Noname manuscript No.
(will be inserted by the editor)

Less Impact Better Service (LIBS)
A Service Paradigm for Internet Telephony

Lefteris Mamatas · Vassilis Tsaoussidis

Received: 18/8/2009 / Accepted: -

Abstract We discuss a new packet service paradigm, called "Less Impact Better Service" (LIBS). In simple terms, LIBS primarily schedules packets based on the delay they cause and cancels service differentiation policies when the cumulative delay due to prioritization becomes significant for non-prioritized packets. Based on LIBS, we evaluate different service policies that prioritize small packets using different service boundaries and we show that, by and large, LIBS satisfies better a number of applications with diverse demands in delay and throughput. We emphasize on Voice over IP applications, which are delay-sensitive but also utilize small packets and rates. Among other traditional performance measures, we also measure fairness in the context of LIBS, that is, we address the question whether the delay experienced per flow is proportional to the delay caused by that flow. We obtained very promising simulation results.

Keywords Service Differentiation · QoS · VoIP

1 Introduction

We propose a service differentiation scheme for small packets, particularly suitable for VoIP applications. We depart from a new service strategy, called "Less Impact Better Service" (LIBS), according to which, small packets that require minor service times and hence cause minor queuing delays, get some limited priority over long packets. The limitation is strictly associated with the cumulative service impact of prioritization on long packets. In [21], [22], [23], [27] we have shown that, based on service thresholds, the

L. Mamatas
University College London
Torrington Place
London WC1E 7JE, UK
E-mail: l.mamatas@ee.ucl.ac.uk

V. Tsaoussidis
Democritus University of Thrace
Vas. Sofias 12
67100 Xanthi, Greece
E-mail: vtsaousi@ee.duth.gr

service gains can be regulated for non-congestive applications, such as sensor applications or other types of applications that use small packets and rates, with almost-zero cost on congestive applications. Here, we apply the properties of LIBS one step beyond; we explore a differentiation scheme to satisfy better the delay requirements of VoIP applications.

In the context of delay-sensitive applications, bandwidth alone could not have a central role; instead, efficient distribution of resources needs to be characterized by the delay suffered by each flow in relation with the delay they cause. The latter is also occasionally associated with the delay that users tolerate. Our service approach promotes small-size packets at small rates, which define "non-congestive" traffic. To avoid starvation and also significant delay impact on congestive traffic, non-congestive traffic prioritization is confined by corresponding service thresholds. Hence, we analyze the behavior of systems where non-congestive traffic has controlled prioritization without affecting congestive traffic. From a user perspective, applications that utilize small data packets and rates (and are also intolerant to long delays) are satisfied, while other applications suffer almost-zero extra delays.

In particular, we apply LIBS to promote VoIP service, following three distinct directions: (i) a generic deterministic improvement, (ii) a stochastic approach, and (iii) a receiver-oriented approach for selected applications only. Our central differentiation scheme exhibits some distinctive but desirable properties:

1. Service granularity is improved: differentiation is feasible on a per-packet basis, not on a per-flow basis. Therefore, prioritized service will not apply when it is not necessitated, even within the same application flow.
2. User-perceived quality is not balanced by service degradation for other users. This happens because the minor occasional cost on congestive applications is not perceivable by their users.
3. Buffer management is costless and occasionally leads to better service for all applications.

Recent approaches to service differentiation rely on overlay architectures and services. They span across a spectrum of architectures and protocols that support call admission control that inevitably wastes resources and impacts other concurrent applications, or application marking along with priority scheduling that results on application-oriented instead of packet-oriented services. Due to its simplicity, the proposed strategy is easily deployable and practically useful; with a minor modification effort user service can have major improvement.

We evaluate LIBS with ns-2 [25] based simulations. We pay particular attention to the evaluation methodology, including the selected units of measurement that are suitable and appropriate. In this context, we use the *R-Factor*, the estimated *Mean Opinion Score (MOS)* and the *Application Satisfaction Index (ASI)*, a new metric for application fairness, which relies mainly on the delay rather than on throughput. *ASI* reflects how fairly applications receive service, delay-wise, under diverse delay expectations and impact [21].

In Section 2 we highlight our perspective in the context of related work. In Section 3 we discuss three distinct LIBS-based policies. In Section 4 we discuss our experimental evaluation methodology, metrics and scenarios. In section 5 we discuss our results and highlight the service gains and losses per policy. In Section 6 we conclude our paper.

2 Related Work & Discussion

Internet service differentiation is application-specific and naturally oriented either by some explicit and strict flow characteristics or by some application class. Even in the latter case, associating application types with service classes introduces inevitably some "classification" cost, relevant with the granularity of classification scale; and requires a rather sophisticated implementation, ranging from packet marking, to shaping, scheduling and dropping schemes. Beyond that, application-based network services inherently involve predetermined requests that the network - one way or another - needs to satisfy, leading it to a prescribed behavior that may not permit maximum system performance. Thus, application-based services are occasionally associated with limited operational flexibility for the network, which in turn may lead to degraded system performance.

Perhaps network engineering would have been different had the pressing demand of application requirements been ignored. For example, a natural principle to lead the design of network services (and consequently the service differentiation policy) could have been the network ability to function, the number of users serviced better without damaging the rest, or the service offered on the basis of the cost to other applications. For instance, it is not unreasonable to service first applications that require minimal time for service; in that case the gain for such applications can be significant, while the cost for the other applications may be small.

A similar, system-oriented scheduling concept has been studied in operating systems, where some schedulers select processes based on their completion time, rather than the time they started (shortest job first). Such a service alone may lead to starvation in case the rate of small processes is sufficient to keep the processor busy; processes demanding more time for completion could never get their turn. However, due to the cost of context switch, the lack of precision in estimating cost-per-process and the limited concurrent presence of processes, this domain had limited scheduling flexibility; our service differentiation scheme guarantees better service for non-congestive data only as far as the service of congestive applications is not degraded. In support of this goal, [1] confirms that the average delay for the system tends to be reduced when customers with short service times are given high priority.

A lot has been done in the networking community aiming at controlling traffic based on specific application requirements. According to the *intserv* approach [16], functionality of network components needs to allow for guarantees through signaling and reservation. In turn, *intserv* requires per - flow state information, which limits scalability. Other approaches require overlay architectures along with protocol modifications, which realize the corresponding relation between application classes and packet marks. Inevitably, class - based approaches (such as *diffserv* [17], [2], CBT[28] or [26]) overcome the limitation of state information but limit similar traffic to the same QoS class, introducing a "classification" error. For example, DCBT with ChIPS [5] extends CBT by providing dynamic thresholds and lower jitter for multimedia traffic. However, it assumes that all multimedia traffic require the same QoS. In [15], the authors introduced the Alternative Best Effort mechanism (ABE). ABE improves performance of delay-sensitive traffic but uses only two possible traffic classifications: delay - and throughput - sensitive. Delay - sensitive applications sacrifice throughput, and vice versa. Traffic Sensitive QoS mechanism (TSQ) [7] allows applications to indicate via marking their preferable delay / throughput sensitivity at packet-level. However, their approach does not imply service per-packet. That is, although applications may mark all or selected packets only, the basis for marking is not the current network

state, which is unknown[1] by the application, but rather the specific properties of a packet. The dynamics of such service interactions imply application-level QoS indeed. Application level QoS, has therefore two undesirable properties: first, it requires a prescription-based network operation, which confines network flexibility to reach an optimal operating point, system-wide, and second, it introduces a "classification" error. Note that in an effort to cancel the "classification" error one has to introduce more detailed categories and classes; that is, to trade it with processing overhead.

LIBS, however, can be integrated into application-oriented QoS strategies, producing a hierarchical service dynamic: service-oriented prioritization, with application - specific requests therein. For example, a favorably-marked large packet will not be serviced first if the following packet will have zero impact to its service. Therefore, different priorities can be assigned via packet marking to the different non-congestive or congestive applications.

We also study here the service scalability of LIBS, as a service-oriented strategy. For example, we introduce NCQ-VoIP, which is designed to achieve high performance for selected applications only. Other service strategies for Internet telephony include Low Latency Queuing (LLQ) [6]. LLQ is a mechanism that provides a strict Priority Queue (PQ) to Class-Based Weighted Fair Queuing (CBWFQ) [32]. With LLQ, delay-sensitive data (such as voice traffic) is dequeued and serviced first. In addition, voice traffic can be favored by a resource reservation protocol (such as RSVP [33]), which requests, collectively, a certain amount of bandwidth and latency at every RSVP-enabled network hop.

3 LIBS-based Policies

3.1 Non-Congestive Queuing (NCQ)

Non-Congestive Queuing (NCQ) [21] assumes that non-congestive traffic cannot exceed a predetermined threshold, called *ncqthresh*, which represents the upper limit of permitted prioritized service. The threshold typically reflects a service percentage for prioritization. However, this percentage corresponds to the number of packets; not the occupied buffer space. Indeed, since service prioritization applies for small packets only, the queue size that corresponds to the prioritized packets, percentage-wise, is much smaller.

Although the perspective of NCQ is more general, initially, we deal with two classes of packets: small packets (i.e., less than 150 bytes in our case), and long packets, which typical Internet applications use for data transfers. NCQ uses priority queuing to implement priority service. That is, within the same buffer, each packet is checked for its length, contrasted to the current state of prioritized service rate and gets priority whenever it satisfies two conditions: (i) length is below 150 bytes and (ii) prioritized service rate[2] is below *ncqthresh*.

At this stage of our work, we do not favor ACKs even though they have a small size. This strategy is expected to increase transmission rate; how far this can happen

[1] even if it is measured, the precision and granularity of measurements are dubious

[2] cumulative, not per-flow

Pseudocode 1 Non-Congestive Queuing

```
for every received packet
begin
   count received packets
   if (packet_length<150) and
      (favored_packets /
      received_packets < ncqthresh)
   then
      packet receives high priority
      count favored packets
   else
      packet receives normal priority
   end
end
```

(considering also the delayed-ACK scheme which is widely deployed) and how far it can impact congestion control, is an open issue. We note that NCQ may occasionally favor a part of a non-congestive data flow. However, a possible slight increase[3] of the re-ordered packets is counterbalanced by the large number of packet drops that are avoided due to the prioritization.

An issue that could be naturally raised is the following: what if applications adjust intentionally the size of their packets to an existing NCQ threshold in order to be favored? In [21], we show analytically that this strategy is usually inefficient due to the significant associated overhead. However, there are certain cases that an application can be favored from the NCQ mechanism, in case it adopts smaller packets. For example, some non-congestive applications may slightly reduce its packet size in order to have performance gains from a LIBS-based packet scheduling algorithm.

Furthermore, in case a malicious application tries to monopolize communication by transmitting small packets at high data rates, the extra introduced overhead is not counterbalanced by the gains. A Denial of Service attack (DoS) that uses aggressive flows consisting of small packets, in the worst case, would disable the prioritization mechanism of NCQ. However, in case the attack uses large packets, some non-congestive applications may not suffer from the exhaustion of the resources.

3.2 Stochastic NCQ (SNCQ)

In SNCQ we guarantee equal opportunities for all non-congestive packets, based on a *priority probability*, which is dynamically adjusted in reverse proportion to the service demand. That is, when the demand increases, the per-packet probability to receive prioritized service is reduced. *Priority Probability (PP)* is defined as:

$$PP = \frac{Priority\_Service\_Supply}{Priority\_Service\_Demand} =$$

$$= \frac{ncqthresh * number\_of\_received\_packets}{number\_of\_small\_packets}$$

We, then, apply the following algorithm:

[3] it is not significant in our experiments

Pseudocode 2 Stochastic Non-Congestive Queuing

```
for every received packet
begin
   count received packets
   count small packets
   (150-byte packets or smaller)
   if (ncqthresh * received_packets >
       small_packets) then
       PP = 1
   else
       PP = (ncqthresh * received_packets)
            / (small_packets)
   end
   small packets receive high priority
   with probability PP
end
```

3.3 NCQ for VoIP Applications (NCQ-VoIP)

Here, we attempt to apply LIBS towards a "better-guaranteed" service. That is, to allow selected applications to utilize more resources than others, however, within the frame of limited impact on others. Since the impact is limited, service guarantee is not always possible; and, since resource allocation favors selected applications only, application service is better than simply differentiated. We call this type of service: "better-guaranteed". In particular, NCQ-VoIP promotes selected VoIP calls by assigning higher probability to packets that belong to calls with bad quality.

Each receiver application calculates the voice quality for each call, based on the measured *R-Factor*. *R-Factor* quantifies voice quality and is a function of voice encoding type, packet-loss-rate and ear-to-mouth delay. A low-pass filter estimates the voice quality of the next packet of that particular call (i.e., smooth *R-Factor*):

$$Smooth\_RFactor = \alpha * Smooth\_RFactor + \beta * RFactor$$

We experimentally set the values of α and β to 0.35 and 0.65, respectively. We have carried out a considerable range of experiments with different α, β values: the above values produced the best results. A better justification and analysis on the the impact of the α, β parameters on the performance of NCQ-VoIP is a a subject of a future work.

The sender is notified about the estimated *R-Factor* value through ACK packets. The next departing packet carries the estimated voice quality[4] and is favored by the network, according the following priority probability (RP):

$$RP = sin(\frac{100 - RFactor}{100} * \frac{\pi}{2})$$

When *R-Factor* equals to 100[5], the priority probability is zero (no further prioritization is allowed). A zero *R-Factor* value (i.e., worst voice quality) gives a 100% probability for VoIP packets to be favored. We selected this specific probability function because it: (i) ranges from 0 to 1; (ii) achieves descent results (in terms of voice quality,

[4] The *R-Factor* is encoded in the available bits of the IP header (4 to 17 bits) [29]

[5] The value 100 represents the best voice quality

fairness and goodput). However, the choice of RP requires further study. Although we claim that currently RP is appropriate, we can't guarantee optimality.

We note that NCQ-VoIP requires an ACK-based transport mechanism or a similar receiver-oriented mechanism that informs the sender of the estimated R-Factor value. For cases it is not viable (i.e., for UDP-based VoIP communication), a network operator may either use NCQ or SNCQ.

4 Evaluation Methodology

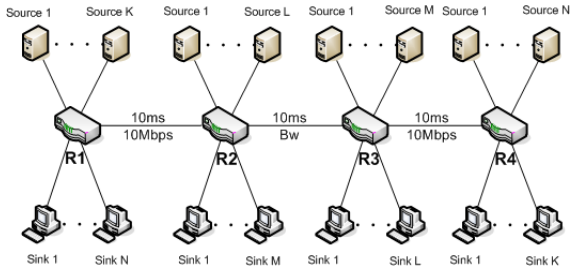


Fig. 1 Complex Network Topology

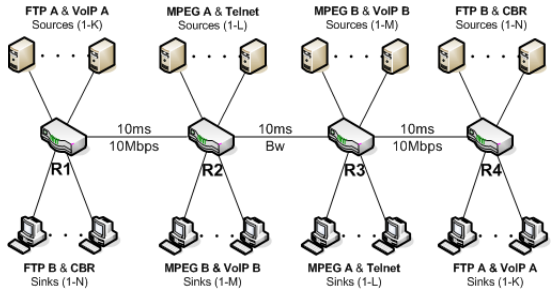


Fig. 2 Complex Network Topology and Different Traffic Patterns

We have implemented our evaluation plan on the ns-2 network simulator [25]. We simulated VoIP traffic based on the following assumptions: During a conversation, speakers alternate between activity and idle periods. Taking into consideration the ON and OFF periods [3], as well as the heavy-tailed characteristics and self-similarity of VoIP traffic [9], we used the *Pareto* distribution for modeling the call holding times. We configured *Pareto* with a mean rate to correspond to transmission rate of 64kbps and the shape parameter was set to 1.5. In accordance with [3], we distributed the ON and OFF periods with means of 1.0s and 1.35s, respectively. We simulated VoIP streams of 64kbps (as the widely-used ITU-T G.711 [20] coding standard) and we set packet sizes at 140 bytes.

In our scenarios, we do not consider the extra introduced delay due to the G.711 voice encoding (10 ms according to [8]) because we focus on the communication delay.

The number of congestive flows is either uniformly random distributed or ranges from 10 to 100. The TCP version we used is TCP NewReno and all implemented mechanisms are based on DropTail. In our evaluation, we focus on the impact of our modification and thus we use DropTail as a reference point. However, the LIBS-based paradigm can be also followed from variations of Active Queuing Management (AQM) algorithms (such as RED [12], BLUE [10] etc).

4.1 Evaluation Metrics

We measure application performance using *Goodput*, *Throughput* defined as:

$$Goodput = \frac{Original\_Data}{Time}$$

$$Throughput = \frac{Data\_Received}{Time}$$

where $Original\_Data$ is the number of bytes delivered to the high-level protocol at the receiver (i.e., excluding retransmitted packets and overhead), $Data\_Received$ the total number of bytes delived and $Time$ the amount of time required for the corresponding data delivery.

We characterize the quality of voice communication using the *R-Factor*, which is included in the E-Model [18], [19], an ITU-proposed analytic model of voice quality. *R-Factor* captures voice quality and ranges from 100 to 0, representing best and worst quality, respectively. *R-Factor* is also associated with the *Mean Opinion Score (MOS)*. *MOS* is the arithmetic average of opinions where "excellent" quality is represented by 5, "good" by 4, "fair" by 3, "poor" by 2 and "bad" by 1. *R-Factor* incorporates several different parameters, such as echo, background noise, signal loss, codec impairments and others. In [8], the authors simplified E-Model to transport-level measurable quantities and resulted in a more suitable *R-Factor* formula. At the beginning of their analysis, the authors expressed *R-Factor* as a sum of four terms:

$R = 100 - I_s - I_d - I_{ef} + A$

The parameter I_s reflects the signal-to-noise impairments associated with typical SCN paths. This is a function of several parameters, none of which are a function of the underlying packet transport. The authors used the default recommendations for all parameters except delay and packet loss and dropped the reference to the Expectation Factor (i.e., parameter A). So, the *R-Factor* expression was reduced to:

$R = 94.2 - I_d - I_{ef}$

Parameter I_d is the impairment associated with the mouth-to-ear delay of the path and is expressed as:

$I_d = 0.024d + 0.11(d - 177.3)H(d - 177.3)$

where d is the one-way delay (in milliseconds) and $H(x)$ is the Heavyside (or step) function.

Parameter I_{ef} is an equipment impairment factor associated with the losses within the gateway codecs. We note that there are no analytical expressions for the equipment impairments. So, the authors extracted estimates from subjective measurements. They defined parameter I_{ef} as:

$I_{ef} = -\gamma_1 - \gamma_2 ln(1 + \gamma_3 e)$

where e is the total loss probability, which takes values between 0 and 1, and the γ's are fitting parameters.

To summarize, we use here the following expression for the *R-Factor*:

$$R = \alpha - \beta_1 d - \beta_2(d - \beta_3)H(d - \beta_3) - \gamma_1 - \gamma_2 ln(1 + \gamma_3 e) \quad (1)$$

where $\alpha = 94.2$, $\beta_1 = 0.024ms^{-1}$, $\beta_2 = 0.11ms^{-1}$, $\beta_3 = 177.3ms$, d expresses the mouth-to-ear delay and e the packet loss rate. For the G.711 codec, $\gamma_1 = 0$, $\gamma_2 = 30$, $\gamma_3 = 15$.

More details on the parameters used in equation (1) can be found in [8].

The *R-Factor* is related to the *MOS* through the following set of expressions:

For $R < 0$**:** $MOS = 1$

For $R > 100$**:** $MOS = 4.5$

For $0 < R < 100$**:** $MOS = 1 + 0.035R + 7x10^{-6}(100 - R)$

For reference purposes, we give the relation of *R-Factor* with *MOS* in Table 1.

R-Factor	Quality of voice rating	MOS
$90 < R < 100$	Best	4.34 - 4.5
$80 < R < 90$	High	4.03 - 4.34
$70 < R < 80$	Medium	3.60 - 4.03
$60 < R < 70$	Low	3.10 - 3.60
$50 < R < 60$	Poor	2.58 - 3.10

Table 1 R-Factor, quality ratings and MOS

In order to measure fairness in the context of LIBS, we use the *Application Satisfaction Index (ASI)* [21]. *ASI* is defined as:

$$ASI = 1 - \frac{\left|\sum_{i=1}^{n} Delay_i - \frac{Data_i}{TotalData} Delay_{max}\right|}{nDelay_{max}}$$

where, n is the number of flows; $Data_i$ is the total transmitted data of the i_{th} flow to the receiver application; *TotalData* is the total transmitted data of all flows; $Delay_i$ is the average queuing delay of the i_{th} flow; and $Delay_{max}$ is the maximum queuing delay in the system. *ASI* ranges from 0 to 1.

Unlike other fairness indices (such as [4], [24], [31]), *ASI* captures the deviation of the actual delay from the expected delay per flow. Note, however, that expected delay is determined by the factor $\frac{Data_i}{TotalData}$. In this context, *ASI* represents the fairness of the *LIBS* architecture, since the expected delay per packet (and in turn, per flow) grows in proportion to the volume of their transmitted packets.

4.2 Evaluation Scenarios

In our experimental analysis, we use three distinct scenarios[6], starting from a complex topology (Figure 1) that incorporates multiple bottlenecks, cross and reverse traffic. Here, we evaluate NCQ, SNCQ and NCQ-VoIP. In the next scenario, we explore whether a LIBS-based mechanism can be incrementally deployed. Such a scenario is important because the incremental deployment of a new network service or protocol is typically a hard problem, especially when it has to be deployed in the routers [13].

[6] additional scenarios that evaluate NCQ can be found in [21]

In the last scenario, we evaluate the mechanisms using a complex topology, five different application types and a random number of users (Figure 2). We detail the three scenarios below:

Scenario 1: LIBS-based mechanisms in a Complex Topology: In this scenario, we carry out experiments that show the impact of NCQ, Stochastic NCQ and NCQ-VoIP in terms of *Goodput*, *Throughput*, *Application Satisfaction Index* and *R-Factor*. We use the topology of Figure 1, which incorporates multiple bottlenecks, cross and reverse traffic. The FTP and VoIP flows transmit data from "source nodes" to the corresponding "sink nodes". The number of FTP flows is uniformly random distributed in [1, 50] and the number of VoIP calls in [1, 10]. We set the *ncqthresh* to 0.03 and the *bw* to 20Mbps.

Scenario 2: Incremental Deployment of LIBS-based Mechanisms: In this scenario, we range the number of routers that incorporate the LIBS-based scheme from 0 to 4. The *ncqthresh* value is set to 0.03 and we have a random number of applications for both congestive and non-congestive traffic; uniformly random distributed in [1, 50] and [1, 10], respectively. The *bw* value is 10Mbps. We measure *Goodput*, *Application Satisfaction Index*, *R-Factor* and *MOS*.

Scenario 3: LIBS-based mechanisms & Different Traffic Patterns: In the last scenario, five different applications (FTP, MPEG, VoIP, Telnet, Sensor Applications) utilize common network resources using the topology illustrated in Figure 2. The number of congestive applications (FTP, MPEG) is uniformly random distributed in [1, 50] and the number of non-congestive applications (VoIP, Telnet, Sensor Applications) in [1, 10]. The sensor applications are simulated using simple Constant Bit Rate (CBR) sources. We set the interval time for both CBR and Telnet applications to 1 sec. The MPEG applications transmit data according to a trace file that is based on a real movie [11]. The value of *ncqthresh* is now 0.03 and the *bw* 20Mbps. We measured *Goodput* for all applications and *R-Factor* for the VoIP calls.

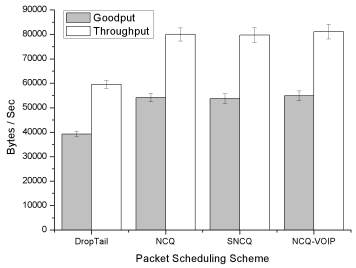
The queue size for all scenarios is 100 packets. Each experiment's duration is 60 sec. All experiments are non-deterministic. We performed 50 runs for each non-deterministic experiment in order to confirm the statistical accuracy of the results.

The distribution of our measurements is often high due to the nature of our experiment. For example, in the case of randomized contention, one flow receives almost always less throughput than 50 flows. In such results, the standard deviation for the measurements does not increase by the number of experimental runs. Each time the standard deviation of the differences between the average value and the samples is very small (e.g., 1-3%), we depict the average values and the confidence intervals (i.e., the above standard deviation). Otherwise, we analyze and interpret our results using frequency tables. More precisely, we group measurements into suitable intervals that allow us to find the most common values. Each figure that uses a frequency table, depicts the percentage of performance improvement/decrease of each measure, having the experimental results of a scenario using DropTail as a reference point.

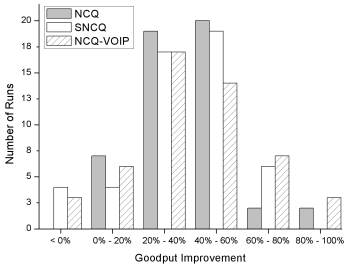
5 Simulation Results

5.1 Scenario 1: LIBS-based mechanisms in a Complex Topology

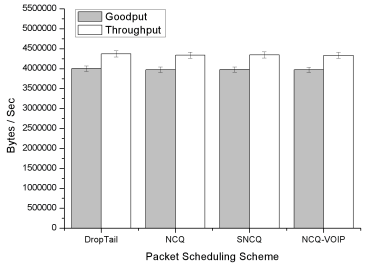
All mechanisms improve both the *Goodput* and *Throughput* for the VoIP applications (Figures 3(a), 3(b)). More precisely, in the 100%, 92% and 94% of the runs, the VoIP



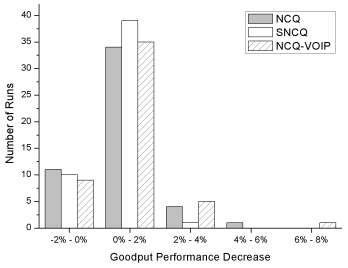
(a) Goodput/Throughput of VoIP Applications



(b) Goodput Improvement for VoIP Applications



(c) Goodput/Throughput of Congestive Applications

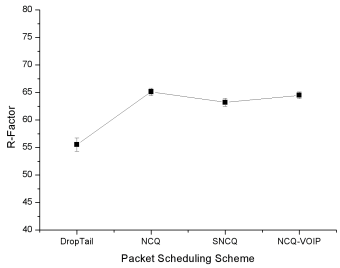


(d) Goodput Improvement for Congestive Applications

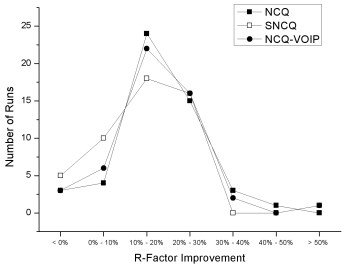
Fig. 3 Performance Results in Terms of Goodput/Throughput

flows improve their *Goodput* in case of NCQ, SNCQ and NCQ-VoIP, respectively (Figure 3(b)). The average *Goodput* values show statistically zero impact on the congestive flows (Figure 3(c)). However, a more clear view comes from a frequency table that groups the measurements according the most common ones, due to the high distribution of the values (Figure 3(d)). According to Figure 3(d), in the majority of the runs (i.e., the 68% for NCQ, the 78% for SNCQ and the 70% for NCQ-VoIP) the congestive flows have a slight performance decrease (0% to 2% - Figure 3(d)). According to Figure 3(d), in a smaller percentage of the runs, the congestive flows achieve a performance improvement (e.g., 22% for NCQ). Furthermore, in the majority of the runs, the non-congestive flows (e.g., 78% for NCQ) improve their *Goodput* from 20% to 60%.

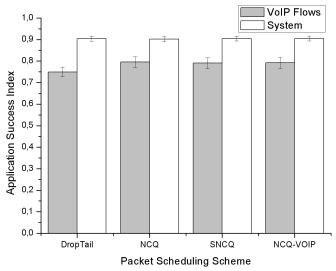
In Figures 4(a), 4(b) we illustrate the voice quality of the participating calls. In the majority of the runs (e.g., 92% for NCQ), the VoIP flows achieve a performance improvement from 10% to 40%. Both NCQ and NCQ-VoIP improve voice quality more than SNCQ (Figures 4(a), 4(b)). Although in Figure 4(c) all mechanisms appear to have the same *Application Satisfaction Index*, according to 4(d), NCQ-VoIP appears more fair than the other two mechanisms. The latter result is not reflected to the average value of *ASI* due to the high distribution of the values (Figure 4(c)).



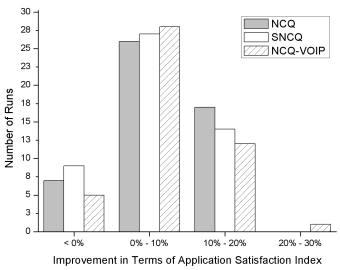
(a) R-Factor of VoIP Applications



(b) R-Factor Improvement



(c) Application Satisfaction Index



(d) Improvement in Terms of Application Satisfaction Index for the VoIP Flows

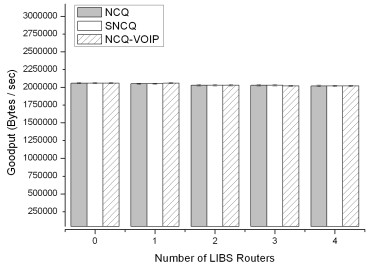
Fig. 4 Impact on Voice Quality & Fairness

5.2 Scenario 2: Incremental Deployment of LIBS-based Mechanisms

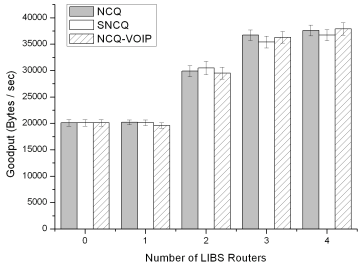
In this scenario, we adjust the number of LIBS routers[7] from 0 to 4, assuming incremental deployment. In figure 5(a), we have only a minor impact on the *Goodput* of the congestive flows for any number of LIBS routers. However, the *Goodput* of the non-congestive applications is significantly improved (Figures 5(b), 5(c), 5(d)). When a LIBS-based mechanism is fully deployed (4 LIBS routers - Figure 5(d)), NCQ-VoIP improves the *Goodput* of the VoIP applications slightly more than NCQ and SNCQ. For example, NCQ-VoIP achieves more than 100% improvement for the 44% of the runs while both NCQ and SNCQ for the 38%. In the case of 2 LIBS routers, the calls increase their *Goodput* (from 1% to 200%) in the 96% of the runs for NCQ, in the 92% for SNCQ and in the 88% for NCQ-VoIP. NCQ achieves a major improvement (over 80%) for the 18% of the runs and both SNCQ and NCQ-VoIP for the 20%.

The significant improvement in terms of *Goodput* for the VoIP applications is reflected in the voice quality of each call. For example, the average R-Factor of the non-congestive applications increases by the number of the LIBS routers (Figure 6(a)). In the case of 75% deployment (3 LIBS routers), in the majority of runs (94% - 96%), the VoIP flows achieve improvement from 0% to 50%. The improvement in terms of *Fairness* is also significant: in the 80% - 82% of the runs we have an improvement on

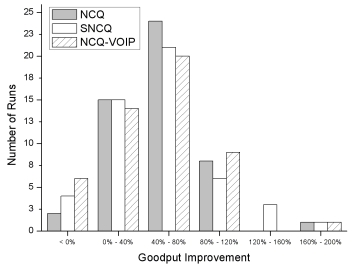
[7] A LIBS router incorporates a LIBS-based mechanism (i.e., one of NCQ, SNCQ or NCQ-VoIP)



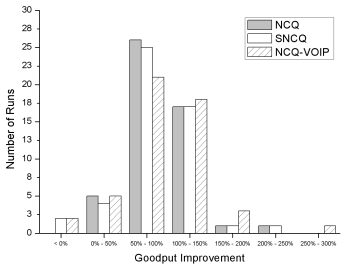
(a) Goodput of Congestive Applications



(b) Goodput of VoIP Applications



(c) Goodput Improvement of VoIP Applications (2 LIBS Routers)



(d) Goodput Improvement of VoIP Applications (4 LIBS Routers)

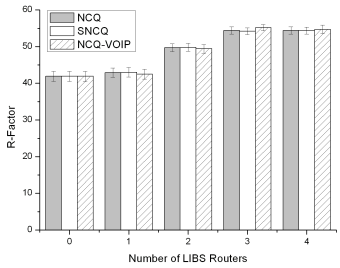
Fig. 5 Performance Results in Terms of Goodput

the *Application Satisfaction Index* (Figure 6(d)). In the same case, NCQ appears more fair than SNCQ and SNCQ more than NCQ-VoIP. In the case of fully deployment (4 LIBS routers), the *Mean Opinion Score* of the calls have a 20% to 40% increase for the majority of the runs (Figure 6(c)).

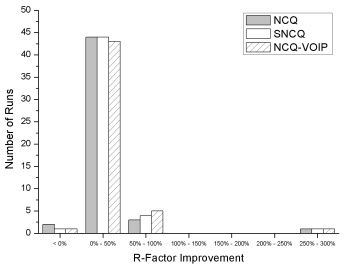
5.3 Scenario 3: LIBS-based mechanisms & Different Traffic Patterns

In the last scenario, we evaluate the three proposed mechanisms using a more complex scenario that incorporates different types of congestive (i.e., FTP and MPEG) and non-congestive applications (i.e., VoIP, Telnet and sensor applications), transmitting data concurrently. For all groups of congestive applications (FTP A, B and MPEG A, B), in the majority of runs we have a slight performance decrease (i.e., 0% - 5%) in terms of *Goodput* (Figures 7(a), 7(b), 7(c), 7(d)). In all cases, NCQ-VoIP appears more friendly to the congestive flows. More precisely, NCQ-VoIP has the higher probability for a run to have a performance increase in terms of *Goodput* for the congestive flows, that is 10% for FTP group A (Figure 7(a)), 6% for MPEG group A (Figure 7(b)), 44% for MPEG group B (Figure 7(c)) and 50% for FTP group B (Figure 7(d)).

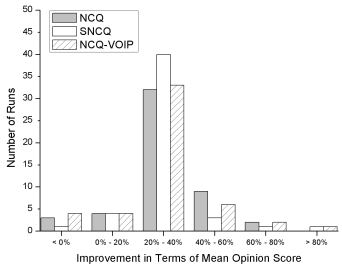
In the group A of VoIP applications, in the 84% of the runs we have a performance improvement in terms of R-Factor for NCQ, in the 82% for SNCQ and in 80% for NCQ-VoIP (Figure 8(a)). In the group B of VoIP applications, in the 76% of runs -



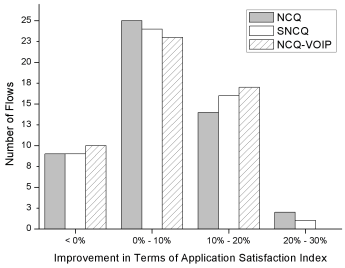
(a) R-Factor of VoIP Applications



(b) R-Factor Improvement (3 LIBS Routers)



(c) MOS Improvement (4 LIBS Routers)



(d) Improvement in Terms of Application Satisfaction Index (3 LIBS Routers)

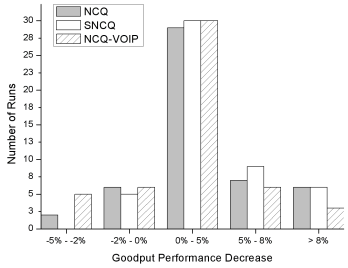
Fig. 6 Voice Quality & Fairness

for all mechanisms - the voice calls achieved improvement (Figure 8(b)). In the case of Telnet applications, in the 74% of runs for NCQ we have a performance increase upto 10% and in the 68% of runs for NCQ-VoIP upto 40% (Figure 8(c)). However, in the case of SNCQ, we have a performance increase only for the 48% of the runs (Figure 8(d)). It is very difficult to have an accurate justification for this specific result, due to the high complexity of the scenario. Consequently, it calls for further investigation.

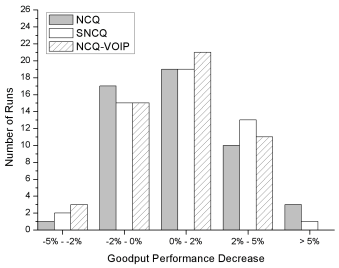
6 Conclusions

We have shown that LIBS is a simple but powerful tool for service differentiation, particularly beneficial for applications that utilize small rates and short packets, such as typical VoIP. We discussed three alternative realizations of LIBS, favoring simplicity (NCQ), sophistication (SNCQ) and quality (NCQ-VoIP). For example, NCQ-VoIP performs well in terms of voice quality, but requires a receiver-oriented mechanism which signals the voice quality measurement to the sender. Clearly, the deployment of the mechanism has an extra difficulty: requires modification of the receiver protocol. We note that other well known proposals use similar approaches, e.g., [14], [30]. In case this is not feasible, a network operator may either use NCQ or SNCQ.

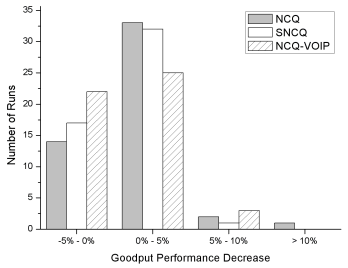
We presented simulation results that point out the following: the application-level performance (such as voice quality) may be significantly improved by favoring only a



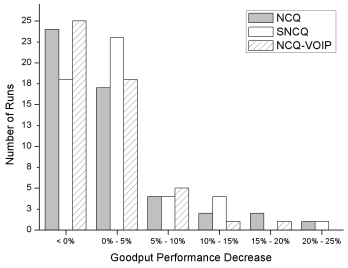
(a) Goodput Performance Decrease (Group A of FTP Applications)



(b) Goodput Performance Decrease (Group A of MPEG Applications)



(c) Goodput Performance Decrease (Group B of MPEG Applications)



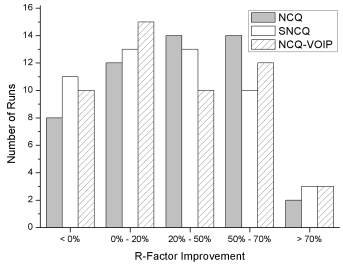
(d) Goodput Performance Decrease (Group B of FTP Applications)

Fig. 7 Goodput of Congestive Applications

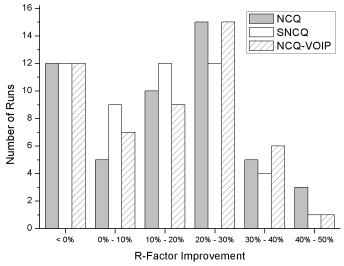
part of a data flow - avoiding a negative impact on the other flows. We have shown that, practically, LIBS can satisfy better a number of users; a leading design issue in today's Internet, which creates hopes for deployment success.

References

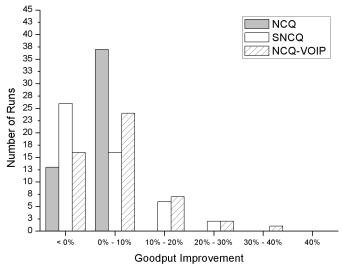
1. D. Bertsekas and R. Gallager. *Data Networks.* Prentice-Hall, Englewood Cliffs, New Jersey, 1987 (2nd Ed. 1991).
2. R. Bless, K. Nichols, and K. Wehrle. A Lower Effort Per-Domain Behavior (PDB) for Differentiated Services, 2003.
3. P. Brady. A Statistical Analysis of On-Off Patterns in 16 Conversations. *The Bell System Technical Journal*, 47:73–91, Jan 1968.
4. D. Chiu and R. Jain. Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks. *Journal of Computer Networks and ISDN*, 17(1):1–14, June 1989.
5. J. Chung and M. Claypool. Dynamic-CBT and ChIPS Router Support for Improved Multimedia Performance on the Internet. In *MULTIMEDIA '00: Proceedings of the Eighth ACM International Conference on Multimedia*, pages 239–248, New York, NY, USA, 2000. ACM Press.
6. Cisco. Low Latency Queuing. http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120t/120t7/pqcbwfq.htm, 2005.
7. M. Claypool, R. Kinicki, and A. Kumar. Traffic Sensitive Active Queue Management. In *Proceedings of the 8th IEEE Global Internet Symposium*, Miami, Florida, March 2005. IEEE.



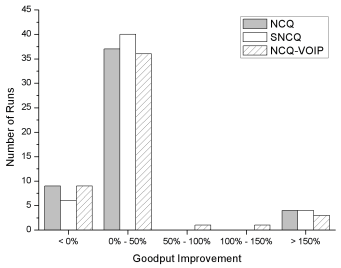
(a) R-Factor Improvement (Group A of VoIP Applications)



(b) R-Factor Improvement (Group B of VoIP Applications)



(c) Goodput Improvement (Telnet Applications)



(d) Goodput Improvement (Sensor Applications - CBR)

Fig. 8 R-Factor & Goodput of Non-congestive Applications

8. R. Cole and J. Rosenbluth. Voice over IP Performance Monitoring. *ACM SIGCOMM Computer Communication Review*, 31(2):9–24, April 2001.
9. T. D. Dang, B. Sonkoly, and S. Molnar. Fractal Analysis and Modeling of VoIP Traffic. In *Proceedings of 11th International Telecommunications Network Strategy and Planning Symposium*, Austria, June 2004. NETWORKS 2004.
10. W. Feng, D. Kandlur, D. Saha, and K. G. Shin. BLUE: A New Class of Active Queue Management Algorithms. Technical Report CSE-TR-387-99, University of Michigan, April 1999.
11. F. Fitzek and M. Reisslein. MPEG-4 and H. 263 Video Traces for Network Performance Evaluation. *Network, IEEE*, 15(6):40–54, 2001.
12. S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.
13. X. He, C. Papadopoulos, and P. Radoslavov. A Framework for Incremental Deployment Strategies for Router-assisted Services. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 2:1488– 1498, 2003.
14. C. Huitema. Real Time Control Protocol (RTCP) Attribute in Session Description Protocol (SDP). Technical report, RFC 3605, October 2003.
15. P. Hurley, J. Boudec, P. Thiran, and M. Kara. ABE: Providing a Low-Delay Service within Best-Effort. *IEEE Network*, 15(3):60–69, 2001.
16. IETF. Integrated services charter. http://www.ietf.org/html.charters/OLD/intserv-charter.html, 2000.
17. IETF. Differentiated services charter. http://www.ietf.org/html.charters/OLD/diffserv-charter.html, 2003.
18. ITU-T Recommendation G.107. The E-Model, a Computational model for use in Transmission Planning, December 1998.

19. ITU-T Recommendation G.113. General Characteristics of General Telephone Connections and Telephone Circuits - Transmission Impairments, February 1996.
20. ITU-T Recommendation G.711. Pulse Code Modulation (PCM) of Voice Frequencies, November 1988.
21. L.Mamatas and V.Tsaoussidis. Differentiating Services with Non-Congestive Queuing (NCQ). *IEEE Transactions on Computers*, 58(5):591–604, May 2009.
22. L. Mamatas and V. Tsaoussidis. A new approach to Service Differentiation: Non-Congestive Queuing. In *ICST First International Workshop on Convergence of Heterogeneous Wireless Networks (CONWIN2005)*, Budapest, Hungary, July 2006. ICST.
23. L. Mamatas and V. Tsaoussidis. Differentiating Services for Sensor Internetworking. In *IFIP Fifth Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2007)*, Corfu, Greece, June 2007. IFIP.
24. M. Marsan and M. Gerla. Fairness in Local Computing Networks. In *IEEE ICC '82*, Philadelphia, June 1982. IEEE.
25. S. McCanne and S. Floyd. NS-2 Network Simulator, 1997.
26. W. Noureddine and F. Tobagi. Improving the performance of interactive tcp applications using service differentiation. *Computer Networks*, 40(1):19–43, 2002.
27. G. Papastergiou, C. Georgiou, L. Mamatas, and V. Tsaoussidis. On Short Packets First: A Delay-oriented Prioritization Policy. *International Journal of Communication Systems, Wiley*, 2009. Accepted.
28. M. Parris, K. Jeffay, and F. Smith. Lightweight Active Router-Queue Management for Multimedia Networking. In *Proceedings of SPIE Conference on Multimedia Computing and Networking*, volume 3654, San Jose, California, January 1999. SPIE.
29. I. Stoica and H. Zhang. Providing Guaranteed Services without Per Flow Management. *ACM SIGCOMM Computer Communication Review*, 29(4):81–94, October 1999.
30. V. Tsaoussidis and C. Zhang. TCP-Real: Receiver-oriented Congestion Control. *Computer Networks*, 40(4):477–497, 2002.
31. V.Tsaoussidis and C.Zhang. The Dynamics of Responsiveness and Smoothness in Heterogeneous Networks. *IEEE Journal on Selected Areas in Communications*, 23(6):1178–1189, 2005.
32. H. Zhang. Service Disciplines for Guaranteed Performance Service In Packet-switching Networks. *Proceedings of the IEEE*, 83(10):1374–1396, 1995.
33. L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A New Resource Reservation Protocol. *Communications Magazine, IEEE*, 40(5):116–127, 2002.