Time-aware Utility-based QoS Optimisation

Calin Curescu and Simin Nadjm-Tehrani

Department of Computer and Information Science
Linköping University
E-mail: calcu/simin@ida.liu.se*

Abstract

This paper presents a time-aware admission control and resource allocation scheme in the context of a future generation mobile network. The quality levels (and their respective utility) of the different connections are specified using discrete resource-utility (R-U) functions. The scheme uses these R-U functions for allocating and reallocating bandwidth to connections, aiming to maximise the accumulated utility of the system. However, different applications react differently to resource reallocations. Therefore at each allocation timepoint we take into account the following factors: the age of the connection, a drop (disconnection) penalty and the sensitiveness to reallocation frequency. Finally, we show the superior performance of our approach compared to a recent adaptive bandwidth allocation scheme.

1. Introduction

A key component of future generation wireless networks is to enable mobile users to use multimedia and data services seamlessly. The bursty nature and variable bandwidth needs of most of the new services call for novel treatments of the network resource management so that application needs are satisfied, and at the same time network provider resources are optimally used. Many existing works in resource allocation focus on one part of this equation to the detriment of the other party. If end-to-end guarantees of user Quality of Service (QoS) requirements are in focus, then some decisions may become suboptimal seen from a system-level perspective, and vice versa. In this paper we approach the problem by methods that bridge this gap.

This paper presents a bandwidth allocation and admission control mechanism to be used in a radio network cell of a future generation communication network. As the only bottleneck we consider the bandwidth of the wireless link between the user equipment (UE) and the base transceiver station (BTS).

The different nature of the wireless channel (as compared to the wireline) makes the QoS delivery more challenging due to three factors. First, the fixed capacity of the allocated wireless spectrum makes them bandwidth-constrained, and hence allocation problems cannot be solved by over-provisioning. Second, the effective bandwidth of the wireless link may fluctuate due to fading and interferences. Third, due to user mobility, the resources available to a user (on a cell to cell basis) might vary greatly during the lifetime of a connection.

The above three factors describe a system where bandwidth availability is highly variable in time, and the system may often find itself in an overload situation. For the bandwidth manager to take the best allocation decisions, we assume that a quantitative measure of the utility (benefit) generated by each connection is available. One way to capture the application-specific perceived quality depending on resource availability is via resource-utility (R-U) functions. Consequently, a straightforward allocation optimisation criterion (which can be easily linked to network operator revenues) is the maximisation of system utility. This can be calculated as the sum of each connections[1] utility.

Moreover, for such an open, dynamic system, resource reallocation might be needed in order to increase total utility or to provide graceful degradation. In order to make more informed decisions on resource reallocation, we also consider, in addition to utility functions, the fact that different applications react differently to resource reallocation. For example, if a hard real-time application is degraded, we would expect no utility from this application. On the other hand, an FTP session will have no restriction to switch between different resource allocation levels, no matter how often.

Therefore we propose a Time-Aware Resource Alloca-

[1] Since each connection represents an application the two terms will be regarded as interchangeable in the following text.

tion scheme (TARA) that aims to allocate bandwidth such that the accrued utility of the whole cell, accumulated over time is maximised. The novelty is that our scheme identifies how resource reallocation decisions affect the utility of the application and integrates this information into the bandwidth management algorithms. We specifically consider the effect of resource change for three different classes of applications. As time goes by, decisions taken by the resource manager will affect the applications in different ways: some will completely lose the accrued utility up to now if they are deprived of bandwidth, some are more sensitive to being dropped, some connections will be more sensitive to bandwidth variations than others. Therefore, we consider changes to utility with respect to three factors: age of the connection, an attached drop (disconnection) penalty, and sensitiveness to reallocation frequency. We treat these factors differently depending on the application class. To evaluate our scheme we have built a simulation platform in which we compare our approach with a reallocation unaware version and a recent published algorithm, namely the Rate-based Bandwidth Borrowing Scheme (RBBS) [5].

The paper is organised as follows. In the reminder of this section we review other approaches to QoS provisioning. Section 2 presents background information about resource-dependent utility maximisation. In Section 3 we identify the factors affecting the utility of a connection if reallocations are performed and incorporate them in our scheme. Sections 4 and 5 present the evaluation setup and the simulation results of our allocation scheme as compared to other. Section 6 concludes the paper.

1.1. Related work

Research on QoS provisioning may pursue different goals. While some research is geared towards end-to-end architectures [1], others address issues at end-system level or network layers. Mechanisms like Intserv, Diffserv, RSVP [3, 2, 4] provide the means of enforcing the necessary QoS parameters (like bandwidth, delay, packet loss probability).

While many applications can be run at different QoS levels, corresponding to a range of resource allocations, without knowing an associated importance value, the QoS management system will not be able to prioritise allocations during overloads. Utility functions provide an appropriate way to specify a quantitative measure of the QoS perceived by the application [9, 10]. The advantage of utility functions over run-time adaptation is that the management system knows a-priori about the value corresponding to different resource allocations and can enforce an optimal solution. Chen Lee et al. [9] use resource-utility functions in a QoS management framework with the goal to maximise the total utility of the system. They propose two approximation algorithms, and compare the run-times and solution quality with an optimal solution based on dynamic programming. In our work we build on top of such a utility maximisation algorithm, but we also take into consideration the effect that bandwidth reallocations have on the connections' generated utility.

Rui-Feng Liao et al. [10] also use "utility functions" in a bandwidth allocation scheme for wireless packet networks. However as opposed to maximising the total utility of the system, they provide "utility fair allocation" to the connections. Their algorithm extends "max-min fair allocation", with utility replacing bandwidth as the fairness criterion. While this scheme provides equality to all connections, it might have counterproductive effects during overload conditions, since it degrades all the existing connection to a low common utility.

Another approach [11, 5], geared towards mobile networks, proposes adaptive bandwidth allocation schemes without an explicit use of utilities. These use a flexible allocation approach, where connections specify a mandatory minimal bandwidth and an ideal maximal bandwidth. Also, both schemes differentiate between real-time and best-effort connections. In the work of Oliveira et al. [11], the allocated amount of bandwidth during the stay in a cell is fixed, it can be changed only at a handoff. El-Kadi et al. [5] provide a more adaptive scheme, by allowing bandwidth to be borrowed from already accepted connections. Although the scheme is adaptive, it does not include a quantitative measure of the importance of the different connections.

In their QoS provisioning system [12] Richardson et al. take a lower layer approach, by using value based and real-time scheduling techniques and working at the packet scheduling level. The value of each packet depends on the value of the connection it belongs to and on its deadline. Total system utility is used to measure system performance.

2. Background

To explain how our bandwidth allocation scheme maximises the total system utility, we must first present the notion of bandwidth dependent utility function, and a utility maximisation algorithm.

2.1. Application utility

The utility of an application (and its associated connection) represents the value assigned by the user to the quality of the application's results. In order to evaluate the utility generated by different resource allocations, we assume that all connections have an R-U function, which is specified by the user of this connection. Since we are concerned with bandwidth allocation, we are interested only in a bandwidth-utility function $u_i : \mathbb{R}^* \rightarrow \mathbb{R}^*$, i identifies the connection and $\mathbb{R}^*$ is the set of non-negative rational numbers. As a reflection of variety of applications, utility functions may exhibit different patterns: concave, convex, linear or step functions, the only restriction being that a R-U function should be non-decreasing.

For the ease of implementation, and to keep complexity low, it is necessary to quantise the utility functions using a small set of parameters. Thus, the utility function can be represented by a list of bandwidth-utility pairs, in increasing order of resource consumption [9]:

$$u_i = \left(\begin{pmatrix} u_{i,1} \\ b_{i,1} \end{pmatrix}, ..., \begin{pmatrix} u_{i,k} \\ b_{i,k} \end{pmatrix} \right)$$

where k is the number of utility levels of connection i.

2.2. System utility maximisation

By using the individual utility functions, and assuming that a certain allocation has been performed, we can compute the utility of the whole system as the sum of the utilities generated by all n connections, $u : \mathbb{R^*}^n \rightarrow \mathbb{R}^*$

$$u(b_1, ..., b_n) = \sum_{i=1}^{n} u_i(b_i)$$

where b_i represents the allocated bandwidth and $u_i(b_i)$ the accrued utility of connection i. Maximising the system utility u is subject to the following constraint:

$$\sum_{i=1}^{n} b_i \leq b\_max$$

where b_max is the maximum bandwidth available in the cell.

The above allocation optimisation problem is an NP-hard problem closely related to the knapsack problem; Lee et al. present several approximation algorithms to solve it [8, 9]. As an input they accept R-U functions, the output being the different resource allocation quotas.

As a basic ingredient in our scheme we use a low complexity algorithm proposed by Lee et al. that nevertheless generates solutions close to the optimal solution [9]. The convex hull optimisation algorithm (referred as $asrmd1$ in [9]) is based on the following observation: given several piece-wise linear, concave R-U functions, concatenating the segments from all the input R-U functions in a decreasing slope order, yields an optimal system-wide R-U function. Bandwidth is allocated to the segments in this order, until depleted. Note that the slope of each segment, $(u_{i,j} - u_{i,j-1})/(b_{i,j} - b_{i,j-1})$, is the criterion on which bandwidth is allocated, first to the more efficient segments (with a higher slope). Since not all R-U functions are concave, the algorithm first approximates all R-U functions by their convex hull frontier.

3. Time-aware QoS optimisation

The R-U functions present the bandwidth-utility dependency in a static manner. These functions might be used in a non-adaptive architecture, where once bandwidth is allocated, it is allocated for the whole duration of the connection. In a dynamic system, where resources need to be reallocated, the utility given by a R-U function will represent only a momentary value ($u_i(t)$). A better measure of the utility generated by a connection would be its accumulated utility (in time), which is the utility generated by the connection over its entire duration.

If the accumulated utility of a connection (u_i^a) corresponds to the integral of all the momentary utilities, that is $u_i^a = \int_0^T u_i(t)dt$, then the following equality holds:

$$u^a = \sum_{i=0}^{n} u_i^a = \sum_{i=0}^{n} \int_0^T u_i(t)dt = \int_0^T \sum_{i=0}^{n} u_i(t)dt = \int_0^T u(t)dt$$

where u^a denotes the system-wide utility accumulated over time and is defined as $u^a = \sum_{i=0}^{n} u_i^a$ and $u(t)$ is the momentary system-wide utility, defined as $u(t) = \sum_{i=0}^{n} u_i(t)$. T is an arbitrary time point and n is the number of all connections that arrived at the system before T. The above equality shows that under this assumption, the maximisation of u_i^a can be achieved by maximising $u_i(t)$ at each time point t.

However, for some application classes $u_i^a \neq \int_0^T u_i(t)dt$. For example, if a voice connection is deprived of bandwidth before the natural end, we can safely assume that all the potential utility generated while it was active has been lost. In the end $u_i^a = 0$, and resources allocated for the duration of the call have been wasted. Therefore, our allocation algorithm needs to take into account the effect reallocations have on the accumulated utility of the connections.

3.1. Dominant factors

We have identified several factors that determine the accumulated utility of a connection, and describe them in the next subsections.

3.1.1. Connection classes. Accepting a connection and allocating a certain amount of bandwidth can be seen as an allocation contract between the user and provider. Each reallocation would thus amount to a contract breach and signing of a new contract. Because of different application types or user preferences, different connections have different tolerance to bandwidth reallocation. We have identified three connection classes depending on their adaptability to bandwidth reallocation. Consequently, the connection class determines the base function by which the accumulated utility is calculated (charged to the user).

Class I represents non-adaptive connections. These are connections that once accepted, with initial utility u_i^{init}, the resource amount cannot be re-negotiated. If the management system cannot assure the initial resource amount at any time-point during the lifetime of the connection, there will be no utility gained for the whole duration of the connection and the connection should be dropped. If the connection is not dropped, the accumulated utility of the connection is calculated by this formula: $\boldsymbol{u_i^a = u_i^{init} \times duration}$. Examples are real-time control data, and real-time data streams.

Class II represents semi-adaptive connections. For this type of connection the lowest utility (respectively bandwidth) experienced during its lifetime is used for calculating the utility for the whole duration: $\boldsymbol{u_i^a = u_i^{min} \times duration}$. If needed, resources can be decreased (with the respective diminish in utility), but increasing the allocated resources would bring no benefit. Examples are streams of sensor readings with different accuracy, and different types of streaming multi-media.

Class III represents fully-adaptive connections. These are the connections with no real-time requirements, and they can adapt to both increases and decreases of the bandwidth. The accumulated utility is the sum of all the momentary utilities over the total duration: $\boldsymbol{u_i^a = \int_0^{duration} u_i(t)\, dt}$. Examples are fetching e-mail, or different types of file transfer.

Note that the shape of the utility function does not depend at all on the class of the connection. The class is not utilised when bandwidth is initially allocated, but is used to describe the behaviour at subsequent reallocations. The R-U functions describe only the bandwidth-dependence, while the classes describe the time-dependence of the utility.

3.1.2. Drop penalty. Connections may have a specified drop penalty (P_drop_i), which represents the customer dissatisfaction when a connection is disconnected after being admitted into the system. While a rejected new connection brings zero utility, dropping an accepted connection should be penalised. Thus, if P_drop_i is specified, dropping an existing connection will set the final utility of the connection to negative values, $u_i^a := -P\_drop_i$. If the utility is representing the revenue of the network operator, a negative utility implies some form of compensation to the user.

3.1.3. Adaptation time. Even flexible applications (class III) might need a certain amount of time to adapt to the new mode after a reallocation. Providing a highly oscillative resource availability pattern might be worse than keeping a connection at a constant, lower resource level. A predetermined adaptation time is a way to reflect the minimum time between reallocations in order to keep the expected utility.

If the time between two bandwidth reallocations (t_inter_i) is less then a specified adaptation time (t_adapt_i) then the utility generated during this interval ($u_i \times t\_inter_i$) should be lower than under normal circumstances. Therefore we scale it down by multiplying it with t_inter_i/t_adapt_i (the quicker the change, the smaller the generated utility). Described as a new form of penalty this translates to $P\_adapt_i = u_i \times t\_inter_i \times (t\_adapt_i - t\_inter_i)/t\_adapt_i$, to be subtracted from u_i^a. Since classes I and II do not benefit from a bandwidth increase, this penalty is meaningful only for class III connections.

3.2. Dynamic reallocation

Having identified the factors which influence the utility of a connection, we incorporate them in our scheme by modifying the initial R-U functions whenever a reallocation is considered.

Because of the highly dynamic environment, constant reallocation is needed in order to obtain the best results. Thus our bandwidth allocation/reallocation algorithm will be invoked periodically. In the beginning, all connections in the system are new connections, thus no modifications are needed. In the following optimisation cycles we have a mix of new connections and old connections. For the old connections, we have to modify the initial R-U functions.

The modifications will represent the change in the accumulated utility of a connection if a reallocation is performed. Therefore, when bandwidth allocation algorithms, such as the one presented in Section 2.2 are applied on the modified R-U functions, they implicitly take into account the class and age of the connection, the drop penalty, and the adaptation time.

3.2.1. Age dependent modifications. We start by giving an example of a reallocation decision where the initial, unmodified R-U functions are used. Assume there is a class I or II connection $conn_1$, which has an R-U function as the one in Figure 2 (a). Assume the total duration of the connection $duration_1 = 10$ seconds, $elapsed_1 = 5$ seconds have already passed, and the allocated bandwidth during this time was $b_1 = 4$. This means that the accumulated utility so far $tmp\_u_1^a = u_1(b_1) \times elapsed_1 = 3 \times 5 = 15$. At this time a new connection $conn_2$ is competing with the old one for the same bandwidth. Assume $u_2(4) = 5$. Because the basic allocation optimisation algorithm is using the slopes of the segments of the R-U functions to make decisions, and $3/4 < 5/4$, it will choose $conn_2$ versus $conn_1$ and $tmp\_u_1^a$ will be lost. Let's see what is the utility gained by the system after the next 5 seconds: $u^a = u_2(4) \times 5 = 5 \times 5 = 25$. If the first connection had been kept the utility would have been $u^a = tmp\_u_1^a + u_1(4) \times 5 = 15 + 3 \times 5 = 30$, thus the swapping decision is wrong. Therefore, to replace an old connection with a new one, the utility generated by the new connection until the completion time of the old connection should be greater than the utility generated by the old connection during its entire lifetime (see shaded areas in Figure 1). In our example, $conn_1$ should be swapped with $conn_2$ only if $u_2(4) \times 5 > u_1(4) \times 10$.

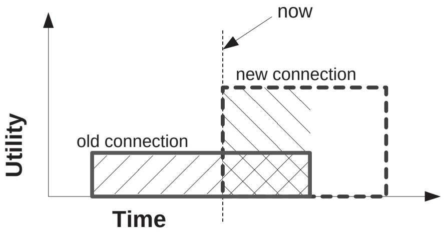

Figure 1. Replacement opportunity

In the above example we assumed that we have the choice only to swap $conn_1$ with $conn_2$. However, a utility function is usually composed of several segments that determine the allocated bandwidth. We further explain how a modified R-U function for a class II connection can be constructed and refer to Figure 2 (a) and (b) as an example.

Each of the segments of the R-U function corresponds to an allocation level l, where $1 \leq l \leq k$, and k is the maximum number of levels. We denote with $b_i(l)$ the bandwidth and with $u_i(b_i(l))$ the utility of level l. For instance, in Figure 2 (a): $b_i(1) = 0$, $u_i(b_i(1)) = 0$, $b_i(2) = 2$, $u_i(b_i(2)) = 1$, etc. We construct the modified utility function, mod_u_i, starting from the actual allocated bandwidth point, $b_i(j)$, where j is the actual level. If the allocated bandwidth stays the same, we keep the same utility value. Furthermore, increasing the bandwidth of a class I or II connection will not increase the accumulated utility of the connection. Thus for all allocation levels l, where $j \leq l \leq k$, we assign:

$$mod\_u_i(b_i(l)) = u_i(b_i(j))$$

However, decreasing the bandwidth results in losing a portion (or all) of the connection's accumulated utility so far (and for which resources have been invested). Thus for all levels l, lower than the actual allocation level, $1 \leq l < j$, we have:

$$lost\_u_i^a(l) = \Big(u_i(b_i(j)) - u_i(b_i(l))\Big) \times elapsed_i$$

$$mod\_u_i(b_i(l)) = u_i(b_i(l)) - \frac{lost\_u_i^a(l)}{duration_i - elapsed_i}$$

Note that for $l < j$ there is a larger difference between two adjacent utility levels in mod_u_i compared to u_i; that is, the slopes of the segments of mod_u_i are steeper. Thus a higher priority will be enforced when considering a bandwidth decrease for the connection. Segments from other connections must have even steeper slopes to be able to take bandwidth from this connection. Note also that the increase in slope is exactly enough so that the lost utility (in case bandwidth is decreased) is recovered during the remaining duration of this connection.

For class I connections, any decrease in bandwidth means the connection is dropped (leads automatically to 0 bandwidth). Otherwise, calculating mod_u_i is similar to class II. The modified R-U function for a class I connection is presented in Figure 2 (c), when starting from an initial R-U as depicted in Figure 2 (a). Since Class III connections are time-insensitive, their initial R-U function is insensitive to age modification.

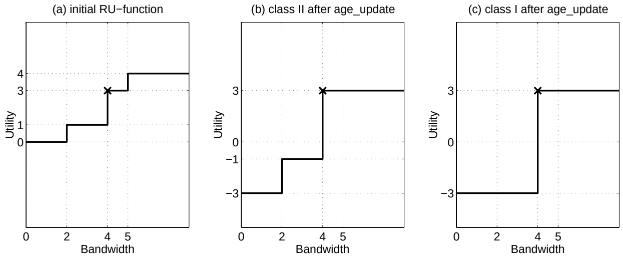

Figure 2. Age modification for class I and II, with $elapsed_i = 5$, $duration_i = 10$, and actual bandwidth $b_i = 4$

Now the question becomes, do we assume that the real duration of every connection is known? Obviously this is too unrealistic to assume. In practice we have to resort to an estimate of a connection's duration. The better the estimation of the connection duration, the more accurate the modification will be. This is because overestimating/underestimating the duration of a connection will underestimate/overestimate the importance of a bandwidth decrease for this connection. In Section 5.4 we further discuss how the system behaves in the absence of an exact knowledge of the duration.

3.2.2. Drop penalty influence. Class I and II connections are dropped (disconnected) whenever their momentary utility becomes zero, since that connection yields no utility in the end. Recall that each connection comes with its own drop penalty, P_drop_i. The effect is computed as follows:

$$mod\_u_i(b_i(1)) = u_i(b_i(1)) - \frac{P\_drop_i}{duration_i - elapsed_i}$$

The drop penalty modifies the R-U function in a similar manner to $lost\_u_i^a$ from the previous section, but it is applied only to the first level (where $b_i(1) = 0$), because if we reduce bandwidth to other levels, the connection is not dropped.

Figure 3 presents the further modification of the class II R-U function from Figure 2 (b) given a drop penalty $P\_drop_i = 8$. Class III connections should not be dropped because of bandwidth shortage, since they can continue at a later time, without penalty.

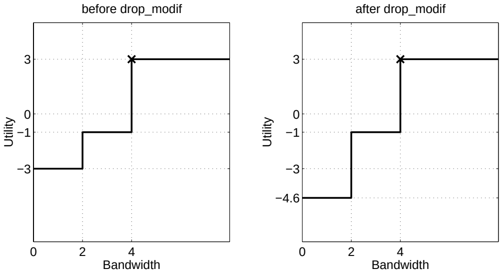

Figure 3. Class II drop modification with $P\_drop_i = 8$, $elapsed_i = 5$, $duration_i = 10$, and actual bandwidth $b_i = 4$

3.2.3. Adaptation time influence. For an adaptive class III connection, if reallocation is performed before the adaptation time since the last allocation has passed ($t\_inter_i < t\_adapt_i$), the gained utility incurs a penalty P_adapt_i, that is calculated as described in Section 3.1.3. If there is an allocation change now, and assuming that the algorithm will keep changing the allocation amount with the same periodicity (that is t_inter_i) for the rest of the connection lifetime, then the modified R-U function is calculated as follows. For all levels l, where $l \neq j$, j being the actual allocation level,

$$mod\_u_i(b_i(l)) = u_i(b_i(l)) - \frac{P\_adapt_i}{t\_inter_i}$$

If no change is made in the allocated bandwidth, then there will be no adaptation penalty. Therefore,

$$mod\_u_i(b_i(j)) = u_i(b_i(j))$$

An example of modifications depending on adaptation time is shown in Figure 4. Since classes I and II do not benefit from a bandwidth increase, and they incur a substantial loss with bandwidth decrease, this penalty is meaningful only for class III connections.

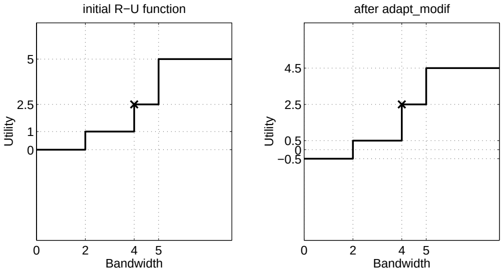

Figure 4. Class III adaptation modification with $t\_adapt_i = 5$, $t\_inter_i = 4$, $P\_adapt_i = 2$, and actual bandwidth $b_i = 4$

3.3. Algorithm overview

To summarise, we present a high-level version of our allocation algorithm. We use u_i to represent the quantised R-U function, $class_i$ is the connection class, $duration_i$ is the expected duration, $elapsed_i$ is the time elapsed since the start of the connection, P_drop_i is the drop penalty, t_adapt_i is the specified adaptation time, t_inter_i is the time since the last change. b_i is the current allocation point, new_b_i the new decision, and $min\_grant\_bw_i$ the lowest bandwidth granted. We also have u^a as system accumulated utility, and $period$ represents the running periodicity of the algorithm. As input the algorithm has all the active connections in the cell (new, old and handoffed).

```
//first deal with duration underestimation:
for $i := 1$ to $n$ do //for all connections
    if $duration_i < elapsed_i + period$ then
        $duration_i := elapsed_i + period$;

//then update the R-U functions
for $i := 1$ to $n$ do
    if $class_i = I$ or $class_i = II$ then
        $u'_i$ := age_modif($u_i, class_i, b_i, duration_i, elapsed_i$);
        $u'_i$ := drop_modif($u'_i, class_i, P\_drop_i, duration_i, elapsed_i$);
    if $class_i = III$ then
        $u'_i$ := adapt_modif($u_i, class_i, b_i, t\_adapt_i, t\_inter_i$);

//apply the convex hull optimisation algorithm to compute the new bandwidth
$(new\_b_1, ..., new\_b_n)$ := convex_hull_opt($u'_1, ..., u'_n$);

//finally the accounting is performed
for $i := 1$ to $n$ do
    if $class_i = I$ or $class_i = II$ then
        if ($class_i = I$ and $new\_b_i \leq b_i$) or
          ($class_i = II$ and $new\_b_i = 0$) then //apply drop penalty
            $u^a_i := -P\_drop_i;$
        else //update accumulated utility so far
            $u^a_i := u_i(min\_grant\_bw_i) \times elapsed_i;$
    if $class_i = III$ then
        $u^a_i := u^a_i + u_i(new\_b_i) \times period$; //update accum. utility so far
        if $t\_inter_i < t\_adapt_i$ then //apply adaptation penalty
            $u^a_i := u^a_i - u_i(b_i) \times t\_inter_i \times (t\_adapt_i - t\_inter_i)/t\_adapt_i;$
        if $new\_b_i \neq b_i$ then // reallocation performed
            $t\_inter_i := 0$ ;
        else $t\_inter_i = t\_inter_i + period$ ;
$u^a := \sum_{i:=1}^{n} u^a_i;$
```

4. Evaluation setup

To evaluate the advantage of using utility-based characteristics of a connection we have compared our scheme, TARA, with a recent adaptive allocation scheme that addresses similar network problems. We begin with a short description of the "Rate Based Borrowing Scheme" proposed by El-Kadi et al. [5]. We then explain how we have reconstructed that algorithm in our simulation environment to make valid comparisons (by ensuring that the choices of parameters were compatible and reproducing their earlier results).

RBBS successfully avoids some rejections by allowing bandwidth to be borrowed from already accepted connections. The borrowable part lies in between a minimum required bandwidth and a maximum desired one as specified by each incoming connection. To provide borrowing fairness, all the connections in the cell lend a proportional share (level) from their borrowable part if needed. Moreover, in order to provide a smooth change in bandwidth allocation, only one share from the borrowable part can be lent at any point in time. Connections are divided in two classes. Class I are real-time connections and are prioritised by reserving a certain amount (e.g. 5%) of bandwidth to be used exclusively by this class during handovers. Class II applications have only best effort requirements. New connections are only accepted if they can be accommodated at the same bandwidth level as the whole cell. Rejecting handovers is more critical, thus class I connections are handed over if their minimum bandwidth requirement is satisfied. Best effort class II connection can be handed over at any bandwidth greater than zero. Whenever enough bandwidth becomes available, connections are replenished and the whole cell moves to a better QoS level.

Table 1. Traffic mix used in the experiments

Applic. Group	Bandwidth Requirement (Kbps)			Connection Duration (sec)			Examples	RBBS class	TARA class	Utility scaling factor
	min	max	avg	min	max	avg				
1	30	30	30	60	600	180	Voice Service & Audio Phone	I	I	1
2	256	256	256	60	1800	300	Video-phone & Video-conference	I	II	1/3
3	1000	6000	3000	300	18000	600	Interact. Multimedia & Video on Demand	I	II	1/10
4	5	20	10	10	120	30	E-Mail, Paging, & Fax	II	III	3
5	64	512	256	30	36000	180	Remote Login & Data on Demand	II	III	1/5
6	1000	10000	5000	30	1200	120	File Transfer & Retrieval Service	II	III	1/7

To get a good comparison of our scheme and the RBBS we have used the same traffic characteristics as those used for evaluation of RBBS [5]. The same traffic mix has been used first by Oliveira et al. [11] and is representative for a future mobile communication network. The first 5 columns of Table 1 are identical with the ones in the RBBS paper. As in their experiments, the maximum required bandwidth and connection duration are not fixed, but follow a geometric distribution with the given minimum, maximum and mean values (columns 2 and 3). The second column from right represents how we mapped the different application groups according to our connection classes.

Since the RBBS is not based on utilities, we had to associate each of the 6 application groups with an R-U function. For example, the shape of the R-U function for application group 3 (the one representing interactive multimedia) is presented in Figure 5. All R-U functions that we used, follow the minimum and maximum bandwidth requirements specified in Table 1, originally taken from the RBBS experiments.

Besides this, the utility of each application group had to reflect their relative importance. For example, even though one might be ready to pay roughly three times more for a video-phone conversation, which has a bandwidth demand of 256 Kbps, the utility per bit is almost three times higher for a audio-phone that requires only 30 Kbps. This information is shown in the rightmost column of Table 1. It represents the utility per bit associated with the maximum required bandwidth (e.g if the maximum required bandwidth of a connection in application group 3 is $4{,}000$ Kbps then the utility for this bandwidth is $4{,}000{,}000 \times 1/10 = 400{,}000$). All the other utility values of the R-U function are calculated following the shape of the function. While assigning utility values is always a subjective problem, we tried to use common practice values in our experiments. Ruben et al. [13] performed a study at Ericsson Cyberlab in Singapore and had access to current conceivable bussiness models.

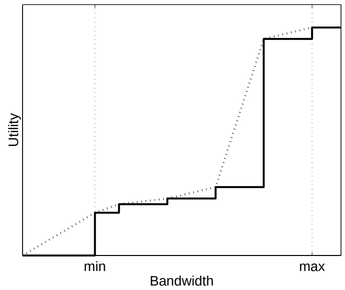

Figure 5. R-U function shape for group 3

Connections arrive on the user equipments (UE) in the network following an exponentially distributed inter-arrival time with a mean of 15 minutes. All the 6 application groups arrive with equal probability. Mobility is modelled in the following way: the time at which a user changes cell follows a geometric distribution starting from 60 sec and mean 300 sec, with equal probability to move in any of the neighbouring cells.

Our simulations were performed in a simulation environment described by Jonasson [7] and built on top of JavaSim, a component-based, simulation environment developed at Ohio State University [14, 6]. We have simulated a hexagon cell-gird of 16 cells, 4×4 , and a go-around world model to preserve uniformity in our gird. Each cell has a capacity of 30 Mbp/s.

For all the schemes the bandwidth allocation/reallocation has been performed with a period of 2 seconds. The drop penalty was set using the following formula $P\_drop_i = 20\% \times u_i^{req} \times avg\_dur$, where u_i^{req} is the maximum required bandwidth, and avg_dur is the average duration (see Table 1). Adaptation time was set to 5 seconds.

As our main performance metric we use the accumulated system utility (u^a) generated by the different connections in the system. The accumulated system utility is independent of the allocation algorithm and is calculated in the same way for all the simulated schemes and according to Section 3.1.

5. Evaluation results

Figure 6 presents the accumulated utility generated by 5 allocation schemes (described shortly) during one simulated hour. On the x-axis we plotted the arrival rate (number of new connections per second). The values in parenthesis represent the corresponding offered load as compared to the capacity of the cell. Thus $0.2(2.56)$ means that the offered load with an arrival rate of 0.2 was 2.56 times the maximum capacity of the cell. The offered load is calculated using the maximum resource requirements of the connections.

For each of the arrival rates and for each bandwidth allocation scheme we conducted five different experiments (by changing the seed of the various distributions) and plotted the average value. The coefficient of variance (CV) was less than 0.06 in almost all of the cases ($CV = \sigma/\mu$, that is the standard deviation divided by the average). A similar statistical confidence applies also to the results presented in the forthcoming figures.

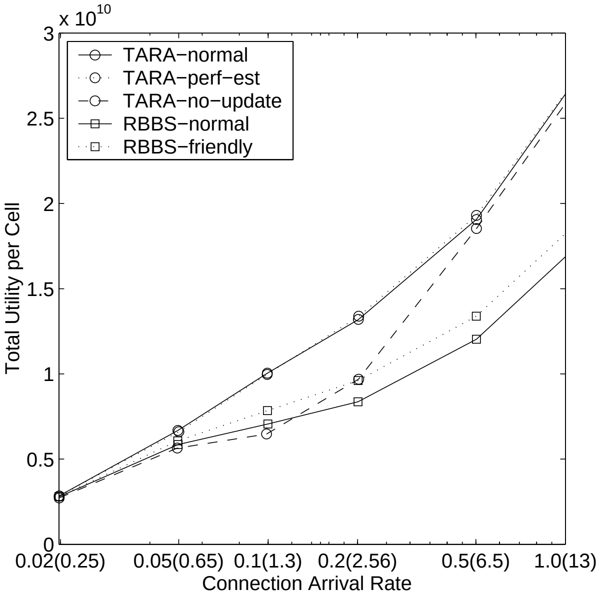

Figure 6. Accumulated utility

5.1. Comparison to basic maximisation algorithm

To see the impact of our class and time aware modifications, we have compared three flavours of TARA. TARA-normal and TARA-perf-estim both use modified R-U functions as presented in Section 3.2. The difference is that for TARA-normal we have used the average connection duration (see Table 1) to estimate the duration of each connection when calculating the modifications (see Section 3.2), while for TARA-perf-estim we used the real duration from the traffic generator. Thus, the latter provides the best possible case to hope for. As depicted in Figure 6, the differences between the curves are marginal and will be further investigated in Section 5.4.

We have also simulated a version of TARA where the modifications of the initial R-U functions are not performed, denoted as TARA-no-update. Basically, TARA-no-update is the convex hull optimisation algorithm (see Section 2.2) invoked periodically. By not taking into consideration the connection classes, the dropping penalty and the adaptation time, TARA-no-update exhibits a 35% decreased system utility when working in areas where the offered load is between 1.3 and 2.6.

At high overloads (corresponding to 0.5 and 1 arrival rate) the applications with the lowest utility per bit, which belong to application group 3, class II, are all rejected at the beginning, and since the lowest utility per bit connections are now application group 6, class III type, which can be put indefinitely on hold, TARA-no-update comes closer to the other two. This is an expected behaviour with a traffic in which the allocation borderline (the last segments bandwidth is allocated to) lies firmly within connection class III.

5.2. Comparison with RBBS

The results for RBBS have been plotted as RBBS-normal. There is a large difference between TARA and RBBS which quickly amounts to 45% when the system gets overloaded. The main factor which contributes to this result is the absence of utility consideration by RBBS. Therefore, while TARA is rejecting only low utility per bit connections, RBBS is rejecting a comparable amount from all application groups.

Besides the original RBBS we also used a slightly modified version of RBBS to make the comparison more favourable towards that scheme (shown as RBBS-friendly). The original RBBS may both lower and raise bandwidth for all connections. Hence, we modified RBBS not to replenish connections of TARA class II (because no utility is gained), and set the borrowable part of TARA class I connections to zero. For both RBBS schemes, reserved bandwidth was $r = 5\%$, number of levels $\lambda = 10$, and borrowing factor $f = 0.5$ [5].

5.3. Choice of performance metric

As the main performance metric, we use the accumulated system utility. Hence, we depart from the traditional call blocking probability (CBP) and call dropping probability (CDP) as performance metrics. We argue that they are obsolete in a system where the required bandwidth of a connection might be only a small fraction of another's demands, but both contribute equally in calculating CBP or CDP. The argument is confirmed by Figure 7, which shows the CBP of the simulations. The application group most blocked by TARA has a big bandwidth demand, and by blocking few of them a lot of bandwidth is saved for other connections. Since RBBS treats all connections equally it has to reject much more connections to equal the number of bits.

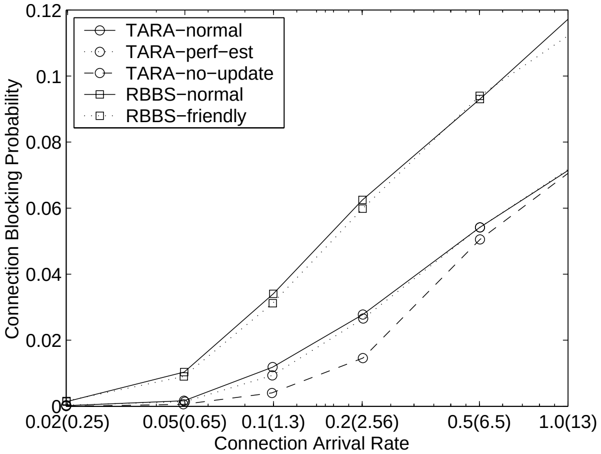

Figure 7. Connection blocking probability

Although the aim of our algorithm is to maximise the utility and not to ensure a low dropping (or blocking) probability, dropping an accepted connection reveals a certain degree of miscalculation. Thus we present the CDP in Figure 8. Since TARA can also drop ongoing connections which are not handed over, we use a different formula for CDP.

$$CDP = \frac{rejectedOngoing + rejectedHandovers}{acceptedNew + acceptedHandovers}$$

Even without reserving a certain amount of bandwidth to be used exclusively for handovers (RBBS reserves 5% for this purpose), TARA-normal and TARA-perf-est are able to keep the number of droppings quite low. Handovers are not regarded as new connections in the cell where they are handed over. Thus, the aging mechanism, the dropping penalty, and the flexibility of class III connections are able to protect handovers as well as other ongoing connections. The consequence of not taking in to consideration these factors is shown in the plot of TARA-no-update. While blocking less connections, it is dropping more than TARA-normal. The effects on the cell utility were already presented in Figure 6.

5.4. Connection duration estimation

Although the age-dependent modifications play an important role in our scheme, the difference between TARA-normal and TARA-perf-est in Figure 6 is marginal. It seems that in most of the cases, the difference between the real duration and the average value, is too small to result in wrong decision (to decisively change the slopes of the R-U functions).

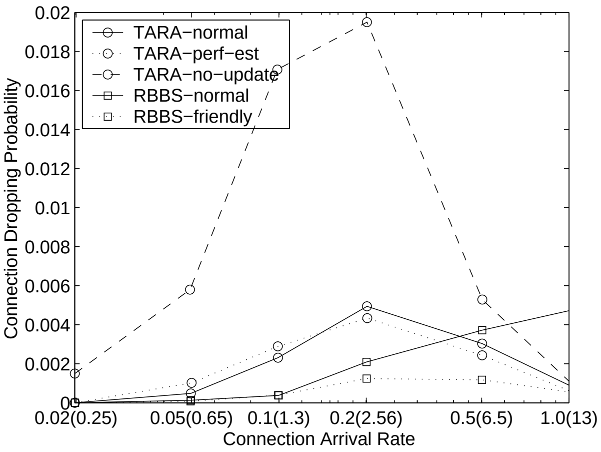

Figure 8. Connection dropping probability

To compare the effects of a traffic mix more sensitive to duration estimation errors, we chose two application groups identical to group 3, but with a relative utility per bit of 0.7 to 1 (see Table 1). The results are plotted in Figure 9. Using the real average (TARA-normal) of the distribution still gives close results to the perfect estimation. They are both contrasted by the TARA-random curve. In the "random" case the estimation value for the duration is chosen randomly between 0 and 10 hours. The results show that if the duration of the connection is approximated by the average duration, the differences in the system utility are quite small compared to a perfect estimation.

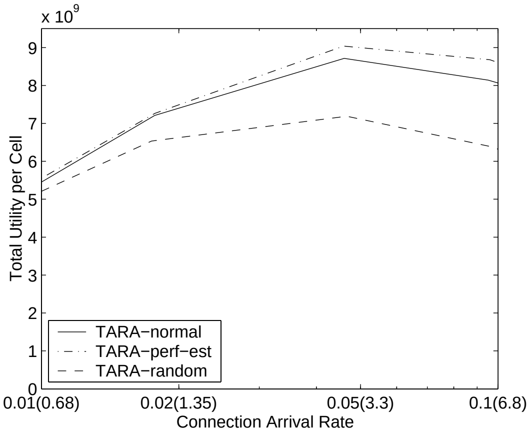

Figure 9. Duration estimation effects

5.5. Overhead considerations

From a computational complexity point of view, the convex hull maximisation algorithm that we use, has a complexity of $O(nL \log n)$, where n is the number of ongoing and new connections, and L is the maximum number of utility levels of an R-U function. The utility function modifications that we introduce have the complexity of at most $O(nL)$, since they have to manipulate each level in the R-U function. The RBBS algorithm has a worst case complexity of $O(n)$, since it has to access each connection when borrowing bandwidth. When borrowing does not occur, that is until free bandwidth is depleted, the algorithm just serves the new incoming connections ($O(1)$).

However, bandwidth reallocations might impose a heavy burden on the system due to executions of control functions and the associated signalling. Since we expect that the reallocation overhead is more important than the computational complexity, we intend to study the tradeoff between optimisation and change overhead as a future work topic.

6. Conclusions

In an open, dynamic system there is a trade-off between optimisation and provisioning. A resource allocation decision might be optimal at a certain timepoint, but as new requests arrive it might become quickly suboptimal. Should we keep the suboptimal allocation or should we reallocate? A reallocation would break the ongoing QoS contract. The novelty of our approach is that we combine both the previous choices in a consistent manner. We synthesise the consequences of reallocation for different types of applications, and use this information while performing a periodic allocation/reallocation optimisation.

More concretely, we have presented an admission control and resource allocation scheme to be used in a future generation mobile network. The scheme is based on an allocation algorithm which maximises system-wide utility, using the utility of each connection specified by a bandwidth dependent utility function. To suit the dynamic nature of the environment, where constant reallocations are required, we identified the effects reallocations have on different connections. Based on their sensitivity to reallocations, connections have been divided into three classes: non-adaptive, semi-adaptive, and fully flexible. These classes react differently to the following identified factors: the age of the connection, a specific drop penalty and the fluctuations in bandwidth allocation. While the application here might seem too specific, we believe that a similar approach can be adopted for other open, dynamic environments (e.g. the link capacity of an Internet provider) or other resource types (e.g. power-aware computing).

To validate our approach, the algorithm has been tested against a baseline that does not take account of the above factors. We have also compared it with a recent adaptive allocation scheme (RBBS), that does not use a value-based approach. Our approach shows significantly increased performance as expressed by the system-wide accrued utility. Another advantage of our scheme is that it provides a consistent treatment of handovers, by taking into account their age-related increased importance when allocating bandwidth.

We conclude by making the following remark. In a future generation mobile network, the bandwidth required by different applications and services will be highly varied, making CBP and CDP obsolete as performance metrics. Instead, the accumulated system utility gives a better performance measurement for such open, dynamic systems.

Acknowledgements

The authors wish to thank Robert Jonasson who implemented the simulator and Mona El-Kadi for providing valuable information about the RBBS scheme.

References

[1] C. Aurrecoechea, A. T. Campbell, and L. Hauw. A survey of qos architectures. *Multimedia Systems Journal, Special Issue on QoS Architecture*, 6(3):138–151, May 1998.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services, rfc 2475. RFC 2475, Dec. 1998.

[3] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview, rfc 1633. RFC 1633, June 1994.

[4] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource reservation protocol (rsvp), rfc 2205. RFC 2205, Sept. 1997.

[5] M. El-Kadi, S. Olariu, and H. Abdel-Wahab. A rate-based borrowing scheme for qos provisioning in multimedia wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(2):156–167, Feb. 2002.

[6] http://javasim.cs.uiuc.edu. Javasim home page.

[7] R. Jonasson. Simulator for resource allocation in future mobile networks. Master's thesis, Linköping University, Oct. 2002.

[8] C. Lee. *On Quality of Service Management*. PhD thesis, Carnegie Mellon University, Aug. 1999. Technical Report CMU-CS-99-165.

[9] C. Lee, J. Lehoczky, R. Rajkumar, and D. Siewiorek. On quality of service optimization with discrete qos options. In *Proceedings of the IEEE Real-time Technology and Applications Symposium*, June 1999.

[10] R. R.-F. Liao and A. T. Campbell. A utility-based approach for quantitative adaptation in wireless packet networks. *Wireless Networks*, 7:541–557, Sept. 2001.

[11] C. Oliveira, J. B. Kim, and T. Suda. An adaptive bandwidth reservation scheme for high-speed multimedia wireless networks. *IEEE Journal on Selected Areas in Communications*, 16:858–878, Aug. 1998.

[12] P. Richardson, L. Sieh, and A. Ganz. Quality of service support for multimedia applications in third generation mobile networks using adaptive scheduling. *Real-Time Systems*, 21(3):269–284, Nov. 2001.

[13] F. Ruben and F. Edlund. Design of a mobile payment intermediary. Master's thesis, Linköping University, LiTH-IDA-Ex-02/X, Dec. 2002.

[14] H.-Y. Tyan and C.-J. Hou. Javasim : A component-based compositional network simulation environment. In *Western Simulation Multiconference - Communication Networks and Distributed System Modeling and Simulation*, June 2001.