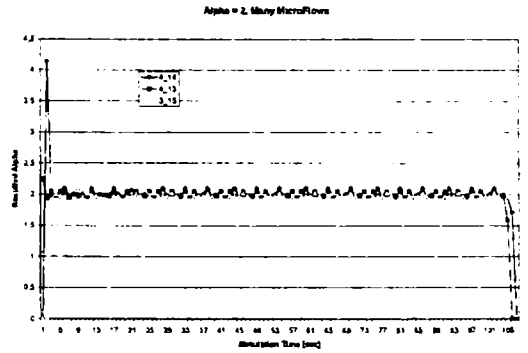

Figure 4. Proportional coefficients with large TCP traffic

As we can see in Fig. 4, even in this case the BLUE successfully shapes the aggregate flows and the resulted in the expected proportionality coefficient. This result also shows that my proposal is scalable in terms of micro-flows aggregated over the communication paths.

V. CONCLUSION

In this work an end-to-end relative differentiation scheme, called network-wide proportional service adaptation for TCP traffic has been proposed. Apart from the already available proposals for TCP traffic transfer that focus over the per-hop performances, this one is defined over an administrative domain. The flows are aggregated based on their ingress and egress points and the performance parameter of the classes is the goodput of these flows.

Then we presented an algorithm proposal, which computes the flow-shares required to sustain the model and the considered architecture. An important aspect of the model is that the resource intensive flow handling is pushed to the edges of the network domain.

We verified then the proposed algorithm by simulations, proposing a new shaping mechanism at the ingress nodes, the BLUE AQM, initially developed for congestion control. The implementation results validated my proposals for both, the algorithm and shaping mechanism. With the proposed solution, the original PropServ architecture can be extended to control the TCP traffic that represent the overwhelming majority of today's Internet traffic.

REFERENCES

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

[2] S. Blake et al., *An Architecture for Differentiated Service*, IETF RFC 2475, December 1998

[3] C. Dovrolis, P. Ramanathan, *A Case for Relativ Differentiated Services and the Proportional Differentiated Model*, IEEE Network 1999, September 1999

[4] C. Dovrolis, D. Stiliadis, P. Ramanathan, *Proportional Differentiated Services: Delay Differentiation and Packet Scheduling*, ACM SIGCOMM 1999, September 1999

[5] Cs. Simon et al., *End-to-End Relatve Differentiated service for IP Networks*, Proceedings of IEEE ISCC 2002, June 2002

[6] W. Stevens, *TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms*, RFC 2001, January 1997

[7] J. Postel, *Transmission Control Protocol*, IETF RFC 793, September 1981

[8] W. Feng, D.Kandlur, D. Saha, K. Shin, *BLUE: A New Active Queue Management Algorithms*, Project Report CSE-TR-387-99, University of Michigan, April 1999