Providing for QoS Maintenance in Wireless IP Environments

Nikos Passas*, Evangelos Zervas†, George Hortopan*, and Lazaros Merakos*

*Dept. of Informatics and Telecommunications
University of Athens
Panepistimiopolis, Ilisia
15784 Athens, Greece
Email: {passas,hortopan,merakos}@di.uoa.gr

†Dept. of Electronics
TEI-Athens
Egaleo
12210 Athens, Greece
Email: zervas@ee.teiath.gr

***Abstract*— A flow rejection algorithm for the dynamic RSVP (dRSVP) is proposed. dRSVP is an enhanced version of RSVP, aiming at providing dynamic Quality of Service (QoS) support in a variable bandwidth environment through guarantees of ranges of bandwidth instead of specific values. Flow rejection occurs when the channel quality decreases to a level that even the minimum bandwidth requirements per flow cannot be fulfilled. The proposed algorithm aims at significantly improving the flow dropping probability, without affecting the bandwidth utilization. Its operation is based on setting priority classes and rejecting the minimum required number of flows per class, in order to guarantee QoS to the remaining. Both mathematical analysis and simulation results show that the overall flow rejection probability can be significantly reduced.**

I. Introduction

Next generation networks are expected to rely on the Internet Protocol (IP) as their fundamental instrument for data transmission. Besides basic accommodation of data, guarantees for specific Quality of Service (QoS) should be provided, implied by the use of advanced network applications, such as voice and video-conference. On the other hand, next generation networks will incorporate a wide range of access systems, including both wired and wireless. Especially wireless local area networks (WLANs) are becoming more and more popular and tend to replace in many cases the traditional wired LANs [1], [2]. To interwork with wired IP networks, WLANs have to incorporate QoS mechanisms for fixed IP, such as Integrated Services (IntServ) [3], which is mainly targeted for access systems.

The Resource Reservation Protocol (RSVP) [4], [5] is the most popular signalling protocol in IntServ for requesting QoS per flow and setting up reservations end-to-end upon admission. This approach is problematic in wireless links, due to the variable available bandwidth they provide, as a result of factors such as interference, fading and node movement. To address this problem, RSVP extensions and modifications have been proposed in the literature, and one of the most promising ones is the so-called dynamic RSVP (dRSVP) [6]. dRSVP modifies the existing RSVP standard, in order to request ranges of QoS instead of specific values. In case the channel quality falls to a level that not even the lowest values can be guaranteed for the admitted flows, the network has to reject one or more of them to maintain QoS to the rest.

One of the key components of dRSVP over wireless, in terms of performance, is the flow rejection algorithm. The initial proposal [6] describes a simple algorithm that discards randomly a number of flows, to reduce the total bandwidth requirements below the offered limit. In this paper, we propose a more sophisticated flow rejection algorithm, trying to significantly improve the performance of dRSVP, especially in terms of flow dropping probability. With minor adjustments, the same algorithm can also be used in RSVP over wireless as well. To show and measure the performance improvement, we use both mathematical analysis and simulations.

The paper is organized as follows. Section II briefly discusses prior work on dRSVP, to give the background. In section III, the proposed flow rejection algorithm is described in detail. The mathematical analysis of section IV shows that the proposed algorithm outperforms the original algorithm under all traffic and channel conditions. Moreover, in section IV a simulation model and results are presented to assess the performance improvement attained by the proposed algorithm. Finally, section V presents our conclusions.

II. Description of dRSVP

RSVP was designed to enable hosts and routers to communicate with each other in order to setup the necessary states for Integrated Services support. RSVP defines a communication "session" to be a data flow with a particular destination and transport layer protocol, identified by the triplet (destination address, transport-layer protocol type, destination port number). Its operation only applies to packets of a particular session, and therefore every RSVP message must include details of the session to which it applies [4]. In the rest of the paper, the term "flow" is equivalent to "RSVP session". The RSVP protocol defines seven types of messages, of which the PATH and RESV messages carry out the basic operation of resource allocation [5].

In contrast to the stable links used in fixed networks, bandwidth of wireless links is subject to variations due to factors, such as interference, fading, and node movement, which cause changes in transmission quality. A static resource reservation based approach (such as RSVP) exhibits low performance in a variable bandwidth environment. For example, if available resources are reduced after admission control, the network may not be able to meet commitments for flows that have been successfully admitted.

To better handle this problem, dRSVP uses ranges of traffic flows specifications, instead of absolute values. The benefit is that intermediate nodes can adjust allocations, as long as these are within the specified ranges, depending on changes of the available bandwidth in the output links.

The main extensions of dRSVP, compared to the standard RSVP, are listed below:

1) An additional flow specification object (FLOWSPEC) in RESV messages and an additional traffic specification object (SENDER_TSPEC) in PATH messages have been introduced, so as to describe ranges of traffic flows.
2) An admission control process has been added, able to handle ranges of required bandwidth.
3) A bandwidth allocation algorithm has been introduced that divides the available bandwidth among admitted flows, taking into account the desired range for each flow, as well as any upstream or downstream bottlenecks.
4) Finally, a flow rejection algorithm has been added, for determining the flows that have to be rejected when the available bandwidth is insufficient to fulfill all requirements.

A complete description of dRSVP can be found in [6]. Here we focus on the last point, the flow rejection algorithm, since its operation significantly affects the overall performance of the protocol.

In case the capacity is insufficient for maintaining even the minimum of the required ranges for all flows, one or more flows must be rejected. This can occur in two cases:

i *When a new flow requests admission*. If bandwidth is insufficient, the admission control will not allow the new flow to be accepted.

ii *When the overall available bandwidth is decreased to a point that the minimum requested bandwidth per flow cannot be maintained for all flows*. In this case, one or more of the flows have to be rejected, in order to maintain the reserved bandwidth for the rest of the flows within limits.

Flow rejection can be performed through standard RSVP signalling (RESV_Tear, PATH_Tear). The algorithm described in [6] rejects flows randomly until the total requested minimum bandwidth is reduced below the available capacity. This can lead to low efficiency, in terms of flow dropping probability and bandwidth utilization, as it might tear down:

1) **more flows than necessary** (this can happen when a large number of flows with low bandwidth requirements is randomly selected for rejection, instead of a smaller number of high bandwidth flows),
2) **high priority flows** (if the algorithm does not differentiate the flows, it can reject high priority instead of low priority flows), or
3) **flows that utilize a big portion of the available bandwidth** (leading to low bandwidth utilization).

In the next section, we propose an algorithm that tries to avoid these situations, leading to lower dropping probability, especially for high priority flows. At the same time, the proposed algorithm maintains the overall bandwidth utilization at similar and slightly improved values.

III. The flow rejection algorithm

In the proposed algorithm, three criteria are considered in order to provide fair and efficient rejection:

1) **Reject the lower priority flows first**. In order to achieve this, a scheme is needed, that classifies flows into a number of priority classes. The priority classes may be defined according to a number of criteria, such as the flow type, characteristics, requirements, or willingness of the user to pay (tariffs). In its simplest form, the classification scheme can use the transport-layer protocol type included in every session identification triplet, to classify flows. For example, UDP flows can be considered as high priority, as they usually carry real time data, while TCP flows can be considered as low priority. Alternatively, an extra specification parameter can be introduced in every PATH and RESV message, referred to as CLASS_SPEC, containing the flow priority. Considering $\mathcal{M}$ priority classes, CLASS_SPEC can take values in the range $[1, \ldots, \mathcal{M}]$. This parameter could be set by the sender when a communication session is initiated.
2) **Minimize the number of rejected flows**. If there are more that one sets of flows that can be rejected, then the set with the fewer members (i.e., flows) should be selected for rejection. This criterion prevents from rejecting a large number of low bandwidth flows.
3) **Prevent underutilization**. This could happen if the algorithm chose to reject one or more flows with large bandwidth requirements. Accordingly, the algorithm should reject a set of flows that leaves a total minimum required bandwidth lower than but as close to the available as possible.

It is clear that these three criteria could conflict with each other. For example, there could be a case where the smaller set of flows that could be rejected belongs to a high priority class, or results in poor utilization of the available bandwidth. For this reason, some sort of ordering of the criteria should be defined. The ordering assumed hereafter is the order in which the three criteria were presented above. Accordingly, the algorithm will start rejecting the flows belonging to the lowest priority class, before proceeding to the higher ones; it will then try to minimize the number of rejected flows within each class, and finally to maintain as high utilization as possible.

More specifically, the algorithm is effected when the available bandwidth of the wireless link falls below the total minimum requested bandwidth of all flows. Assuming an ascending list of flows $\{f_{i1}, f_{i2}, \ldots, f_{in_i}\}$ in each priority class C_i, according to their minimum bandwidth requirements, the algorithm starts with the first flow f_{11} of the lowest priority list C_1 and checks if, by rejecting it, the total required bandwidth falls below the available. It continues traversing the list until

i) either such a flow is found, or
ii) the end of the list is reached.

In case (i), it rejects the flow and stops, since the total minimum required bandwidth is below the available. In case (ii), it rejects the last flow of the list f_{1n_1} (the one with the maximum bandwidth requirements in the list) and starts over from the beginning of the list. If all of the flows in the list are rejected (i.e., all the flows of the particular class), it continues with the flows of the next higher priority class C_2, and so on, until the total minimum requested bandwidth falls below the available, or until all flows have been rejected. At the end, the difference between the total minimum requested bandwidth of the remaining flows and the available bandwidth is proportionally shared among flows, as long as none of the flows gets more than the maximum bandwidth requested.

The same flow rejection algorithm can also be used in the case of standard RSVP over wireless, by simply assuming that the range of bandwidth requirements degenerates to a single value (i.e., minimum and maximum requested bandwidth have the same value).

IV. Analysis and Simulation Results

A. Mathematical Analysis

In this section, a simple mathematical analysis is presented, in order to compare the proposed and the original flow rejection algorithms in terms of the mean number value of flows dropped.

We assume a single priority class and a single channel transition from a "good" state, of infinite bandwidth, to a "bad" state of finite bandwidth equal to B. Given the number of flows present in the system, n, we seek the mean number $\bar{N}_{n,B}$ of flows dropped, if the channel turns in the "bad" state with available bandwidth B.

For a single priority class the original and the proposed algorithm can be described as follows:

Original dRSVP rejection algorithm: If the sum of minimum requirements of the n flows is greater than B, start dropping flows randomly, until the sum falls below B.

Proposed dRSVP rejection algorithm: Sort the flows in an ascending order according to their minimum requirements and drop the smallest flow that renders the remaining total minimum requirements less than B. If none is found, drop the largest one from the list and continue with the remaining $n-1$ flows, until the sum of minimum requirements falls below B.

Before proceeding with the analysis, we establish our notation. Let L_i denote the minimum bandwidth requirement of the i^{th} flow, $i = 1, \ldots, n$. We assume that $\{L_i\}_{i=1}^n$ is a sequence of independent and identically distributed random variables with probability density function (p.d.f.) $f(x)$. We further assume that L_i, $i = 1, \ldots, n$ is uniformly distributed in the interval $[L_l, L_h]$. The ordered minimum bandwidth requirements are denoted by $\{L'_i\}_{i=1}^n$, that is

$$L_l < L'_1 < L'_2 < \cdots < L'_n < L_h \quad (1)$$

Let S_n denote the sum of the n random variables $L_1, L_2, \ldots, L_n$, which is equal to the sum of the ordered random variables $L'_1, L'_2, \ldots, L'_n$. Under the original algorithm, each dropped flow contributes an average reduction on the consumed bandwidth equal to $(L_h + L_l)/2$. Therefore,

$$\bar{N}_{n,B} = \frac{E[S_n] - B}{(L_h + L_l)/2} \quad (2)$$

where $E[\cdot]$ is the expectation operator. Under the proposed algorithm, dropping more than one flow requires to drop the largest one first. Therefore, all the flows dropped will have minimum bandwidth demands greater than a value, L', in the range $[L_l, L_h]$. However [7], if $U'_1, \ldots, U'_m$ denote the order statistics of a set of m uniform $[L_l, L_h]$ random variables, then, given that $U'_1 = L'$, $U'_2, \ldots, U'_m$ are distributed as the order statistics of a set of $m-1$ uniform $[L', L_h]$ random variables. In this case,

$$\bar{N}_{n,B} = \frac{E[S_n] - B}{(L_h + L')/2} \quad (3)$$

which is smaller than the corresponding value in (2). L' can be estimated through the relation

$$\bar{N}_{n,B}/n = (L_h - L')/(L_h - L_l) \quad (4)$$

Figure 1 shows the mean number of dropped flows for different values of the available bandwidth in the bad state, using the mathematical analysis described above with $E[S_n]$ estimated using the Central Limit Theorem. We consider $n = 100$ flows having minimum requested bandwidth a randomly chosen number in the range $[10, 100]$. As can be observed, the performance is better for medium values of the bandwidth B in the range (1000-3000), where the improvement can be as high as 35%. For very small (close to zero) or very large values (> 5500) of the available bandwidth, both algorithms reject almost all or none of the flows respectively.

Next, we investigate the relative improvement of the proposed algorithm as a function of the interval length of the minimum requested bandwidth under the same traffic load conditions, i.e., same value of mean minimum requested bandwidth ($\mu = (L_h + L_l)/2$). To this end, in Figure 2 we plot $\eta = \frac{\Delta \bar{N}_{n,B}}{n}$, where $\Delta \bar{N}_{n,B}$ denotes the difference of the mean number of dropped flows under the original and the proposed algorithm, versus the parameter σ/μ, where σ is the standard deviation of a random variable uniformly distributed in the interval $[L_l, L_h]$. In each case, $n = 100$, $B = 2000$ and $\mu = 55$. As can be observed, the performance of the proposed algorithm increases with the interval length, since there is a variety of flows with different minimum bandwidth requirements to choose from. Therefore, the proposed algorithm is expected to perform better in environments

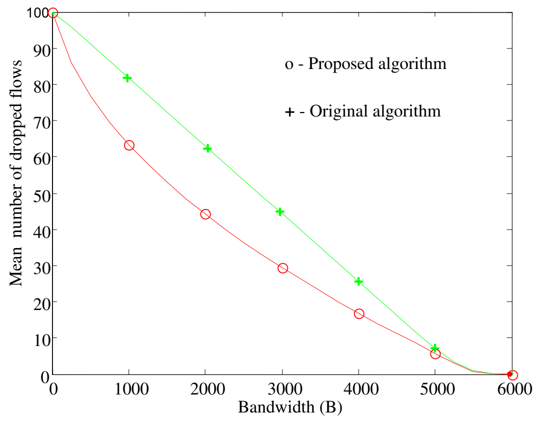

Fig. 1. Mean number of dropped flows vs. Bandwidth (Analytical and Simulation Results)

with diverse network applications, where the variability of bandwidth requirements is larger.

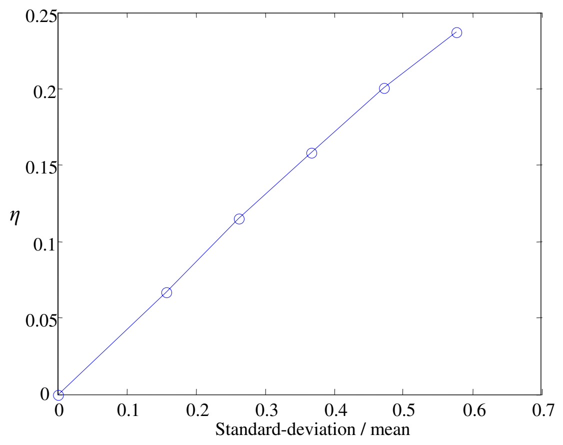

Fig. 2. η vs. σ/μ.

B. Simulation Model and Results

In this section, instead of the simple analytical model used in the previous section, a system model that captures the full functionality of the algorithm is defined and used to evaluate the performance of the proposed algorithm via simulation. This simulation model has been developed using the MATLAB programming tool [8], and is described below:

1) The model considers flows belonging to three different priority classes, C_1=Low, C_2=Medium and C_3=High, although the algorithm can work with any number. The proposed algorithm starts dropping flows belonging to priority class Low, as described in detail in the previous section.
2) Each priority class includes a mix of flow types with different minimum bandwidth requirements. We assume that the type of a flow is an independent discrete random variable (an example is given in Table I).
3) The number of active flows is not constant. In each class, flows arrive according to an independent Poisson process. Flow durations are assumed exponential independent random variables.
4) The channel can be found in two states, 'good' or 'bad', offering two levels of finite bandwidth, respectively. The time the channel stays in each state is an exponential random variable, thus the channel can be modeled as a two state Markov chain.
5) New flows are not allowed to enter the system when bandwidth is not sufficient to satisfy the minimum requirements of all (existing and new) flows.

To evaluate the performance improvement of the proposed algorithm, we considered three scenarios, where flows in each class were generated as in Table 1. As can be observed, the flow mix included a large number of "light" flows and a smaller number of "heavy" flows. To experiment with different conditions, we used different channel capacities and flow interarrival times per class. In all scenarios, we measured the dropping probability per class, defined as the average number of flows dropped divided by the total number of flows in the class.

TABLE I
DIFFERENT BANDWIDTH REQUIREMENTS PER CLASS.

Flow Types	Minimum Bandwidth Requirements	Probability
1	0.5	0.6
2	2	0.25
3	5	0.1
4	10	0.05

Scenario 1

Here, we considered a low capacity channel (Table II) and a range of comparatively large mean interarrival times, resulting in a relatively small number of active flows at any instance of time. The mean flow duration was equal to 4 for all flows, while six experiments were performed with different interarrival times per class in the range $[0.25, 0.75]$. As can be seen in Figure 3, the proposed algorithm attains lower overall dropping probability in all experiments, while for high priority classes C_2 and C_3 the performance is significantly improved. The increased dropping probability for C_1 is considered acceptable, since this is the low priority class, including mostly best-effort flows.

Scenario 2

In this scenario, the channel capacity was increased by a factor of 5 (Good=300, Bad=200), while the range of mean interarrival times was decreased by the same factor ($[0.05, 0.15]$), resulting in a considerably larger number of active flows. The mean dwell time per channel state and the mean flow duration were as in Scenario 1. Six experiments were performed with different interarrival times per class, and the results are presented in Figure 4. Again, the overall dropping probability is improved significantly, especially for large mean interarrival times. In this scenario, the dropping probability for C_3 was always equal to zero under the proposed algorithm, while, as

TABLE II

CHANNEL CAPACITY IN GOOD AND BAD STATE.

Channel State	Capacity	Mean state dwell time
Good	60	8
Bad	40	1

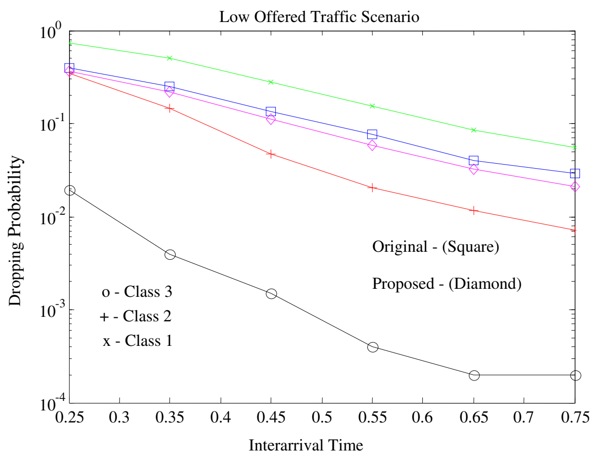

Fig. 3. Dropping probability vs. Mean interarrival time (Low offered traffic scenario).

shown in the figure, for C_2 it was much lower than the mean value of the original algorithm.

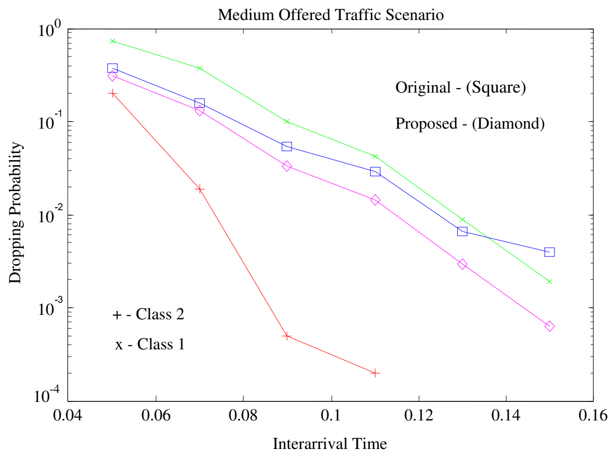

Fig. 4. Dropping probability vs. Mean interarrival time (Medium offered traffic scenario).

Scenario 3

In the last scenario, the channel capacity was again increased by a factor of 5 (Good=1500, Bad=1000), and the range of the mean interarrival times was decreased by the same factor ($[0.01, 0.026]$), to result in a much larger number of active flows. The mean dwell time per channel state and the flow duration were as in the previous scenarios. The results for different interarrival times within the specified range are consistent with the previous scenarios, as shown in Figure 5. In all figures, the gain for the high priority classes (C_2 and C_3) is increased for higher interarrival times (i.e., reduced overall traffic), since the proposed algorithm can reduce the bandwidth requirements by rejecting mostly flows from low priority class C_1.

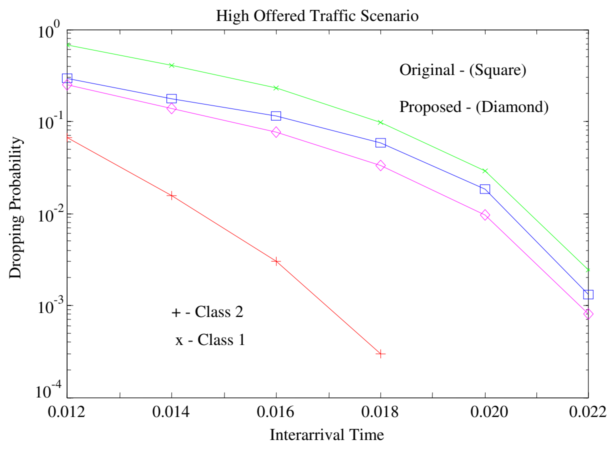

Fig. 5. Dropping probability vs. Mean interarrival time (High offered traffic scenario).

In all scenarios, the channel utilization of the proposed algorithm was equal or slightly improved compared to the original algorithm, indicating that the proposed algorithm does not effect this parameter. In Table III, we present the channel utlization results for scenario 2.

TABLE III

MEAN UTILIZATION FOR SCENARIO 2.

Mean interarrival time	Original Algorithm	Proposed Algorithm
0.05	0.9086	0.9188
0.07	0.8528	0.8601
0.09	0.7579	0.7657
0.11	0.6604	0.6581
0.13	0.5728	0.5719
0.15	0.5051	0.5043

V. CONCLUSIONS

Providing for QoS guarantees in wireless IP environments is a challenging issue, mainly due to variations of the available bandwidth offered to the users. dRSVP efficiently handles this problem by allocating ranges of bandwidth per flow, instead of absolute values.

In this paper, a flow rejection algorithm was proposed, that further improves the performance of dRSVP in terms of flow dropping probability, especially for high priority flows. The operation of the algorithm is based on setting priority classes and rejecting the minimum number of flows per class. With minor adjustments the same algorithm can also be used in the case of RSVP over wireless interfaces. A simple analytical model was used to prove the improvement attained by the proposed algorithm. Moreover, simulation results showed that the total flow dropping probability can be reduced significantly, while the improvement for high priority classes can be even more impressive. In all conditions, the algorithm attained a similar and slightly improved bandwidth utilization.

References

[1] "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High Speed Physical Layer in the GHz band," IEEE Standard P802.11, IEEE, 1999.

[2] "Broadband Radio Access Networks (BRAN), HIPERLAN Type 2, System Overview," ETSI TR 101 683, ETSI, Feb. 2000.

[3] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," RFC 1633, June 1994.

[4] P. White, "RSVP and Integrated Services in the Internet: A Tutorial," *IEEE Communication Magazine*, May 1997.

[5] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification," RFC 2205, Sept. 1997.

[6] M. Mirhakkak, N. Schult, and D. Thomson, "Dynamic Bandwidth Management and Adaptive Applications for a Variable Bandwidth Wireless Environment," *IEEE J. Selected Areas Commun.*, vol. 19, Oct. 2001.

[7] S. Ross, *Stochastic Processes*. New York: J. Wiley, 1983.

[8] MATLAB, "User's Guide: High Performance Numeric Computation and Visualization Software," The MathWorks Inc. South Natick, MA, 1994.