INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
*Int. J. Commun. Syst.* (2015)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/dac.2974

# Efficient congestion control mechanism for flow-aware networks

Jerzy Domżał*,†, Robert Wójcik, Piotr Chołda, Rafał Stankiewicz and Andrzej Jajszczyk

*Department of Telecommunications, Faculty of Computer Science, Electronics and Telecommunications, AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Kraków, Poland*

SUMMARY

Transmission based on flows becomes more and more popular in teleinformatics networks. To guarantee proper quality of service, to enable multipath transmissions, or just to increase transmission effectiveness in a network, traffic should be sent as flows. Flow-aware networking architecture is one of the possible concepts to realize flow-based transmissions. In this paper, the efficient congestion control mechanism (ECCM) is proposed to improve transmission in flow-aware networks (FAN). The mechanism makes it possible to minimize acceptance delay of streaming flows (served with high priority) without deteriorating other transmissions in the network. It is confirmed by simulation experiments that the implementation of FAN with the ECCM mechanism is a promising solution for the Future Internet. Copyright © 2015 John Wiley & Sons, Ltd.

Received 14 April 2014; Revised 13 November 2014; Accepted 12 March 2015

KEY WORDS: flow-aware networks; approximate flow-aware networks; efficient congestion control mechanism; congestion control; QoS

## 1. INTRODUCTION

In today's Internet, streaming transmissions supporting web services such as movies via YouTube or video, or voice connections through Skype, generate the majority of traffic [1]. Internet service providers (ISPs) have to ensure proper transmission parameters in their networks to serve such traffic with an acceptable level of QoS. The most important factors for transmission of streaming flows are the call acceptance delay, transmission delay, packet loss, and connection reliability. To fulfill these requirements, ISPs frequently add extra bandwidth rather than implement complex QoS architectures.

In this paper, we propose the efficient congestion control mechanism (ECCM) for flow-aware networks (FAN). A complete architecture of FAN as a proposal for the Future Internet was presented in 2004 [2]. It assumes that packets in a network represent flows, which are served according to the specified policy. The streaming flows – for example voice or video connections – are transmitted with a high priority while the elastic flows – for example data transmission – are served if there are no packets of streaming flows in the queue. The classification of flows in FAN is implicit. If a flow has more than the maximum transmission unit (MTU) bytes in the queue, it is considered as elastic one. Otherwise, it is classified as streaming one. There are two well-known versions of FAN, with the priority fair queuing (PFQ) or priority deficit round robin (PDRR) algorithm for scheduling of packets in the queues. The newest version of FAN, called approximate FAN (AFAN), was presented in [3] and assumes that scheduling of packets is based on the approximate fair dropping (AFD) algorithm [4]. As a result, the packet service is less complex than in previous solutions. AFAN is the most promising solution for FAN, therefore we focus our attention on this architecture. The ECCM

*Correspondence to: Jerzy Domżał, Department of Telecommunications, Faculty of Computer Science, Electronics and Telecommunications, AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Kraków, Poland.
†E-mail: jdomzal@kt.agh.edu.pl

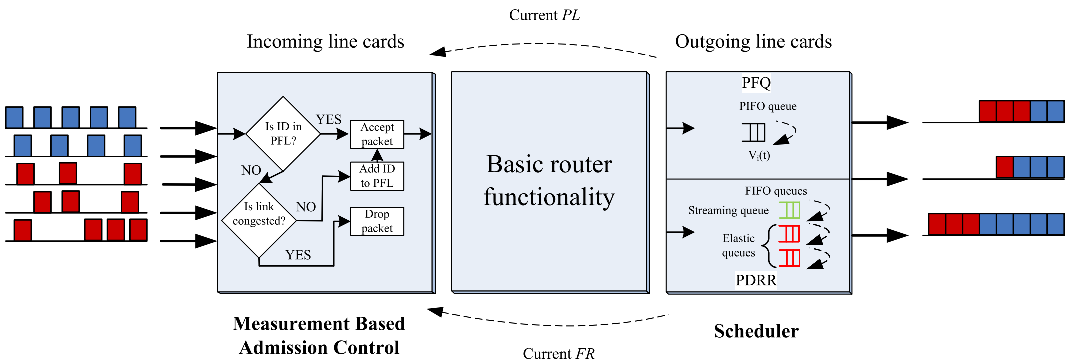


Figure 1. The cross-protect router.

ensures fast acceptance of streaming flows in FAN routers, even in congestion, and may be used in each FAN implementation. Moreover, it does not significantly affect the other transmissions in a network and is fully automatic.

The cross-protect router (also known as the XP router) is the main element of FAN. Its functionality is presented in Figure 1. The measurement-based admission control (MBAC) accepts or rejects the packets of flows. When congestion is not observed, all packets are accepted, and the identifiers of flows represented by these packets are added to the protected flow list (PFL). In congestion, only packets whose flows are recorded in the PFL are accepted. The scheduler block decides on fair queuing of accepted packets. Moreover, the values of two congestion indicators are periodically estimated in this block. The fair rate (*FR*) is the rate which is or might be realized (in congestionless state) by a flow, and the priority load (*PL*) represents the level of priority traffic in the link. If the border values of *FR* or *PL* (*min_FR* or *max_PL*) in the outgoing link are exceeded, the congestion is noticed.

Flow-aware networks have many advantages. Among them, the most important is scalability. It was proved that the number of active flows (which have packets in the queue) to be scheduled does not increase with link capacity and amount of traffic to send. The authors of [5] shown that thanks to the admission control block, this number may be limited to the reasonable number. The second important advantage of FAN is fairness among flows, which is ensured if only links are not saturated (this is ensured by the MBAC). What is also very important, FAN conform to the net neutrality paradigms [6]. In each FAN version, the differentiation of packets is implicit and made by each router in the network independently. In this way, it is possible to differentiate services and maintain fairness and neutrality. Flow-based approaches, like OpenFlow [7] become more and more popular. Flow-aware networking is still an interesting research issue for scientists and researchers all over the world. New solutions for FAN have been recently proposed in [8–12]. The authors of these papers propose new mechanisms for congestion control, improving fairness, routing, and energy efficient transmission in FAN. The analysis of the first tests of the XP router prototype was presented in [13]. The authors of [14] propose a model for mapping multiple classes of streaming and elastic flows into two-dimensional scheme. They argue that the assumption of two traffic classes allows for performance analysis in large models and such method is less complex than alternative solutions. A new QoS model for traffic based on flows in wireless networks was proposed in [15]. The authors present the algorithm for QoS-aware fair scheduling of packets, which ensures high priority for selected flows and fairness among different flows being served by the same node. Some new traffic management solutions to be used at different layers for networks based on flows have been published in recent years, too. In [16], the authors propose new algorithms for fair management of flows in networks where route discovery mechanism is implemented. In the proposed solution, selected aggregates of flows (flow classes) are transmitted with high QoS in a network with multiple simultaneously activated routes at the network layer. At the data link layer, traffic may be managed to minimize data packets losses, for example, in full-duplex Ethernet networks by using the PAUSE flow control mechanism (the mechanism that generates PAUSE



DOI: 10.1002/dac

frames at data link layer) [17]. The generation of PAUSE frames at the media access layer (MAC) control sublayer in congestion that causes transmission of selected devices is stopped and continued when the resources become available. In [18], a new mechanism for transmission control protocol (TCP)-trunking flow control is proposed. It controls the transmitted segment size and makes it possible to transmit many TCP flows in one trunk link based on the current conditions in the network. As a result, TCP connection throughput is optimized.

The remainder of the paper is organized as follows. The AFAN architecture is presented in Section 2. In Section 3, the ECCM is introduced. Section 4 presents the results of simulation analysis and Section 5 concludes the paper.

## 2. APPROXIMATE FLOW-AWARE NETWORKS

The measurement-based admission control block operates exactly the same way in each FAN version. The only difference between them is the scheduling algorithm. Moreover, the values of *FR* and *PL* are calculated in AFAN in a different, more effective way than done in its predecessors.

The values of *PL* in AFAN are estimated in a different way than in the other two versions of FAN. We have to add-in a counter a number of bytes of priority packets departing the router. An estimate of the *PL* is as follows:

$$PL = \frac{(pb(t_2) - pb(t_1)) \times 8}{C\ (t_2 - t_1)} \tag{1}$$

where $pb(t)$ is the value of the mentioned counter at time $t$, $(t_1, t_2)$ is the measurement interval (in seconds), and $C$ is the link bit rate.

In FAN implementation with PFQ or PDRR, the counter of priority packets is incremented when streaming packets arrive at the router. While they may not be served, such a measure is less precise than the method proposed in this paper.

The values of *FR* are computed as follows:

$$FR = \frac{\max\{S \times C, FB \times 8\}}{t_2 - t_1} \tag{2}$$

where *FB* means a number of bytes of packets of elastic flows sent during the time interval $(t_1, t_2)$ in reference to the number of active elastic flows, *S* is the sum of inactivity intervals during the $(t_1, t_2)$ period, *C* is the observed link capacity.

In FAN implementation with PFQ or PDRR, to compute values of *FR*, the fictitious flow is generated and its rate is observed. Such a method is more complex than the one proposed in this paper.

In the following section, the operations on packets in AFAN are presented in details.

### 2.1. Operations on queues in approximate flow-aware networks

In AFAN, packets may be queued in one of two FIFO queues (one for streaming and one for elastic packets). If the incoming packet represents a flow whose number of queued bytes is lower than or equal to MTU, this packet is sent to the queue for priority packets (the same process is used in any FAN architecture). On the other hand, the value of the approximate buffer size (*ABS* parameter) needs to be estimated from the following formula:

$$\begin{cases} ABS = (1 - w_q)ABS + w_q q & \text{if the queue is nonempty} \\ ABS = (1 - w_q)^m ABS & \text{if the queue is empty} \end{cases} \tag{3}$$

The $w_q$ parameter in this formula is a queue weight, $q$ means the buffer size, and $m$ is the number of packets possible to be sent when the line is free. $m$ is calculated as follows:

$$m = (time - time_q)/s \tag{4}$$



DOI: 10.1002/dac

In this formula, $time$ is the current time, $time_q$ means beginning of the idle time, and $s$ represents the time when a packet is being served.

Two thresholds are set in the buffer. If the value of $ABS$ is greater than or equal to max_$th$, the arriving packet is rejected. If the $ABS$ is greater than or equal to min_$th$ and lower than max_$th$, we have to draw a packet from the elastic queue. Then, the identifiers of both flows are compared. If both packets belong the same flow, the randomly selected one is dropped and the packet $p$ is dropped with the probability $P_{\text{AFAN}}$. This probability is calculated in exactly the same way as in the AFD algorithm. Packet $p$ is queued and served in each other case.

The authors of [19] propose two methods for estimating the $P_{\text{AFAN}}$ values. They suggest that the second method, called 'uniform random variables' better fits the needs of the queuing algorithm. $P_{\text{AFAN}}$ is calculated as follows:

$$P_{\text{AFAN}} = P_{\text{AFAN\_temp}} / \left(1 - count \cdot P_{\text{AFAN}}\right) \tag{5}$$

The values of $P_{\text{AFAN\_temp}}$ are estimated from:

$$P_{\text{AFAN\_temp}} = max\_p\,(ABS - min\_th)\,/\,(max\_th - min\_th) \tag{6}$$

In this formula, $max\_p$ is the maximum acceptable value of $P_{\text{AFAN\_temp}}$. We have to be aware that the value of the $max\_p$ parameter has to be set properly to estimate $P_{\text{AFAN}}$ efficiently.

The expected value for this method is calculated as follows:

$$E[X] = 1/(2 * P_{\text{AFAN}}) + 1/2 \tag{7}$$

Analyzing this parameter, we may say that if we set $max\_p$ to 0.02 (as we did in our tests) and the $ABS$ is equal to $(min\_th + max\_th)\,/2$, two hundred of arriving packets will be dropped. The value of the $max\_p$ parameter should be set in such a way that the $P_{\text{AFAN}}$ probability will change slowly. As a result, the fluctuations in the $ABS$ will be minimized.

It is very simple to select a packet to be sent. First, packets from the priority queue are served. Next, if there is no packet in the priority queue, elastic packets are selected to be served. In PFQ and in PDRR, this process is more complex. In PFQ, we have to maintain the virtual pointer between streaming and elastic packets. In PDRR, each elastic flow has its own queue and they are served with the round robin regime.

### 2.2. *Complexity of approximate flow-aware networks*

Both the queuing and dequeuing operations are less complex in AFAN than in the other FAN versions, which is a strong argument for promoting this solution. The buffer occupation is estimated in each FAN architecture when a new packet representing an elastic flow arrives to the router, and based on the received value similar operations are performed. However, the queuing process of streaming flows in AFAN or in PDRR is less complex than in PFQ (where packets of streaming flows are queued in a similar way to packets of elastic flows in one push-in first-out (PIFO) queue).

The AFAN algorithm is significantly less complex than the other FAN versions when comparing dequeuing operations. It is only necessary to check if the priority queue is empty, and if not, to serve packets from it first. In the other case, packets from the elastic queue are served. In PFQ or PDRR, this process is much more complex. The additional structures, such as the active flow list (AFL), necessary in both well-known versions of FAN, are not implemented. As one can see, AFAN is a promising architecture. It is worth noting that it is less complex than its predecessors and easy to be implemented in XP routers.

## 3. THE EFFICIENT CONGESTION CONTROL MECHANISM

Six mechanisms to control congestions in FAN have been proposed so far. In the enhanced flushing mechanism (EFM), remove active elastic flows (RAEF), remove and block active elastic flows



DOI: 10.1002/dac

(RBAEF), remove and prioritize in access active elastic flows (RPAEF), enhanced flushing mechanism with priority (EFMP), and remove and accept most active flows (RAMAF) mechanism, congestion is eliminated by removing all or a part of identifiers registered in the protected flow list from time to time [12, 20–22]. This way, new flows may be accepted. The goal of the mechanisms is to reduce the acceptance delay of streaming flows in XP routers. According to [23], international streams (e.g. intercontinental voice calls) should begin transmission within 11 s, while the acceptance delay for local streams should not exceed 6 s. Without such mechanisms, streaming flows may have to wait for transmission for a long time in some cases.

In the EFM, the identifiers of the elastic flows are deleted from the PFL in congestion periodically. As a result, the congestion is not observed for a moment and new flows may be accepted in the router. However, when a link becomes congestion-less, all identifiers of flows waiting for transmission (including those elastic removed before) are added to the PFL. While this operation is invoked periodically, the number of identifiers in the PFL increases and as a result the values of fair rate decrease. It means that the rate achieved by elastic flows is significantly lower than the assumed $min\_FR$.

In the RAEF mechanism, the identifiers of most active elastic flows (being active for at least a specified time period) are removed from the PFL periodically. In comparison with the EFM, the number of removed identifiers is lower in RAEF, however, still new elastic flows may be accepted when a link becomes congestion-less.

The RBAEF mechanism is an extended version of the RAEF. In this solution, the identifiers are deleted from the PFL based on the same assumption as in the RAEF. However, they then are added to the blocked flow list for a short, fixed-time period. Then, in a congestion-less state, they are not accepted immediately. This gives the preference to new flows and limits the total number of flows in the PFL after removal of identifiers. The disadvantage of this solution is, however, that the transmission of removed flows may be stopped for a long time, which may not be acceptable.

In RPAEF, identifiers of elastic flows are deleted from the PFL periodically, as in the case of RAEF or RBAEF. However, in this case, the identifiers of removed flows are then registered in the priority access flow list (PAFL) for a short time period. If a packet arriving at the admission control block when the outgoing link is not congested represents the flow with identifier in the PAFL, the packet is accepted. Otherwise, the packets of flows without identifiers in the PAFL are accepted with low probability $P_{\mathrm{RPAEF}}$. This probability is set to 1 if there is no identifier in the PAFL. The goal of such a solution is to accept new streaming flows quickly, and to ensure a short inactivity time of elastic flows whose identifiers are removed from the PFL. Moreover, the proposed mechanism allows for limiting the number of all flows accepted after a cleaning action of the PFL content. However, this number sometimes tends to be too high, especially in a highly loaded links.

The EFMP mechanism, along with the RAMAF mechanism presented in the succeeding section, is one of two newest congestion control solutions for FAN. It assumes that periodically (once a *pfl_flushing_timer* in congestion) the identifier of one selected elastic flow is deleted from the PFL in congestion and written to the PAFL. This operation is repeated as many times as the outgoing link becomes congestion-less. The flow is selected according to one of the policies:

- Oldest-flow policy,
- Most-active-flow policy.

In the first case, the oldest flow (the flow among active flows which begun transmission as first) is selected. The operation on a packet in EFMP with this policy is shown in Figure 2. In the second case, the flow with the most bytes in the queue is removed from the PFL. The removed identifiers are written to the PAFL and added again to the PFL when the link becomes congestion-less. In this solution, the transmission of removed flows is broken for a moment, which results in lower transmission rate. Moreover, after removing the identifiers from the PFL, new elastic flows may be added, which deteriorates transmission of previously accepted flows.

In the RAMAF mechanism, $N$ identifiers of most active flows are periodically (once a *cleaning_timer* in congestion) deleted from the PFL and registered in the PAFL. This number



DOI: 10.1002/dac

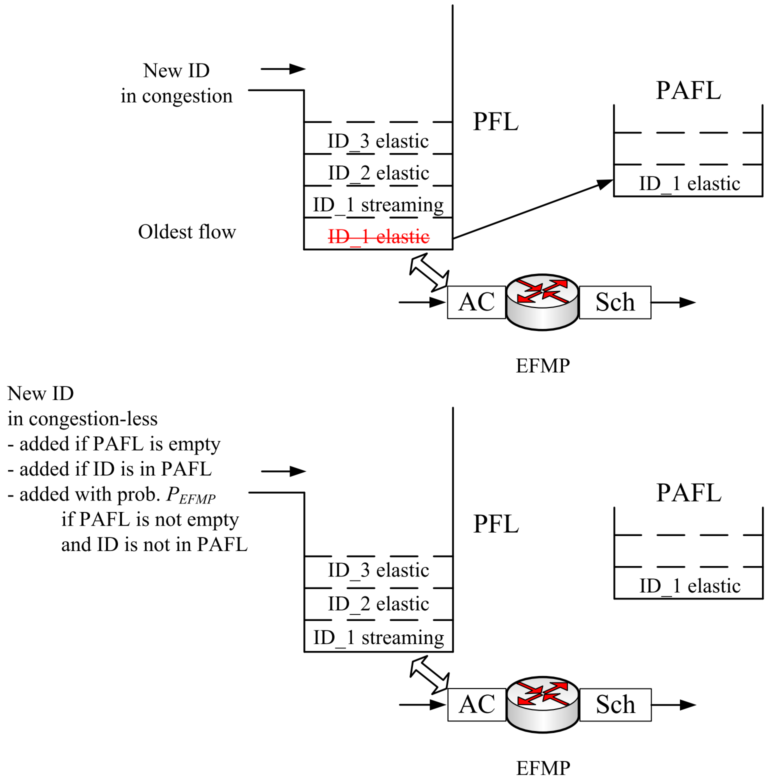


Figure 2. The enhance flushing mechanism with priority with the oldest-flow policy. PFL, protected flow list; PAFL, priority access flow list.

is calculated based on the queue occupancy and may change dynamically. The value of $N$ is calculated as follows:

$$N = PFL\_size - (100/minFR - 1) \quad (8)$$

where $PFL\_size$ means the number of identifiers in the PFL. Next, in the congestion-less state, the identifiers removed before are added to the PFL again. As a result, the oscillations of the $FR$ values around the $min\_FR$ are minimized. Moreover, new streaming flows may begin to be transmitted (identifiers of new elastic flows are deleted from the PFL when RAMAF finishes its operation). The operations in RAMAF are presented in Figure 3. The main advantage of the mechanism is that it works dynamically. Only the value of the maximum acceptance delay of streaming flows has to be assumed and the rest is performed automatically. However, the mechanism is very complex.

In this paper, we propose and analyze the ECCM. Unlike the congestion control mechanisms mentioned previously, ECCM does not remove the identifiers of already accepted flows in order to eliminate congestion. Instead, it controls the value of $FR$. As a result, the ECCM ensures more effective transmission of flows in comparison to other solutions, and is less complex than the most advanced and promising approaches.

The ECCM mechanism assumes additional operations in the MBAC in comparison to basic FAN. The pseudo-code for realization of the ECCM is presented in Figure 4, and the operations on packets in ECCM are shown in Figure 5.

The procedure of the ECCM may be triggered when a packet of a new flow arrives at the router in congestion. If time from notification of congestion in a link ($current\_time - congestion\_time$) exceeds an acceptable fixed value ($max\_accept\_delay$), the value of fair rate is set to $min\_FR$ (lines 3–7 in Figure 4). The value of the $max\_accept\_delay$ parameter should be set statically by



DOI: 10.1002/dac

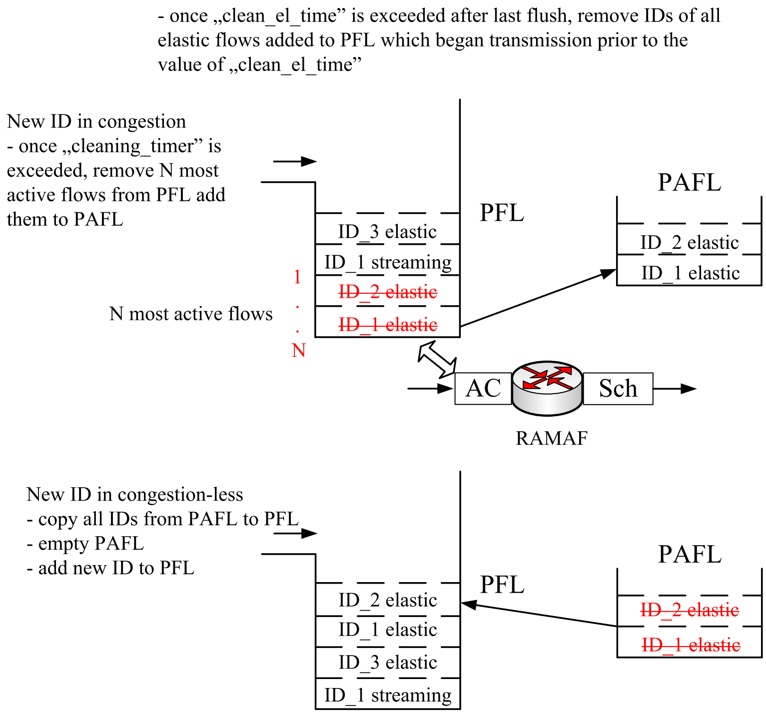


Figure 3. The remove and accept most active flows mechanism. PFL, protected flow list; PAFL, priority access flow list.

```
######### on a new flow packet p arrival in congestion #########
current_time = Scheduler :: instance().clock()
if current_time − congestion_time > max_accept_delay && control_param = 0 then
begin
    stop procedure of estimation of FR
    control_param = 1
    FR = min_FR
    ECCM_time = current_time
end
if control_param = 1 && current_time > ECCM_time + 0.5 × FR_int then
begin
    congestion_time = current_time
    start procedure of estimation of FR
    control_param = 2
end
if control_param = 2 && current_time > ECCM_time + FR_int then
begin
    for (i = 1; i <= pfl_size; i++) do
    begin
        active_time(i) = current_time − first_time(i)
        if flow_bytes(i) ≥ MTU then
        begin
            if active_time(i) > 0.5 × FR_int && active_time(i) < FR_int then
                remove ID(i) from PFL
        end
    end
    control_param = 0
end
########################################################
######### in procedure of estimation of FR #########
if FR > min_FR then
    congestion_time = Scheduler :: instance().clock()
```

Figure 4. Pseudo-code of the efficient congestion control mechanism. PFL, protected flow list; MTU, maximum transmission unit; *FR*, fair rate.



DOI: 10.1002/dac

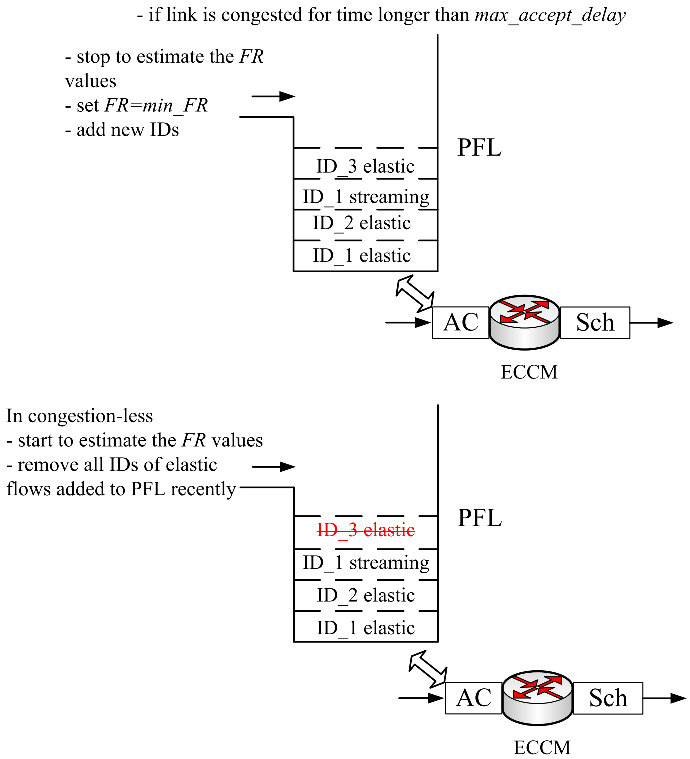


Figure 5. The efficient congestion control mechanism. PFL, protected flow list; *FR*, fair rate.

the operator according to her needs. In the next section, we present the simulation results for different values of this parameter. Change of the *FR* value to $min\_FR$ allows to eliminate congestion, which is observed when fair rate is below $min\_FR$, and to accept new flows. We also need to stop the process of estimation of the *FR* values (line 5) and set the *control_param* to 1 (line 6) and $ECCM\_time$ to *current_time* (line 8) to indicate the beginning of the ECCM procedure. After time equal to $0.5 \times FR\_interval$ (half of time interval between two estimations of the fair rate values; line 10) we set the *congestion_time* to *current_time* (line 12) to indicate the last time of congestion and start the procedure of estimation of the *FR* again (line 13) and the *control_param* to 2 (line 14). $0.5 \times FR\_interval$ is sufficient to accept all flows waiting for acceptance. Next, we need to remove the identifiers of elastic flows added during the ECCM procedure (lines 16–26) and set the *control_param* to 0 (line 27). To find all elastic flows accepted during the ECCM procedure, we have to look for flows which begun transmission in time period when fair rate was equal to $min\_FR$ and which have more than MTU bytes in the queue (this allows for recognition of elastic flows). Each time the *FR* value is greater than $min\_FR$, the value of *congestion_time* must be set to the time when the last *FR* value was estimated (lines 31–32). In this way, we know when the link becomes congested.

The ECCM is less complex than EFMP or RAMAF and is definitely much easier to be implemented in XP routers. We do not need two lists (PAFL is not necessary). As a result, less memory (with fast access) is needed. Moreover, and what is most important, we do not need to remove any identifier from the PFL when congestion is observed. To find the oldest or most active flows in EFMP or in RAMAF, we have to check the whole PFL content, which may result in lack of scalability of these solutions. In ECCM, it is not necessary. We must only change the value of the *FR* parameter for a moment and to remove identifiers of recently added elastic flows, which does not need to be done immediately. Such identifiers may be found in an easy way – their identifiers are written at the top of the PFL. This short analysis shows the strength of advantages of the ECCM, especially when considering its implementation. Concluding, less resources are needed to be used and fewer operations need to be performed in ECCM than in its predecessors.



DOI: 10.1002/dac

## 4. SIMULATION ANALYSIS OF AFAN WITH EFMP, RAMAF, OR ECCM

The simulation experiments, described in this section, were performed in the ns-2 simulator [24]. The goal is to show how the ECCM, EFMP (we chose the oldest-flow policy), and RAMAF improve the performance of transmission of streaming flows in AFAN architecture. We decided to present the results only for AFAN, because it is the most promising FAN architecture. Moreover, the results are similar for each FAN architecture.

The simulated topology is presented in Figure 6. It is simple, yet adequate for analyzing the new mechanism in FAN. All routers in FAN operate independently. As a result, the decisions in routers are taken locally, without any information from other devices. No signalling among XP routers is needed in a network. It means that if a solution works in one link, it will also work in any other FAN link. Moreover, the simulation results prove that the proposed solution is scalable and may work in links with any capacity.

At the outset, the capacity of the FAN link was set to 100 Mbit/s. Of course, this value is too low when considering core links. However, the obtained results are scalable. The only reason for making simulations for a low capacity core link is the time needed for executing experiments. The capacity of other (access) links was set to 10 Mbit/s. It is a reasonable assumption that access links have lower capacity than the core link; however, they are able to saturate the core link. The simulations were repeated at least 10 times for each experiment. 95% confidence intervals were calculated using the Student's *t*-distribution.

The volume of traffic to be sent by each of 400 elastic flows from nodes $S_{Ei}$ to $D_{Ei}$ was generated accordingly to the Pareto distribution (the mean size of file to be sent was set to 210 Mbit and the shape parameter was set to 1.5). Exponential distribution was used to generate the inter-arrival times of TCP-based elastic flows (the mean value was set to 0.2 s) and 20 user datagram protocol (UDP)-based streaming flows (the mean value was set to 1 s) being sent from $S_{Si}$ to $D_{Si}$. For elastic flows, we decided to set the packet size to 1000 bytes, which is a typical value for IP networks. The transmission rate of UDP flows was set to 80 kbit/s and the packet size was set to 100 bytes (as is the case in a typical Skype VoIP connection). We assumed that the duration of simulation runs was 300 s. It allowed to observe the values of the analyzed parameters. The values of the *PL* parameter were estimated every 50 ms while the measurement interval for the fair was set to 0.5 s. These values guaranteed stable transmission and were chosen experimentally. The value of *max_PL* was set to 70% of link capacity, the value of *min_FR* was set to 5% of link capacity, and the warm-up period was set to 20 s, which means that we did not observe results obtained before 20 s. The crucial parameters for our analysis are $FR$ and $min\_FR$. The assumed values mean that each elastic flow should have guaranteed bandwidth on the level of 5 Mbit/s and the state of the FAN link (congested or not) is observed two times a second. These values are reasonable and we

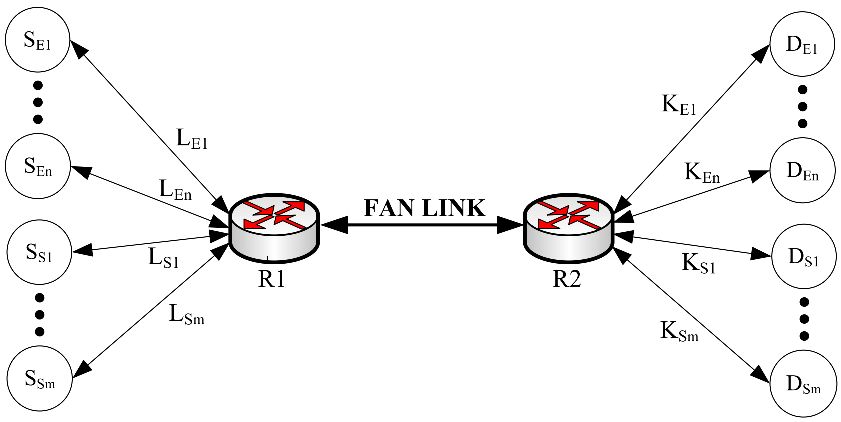


Figure 6. Simulation topology.



DOI: 10.1002/dac

did not change them in our simulations. Moreover, the ECCM algorithm takes into consideration the changes of values of these parameters.

For AFAN, the queue was set to 10 000 packets, $min\_th$ (the minimum threshold) to 4000 packets, $max\_th$ (the maximum threshold) to 9000 packets, and the $w_q$ (the queue weight) parameter was set to 0.002. These values were set experimentally, based on the analysis presented in [3].

The simulation results are presented in Figures 7–9. The *CM_interval* stands for the time between two runs of a congestion control mechanism (*pfl_flushing_timer* for EFMP, *cleaning_timer* for RAMAF, and *max_accept_delay* for ECCM). First, we observed the values of acceptance delay for streaming flows. We can see that the results obtained for the RAMAF and the ECCM are significantly better than those observed for the EFMP. The EFMP has to be run more frequently, because for each run only one flow is deleted from the PFL. We present the results for this mechanism only for values of the *pfl_flushing_timer* ranging from 1 to 4. For higher values of this parameters, the results of acceptance delay are completely unacceptable. For lower values of the *CM_interval* parameter, the results are similar for RAMAF and ECCM. When the values of the *CM_interval* parameter increase, the ECCM appears to be better. The acceptance delay of streaming flows increases with increasing values of the *CM_interval* parameter. As one can see, streaming flows are accepted faster if the values of the *CM_interval* parameter are lower. Moreover, based

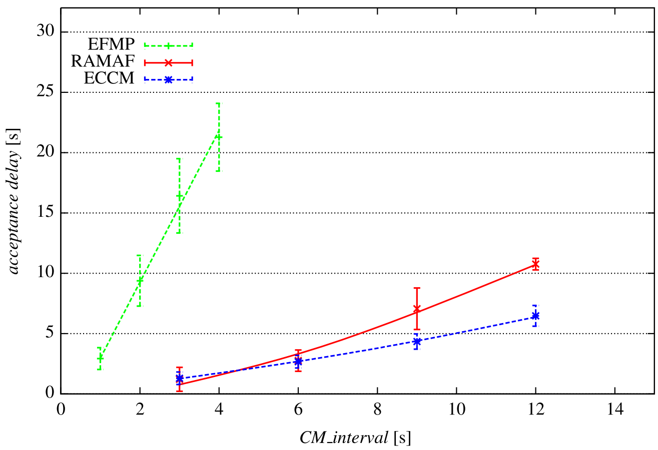


Figure 7. Acceptance delay of streaming flows. EFMP, enhanced flushing mechanism with priority; RAMAF, remove and accept most active flows; ECCM, efficient congestion control mechanism.

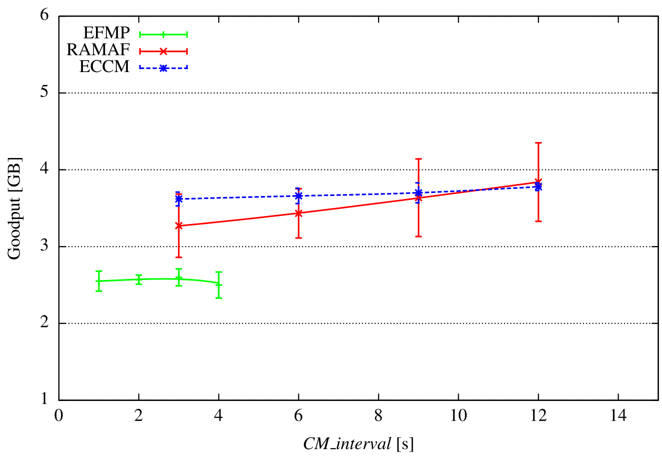


Figure 8. Goodput of elastic flows. EFMP, enhanced flushing mechanism with priority; RAMAF, remove and accept most active flows; ECCM, efficient congestion control mechanism.



DOI: 10.1002/dac

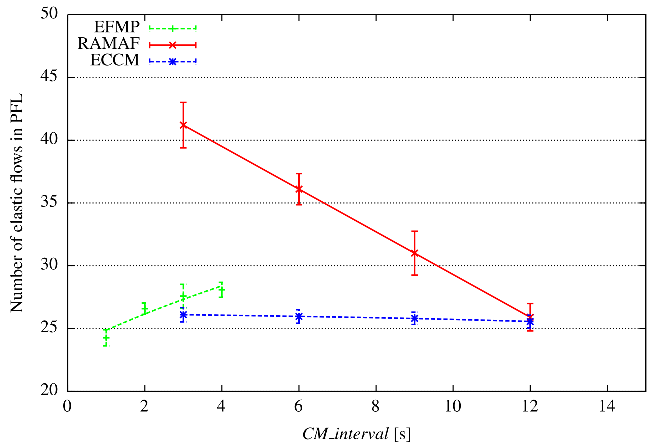


Figure 9. Number of identifiers of elastic flows in protectd flow list after flushing. EFMP, enhanced flushing mechanism with priority; RAMAF, remove and accept most active flows; ECCM, efficient congestion control mechanism.

on [23], the acceptable values of the acceptance delay for local connections should be lower than 6 s. Such values are observed when the *CM_interval* parameter is equal to 3 or 6 (for ECCM, the results for $max\_accept\_delay = 9$ are also acceptable). For higher values of *CM_interval,* the results are unacceptable. It should be noted that under the same conditions, in AFAN, without any congestion control mechanism, streaming flows were accepted after $201.66 \pm 33.90$ s. This analysis shows that RAMAF and ECCM ensure much better performance for streaming flows than the EFMP. Moreover, the ECCM give slightly better results than RAMAF – it is sufficient to run this mechanism once for 9 s while RAMAF has to operate at least once for 6 s.

In Figure 8, we can see that values of goodput (the mean rate of data successfully delivered to destination by elastic flows) slowly decrease (when using RAMAF) and are almost unchanged (when using ECCM) with increasing intensity of congestion mechanisms runs. Also in this case, the results obtained for the EFMP are worst. In this mechanism, new elastic flows may be accepted in congestion, which deteriorates transmission of other elastic flows. In RAMAF and in ECCM, the identifiers of such flows are deleted from the PFL in short time after acceptance. The mean value of goodput in AFAN without any congestion control mechanism was equal to $3.71 \pm 0.48$ Mbit/s. When we use RAMAF, and ECCM in particular, the obtained values are only insignificantly lower. Thus, we can conclude that both the RAMAF and the ECCM ensure a short acceptance delay of streaming flows without a significant impact on the other traffic in a network. The results obtained for the ECCM are slightly better than those observed for RAMAF.

The mean numbers of elastic flows accepted in the admission control block after congestion control operations for each analyzed mechanism are shown in Figure 9. For the basic AFAN, the mean number of flows with identifiers registered in the PFL was $22.9 \pm 0.53$. We can see that the values presented for EFMP increase with increasing values of the *CM_interval* parameter and are greater than the mean value noticed for the basic AFAN. However, the observed difference is not significant. On the other hand, we have to be aware that the EFMP consumes much more time than RAMAF or ECCM for its operations. As a result, usually more flows than it could be concluded from Figure 9 transmitted their traffic in the network, and that is why the goodput observed for the EFMP is poor. The number of elastic flows with identifiers registered in the PFL for RAMAF increase with increasing frequency of RAMAF operations. For *CM_interval* equal to 3, the observed value is twice greater than for basic AFAN. We can see that the observed values are correlated with goodput – when the number of identifiers of elastic flows in the PFL is greater, the goodput is lower. In ECCM, identifiers are not removed from the PFL, and new elastic flows are not accepted in congestion. The mean numbers of identifiers of elastic flows in the PFL are almost constant (the same is observed for goodput) and only slightly greater than for basic AFAN.



DOI: 10.1002/dac

Based on the analysis presented previously, we may conclude that ECCM ensures much better transmission parameters than EFMP and slightly better than RAMAF. Moreover, ECCM is less complex than both other mechanisms and easier to be implemented. We have to be aware that in ECCM, we do not need to remove identifiers of elastic flows to eliminate congestion. As a result, lower number of read and write operations in memory has to be performed. Moreover, the router's CPU has to process lower number of operations. Comparing the ECCM to RAMAF, we may assume that for *CM_interval* equal to 3 s in congested network, the congestion control mechanism may be performed even 75 times (*CM_interval* plus twice *FR* measurement interval) during 300 s of a simulation run. Assuming also that in RAMAF we have around 40 identifiers of flows in the PFL, we may estimate that we have to check the content of PFL 3000 times (40x75, read operation) than to write identifiers of 825 flows to PAFL (11x75, write operation) and finally to write identifiers of 825 flows to the PFL. In ECCM, we only have to set the *FR* to *min_FR* for 75 times (write operation). In both cases, for RAMAF and ECCM, we have also to set control parameters and to remove identifiers of new elastic flows from the PFL added during a congestion control mechanism operation. This short analysis shows that many more write/read operations have to be performed in RAMAF in comparison to ECCM, which confirms that ECCM is less complex than RAMAF.

We also made additional 50 long simulation runs for a 1 Gbit/s FAN link to show that the RAMAF and ECCM congestion control mechanisms are scalable. We examined the basic AFAN link, the RAMAF mechanism with two values of the *cleaning_timer* parameter (3 and 6 s), and the ECCM with two values of the *max_accept_delay* parameter (3 and 6 s). The simulation parameters for 100 Mbit/s and 1 Gbit/s FAN links are summarized in Table I.

The results of the last experiment are presented in Tables II and III. We can see also that in this case, the implementation of any of the mechanisms allows to minimize the acceptance delay of streaming flows. Moreover, the values of goodput show that the ECCM is slightly better than

Table I. Values of simulation parameters.

| Parameter | Value for 100 Mbit/s link | Value for 1 Gbit/s link |
|---|---|---|
| No. of simulation runs | 400 | 50 |
| Duration of a simulation run | 300 s | 300 s |
| No. of elastic flows (TCP) | 400 | 4000 |
| Size of elastic flows generated with Pareto distribution | k = 1.5, mean size = 210 Mbit | k = 1.5, mean size = 210 Mbit |
| Packet size of elastic flows | 1000 B | 1000 B |
| Interarrival of elastic flows generated with exponential distribution | mean interarrival time: 0.2 s | mean interarrival time: 0.2 s |
| No. of streaming flows (UDP) | 20 | 20 |
| Rate of streaming flows | 80 kbit/s | 80 kbit/s |
| Packet size of streaming flows | 100 B | 100 B |
| Interarrival of streaming flows genera--ted with exponential distribution | mean interarrival time: 1 s | mean interarrival time: 1 s |
| Capacity of FAN link | 100 Mbit/s | 10 Mbit/s |
| Capacity of access links | 1 Gbit/s | 100 Mbit/s |
| Size of buffer in R1 | 1000 packets | 10 000 packets |
| Measurement interval for the *PL* | 50 ms | 50 ms |
| Measurement interval for the *FR* | 500 ms | 500 ms |
| $max\_PL$ | 70% | 70% |
| $min\_FR$ | 5% | 5% |
| Flow time out | 20 s | 20 s |
| Warm-up time | 20 s | 20 s |
| $min\_th$ | 4000 packets | 4000 packets |
| $max\_th$ | 9000 packets | 9000 packets |
| $w_q$ | 0.002 | 0.002 |

*Notes*: FAN, flow-aware networks; *PL*, priority load; *FR*, fair rate; TCP, transmission control protocol; UDP, user datagram protocol; $min\_th$, minimum threshold; $max\_th$, maximum threshold; $w_q$, the queue weight.



DOI: 10.1002/dac

Table II. Acceptance delay of streaming flows in 1 G FAN link.

| CC Mechanism | *waiting_time* [s] | |
|---|---|---|
| Basic FAN | 82.70 ± 3.05 | |
| | *CM_interval* [s] | |
| | 3 | 6 |
| RAMAF | 2.42 ± 0.45 | 2.93 ± 0.40 |
| ECCM | 1.42 ± 0.70 | 2.57 ± 0.60 |

*Note*: FAN, flow-aware networks; RAMAF, remove and access most active flows; ECCM, efficient congestion control mechanism.

Table III. Goodput of elastic flows in 1 G FAN link.

| CC Mechanism | *goodput* [GB] | |
|---|---|---|
| Basic FAN | 4.42 ± 0.16 | |
| | *CM_interval* [s] | |
| | 3 | 6 |
| RAMAF | 3.32 ± 0.27 | 3.80 ± 0.26 |
| ECCM | 3.57 ± 0.11 | 4.28 ± 0.16 |

*Note*: FAN, flow-aware networks; RAMAF, remove and access most active flows; ECCM, efficient congestion control mechanism.

RAMAF. The transmission efficiency of elastic flows in AFAN with ECCM is similar to this observed for the basic AFAN. The results demonstrate that both the RAMAF and the ECCM congestion control mechanisms are scalable.

## 5. CONCLUSION

Approximate flow-aware networking is a new and promising concept for FAN. This is a simple and scalable architecture, which conforms to the network neutrality paradigm. It is also less complex than its predecessors. The RAMAF and the ECCM minimize acceptance delays of priority flows in XP routers. Moreover, they do not deteriorate transmission of other types of traffic in a network. Both these mechanisms ensure better performance in a network than the EFMP. However, the RAMAF is a complex mechanism and the number of operations to be made when it is implemented may be very high. The ECCM is a fully automated solution that does not degrade the transmission of elastic traffic in a network and, what is very important, it is easy to be implemented in XP routers. The number of operations to be made is low. The AFAN architecture with the ECCM is a complete solution to be used in the scalable and efficient Future Internet.

## ACKNOWLEDGEMENTS

This research was carried out with the support from the ‘Implementation of Cross-Protect Router’ project founded by the National Centre for Research and Development in Poland under the LIDER programme.

## REFERENCES

1. Labovitz C, Iekel-Johnson S, McPherson D, Oberheide J, Jahanian F. Internet inter-domain traffic. *ACM Sigcomm 2010*, New Delhi, India, 2010; 75–86.
2. Kortebi A, Oueslati S, Roberts J. Cross-protect: implicit service differentiation and admission control. *IEEE HPSR 2004*, Phoenix, USA, 2004; 56–60.
3. Domzal J, Jajszczyk A. Approximate flow-aware networking. *IEEE ICC 2009*, Dresden, Germany, 2009; 1–6.
4. Psounis K, Pan R, Prabhakar B. Approximate fair dropping for variable-length packets. *Micro, IEEE* 2001; **21**:48–56.

5. Kortebi A, Muscariello L, Oueslati S, Roberts J. On the scalability of fair queueing. *ACM HotNets-III*, San Diego, USA, 2004.
6. Domzal J, Wojcik R, Jajszczyk A. QoS-aware net neutrality. *First International Conference on Evolving Internet, 2009. INTERNET '09*, Cannes, France, 2009; 147–152.
7. Available from: OpenFlow, www.openflow.org (accessed April 10, 2014).
8. Domzal J. Intelligent routing in congested approximate flow-aware networks. *IEEE Globecom 2012*, Anaheim, USA, December 2012; 1751–1756.
9. Cuda D, Indre R-M, Le Rouzic E, Roberts J. Building a low-energy transparent optical wide area network with multipaths. *Journal of Optical Communications and Networking* 2013; **5**(1):56–67.
10. Post G, Noirie L. Method for processing data packets in flow-aware network nodes, 2012. US Patent 20,120,314,709.
11. Carofiglio G, Muscariello L. On the impact of TCP and per-flow scheduling on internet performance. *IEEE/ACM Transactions on Networking* 2012; **20**(2):620–633.
12. Domzal J, Wojcik R, Lopez V, Aracil J, Jajszczyk A. EFMP – a new congestion control mechanism for flow-aware networks. *Transactions on Emerging Telecommunications Technologies* 2014; **25**:1137–1148.
13. Domzal J, Dudek J, Jurkiewicz P, Romanski L, Wojcik R. The cross-protect router: implementation tests and opportunities. *IEEE Communications Magazine* 2014; **52**(9):115–123.
14. Gero BP, Palyi PL, Racz S. Flow-level performance analysis of a multi-rate system supporting stream and elastic services. *International Journal of Communication Systems* 2013; **26**(8):974–988.
15. Li Y, Yang Y, Zhou L, Wei A, Cao C. QoS-aware fair packet scheduling in IEEE 802.16 wireless mesh networks. *International Journal of Communication Systems* 2010; **23**(6-7):901–917.
16. Rubin I, Zhang R. Max–min utility fair flow management for networks with route diversity. *International Journal of Network Management* 2010; **20**(6):361–381.
17. Aweya J, Montuno DY, Ouellette M. Limitations of current ethernet switch architectures for enhanced flow control and service differentiation. *International Journal of Network Management* 2009; **19**(3):153–174.
18. Yu Y-T, Tong S-R. Adaptive transmission control protocol-trunking flow control mechanism for supporting proxy-assisted video on demand system. *International Journal of Communication Systems* 2012; **25**(10):1363–1380.
19. Floyd S, Jacobson V. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking* 1993; **1**:397–413.
20. Domzal J, Jajszczyk A. New congestion control mechanisms for flow-aware networks. *IEEE ICC 2008*, Beijing, China, 2008; 12–16.
21. Domzal J, Wojcik R, Jajszczyk A. Reliable transmission in flow-aware networks. *IEEE GLOBECOM 2009*, Honolulu, USA, 2009; 1–6.
22. Domzal J, Ansari N, Jajszczyk A. Congestion control in wireless flow-aware networks. *IEEE ICC 2011*, Kyoto, Japan, 2011; 1–6.
23. Network grade of service parameters and target values for circuit-switched services in the evolving ISDN, 1999. Rec. ITU-T E.721.
24. *Network simulator ns-2*, 2011. Available from: http://nsnam.isi.edu/nsnam (accessed April 10, 2014).




DOI: 10.1002/dac