A bidding algorithm for optimized utility-based resource allocation in ad hoc networks

Calin Curescu, *Member, IEEE,* and Simin Nadjm-Tehrani, *Member, IEEE*

***Abstract*— This article proposes a scheme for bandwidth allocation in wireless ad hoc networks. The quality of service (QoS) levels for each end-to-end flow are expressed using a resource-utility function, and our algorithms aim to maximize aggregated utility. The shared channel is modeled as a bandwidth resource defined by maximal cliques of mutual interfering links.**

We propose a novel resource allocation algorithm that employs an auction mechanism in which flows are bidding for resources. The bids depend both on the flow's utility function and the intrinsically derived shadow prices. We then combine the admission control scheme with a utility-aware on-demand shortest path routing algorithm where shadow prices are used as a natural distance metric.

As a baseline for evaluation we show that the problem can be formulated as a linear programming (LP) problem. Thus, we can compare the performance of our distributed scheme to the centralized LP solution, registering results very close to the optimum. Next we isolate the performance of price-based routing and show its advantages in hotspot scenarios, and also propose an asynchronous version that is more feasible for ad hoc environments.

Further experimental evaluation compares our scheme with the state-of-the-art derived from Kelly's utility maximization framework and shows that our approach exhibits superior performance for networks with increased mobility or less frequent allocations.

***Index Terms*— Mobile computing, pricing and resource allocation, quality of service, optimization, performance evaluation of algorithms and systems.**

I. Introduction

Mobile ad hoc networks are formed by wireless nodes that move freely and have no fixed infrastructure. Each node in the network may act as a router for other nodes, and flows follow a multi-hop path from source to destination. The infrastructure-less flexibility makes ad hoc networks a strong complement to cellular networks, and ideal for many novel scenarios, such as cooperative information sharing, defence applications, and disaster management. Mobile ad hoc networks will support a wide range of services in which soft real-time (multimedia), and high-priority critical data, seamlessly integrate. As society becomes dependable on the provision of such services, their availability under overloads becomes a critical issue.

In comparison to wireline networks, wireless multi-hop networks will always be more resource constrained due to several fundamental differences. The first major issue is the limited spectrum of the locally shared communication channel. Neighbouring nodes can interfere and cannot transmit independently. The second major difference is the mobility of the nodes and its effect on the established paths. That is, paths are constantly created and destroyed, requiring flow rerouting in the latter case. Network resources such as bandwidth and power have to be dealt with in fundamentally different ways compared to wireline or centralised cellular networks. Resource availability can quickly change, and therefore continuous resource reallocation is needed to provide graceful degradation during overloads, or quality of service (QoS) improvements when more resources become available.

An ad hoc network that is designed for adaptive and autonomic reaction to failures and overloads should take advantage of the flexibility of the service it provides. Best-effort connections are considered to tolerate any changes in their allocation, whereas real-time flows might require a fixed allocation, otherwise the so far accrued utility will be lost. If every service is associated with multiple levels of acceptable quality, the flows in the network can be regularly adapted to achieve optimised QoS. In addressing the above, our scheme supports allocation algorithms that provide differentiation among flows and enforce resource assurance for each flow (subject to system-wide optimisation).

Our approach is based on *utility functions* that capture how the user values the flow's different resource allocation levels. This approach allows for flexible allocations without needing online QoS negotiations. Utility functions provide the means for the network to revise its allocation decisions on-the-fly and optimise resource usage. For instance, choosing an allocation that maximises the aggregated utility of the flows in the network has been shown to be a powerful mechanism for optimising resource allocation instantaneously [1], but also in a time-aware context (i.e. over the age of a given flow) [2].

The contributions of the paper are as follows. We propose and evaluate a combined routing, admission control, and resource allocation scheme that aims to maximise the aggregated utility of the system. As part of this scheme, two novel utility-based algorithms are presented. The core of the scheme is a distributed, QoS-aware, price-based allocation algorithm that allocates bandwidth to flows using only locally available information. A complementary price-based routing algorithm for choosing the most advantageous path for the flows is also proposed.

We start by formulating the allocation problem as a linear programming (LP) maximisation problem. To properly divide the shared channel in an ad hoc setting we use the concept of *clique resource* [3], [4]. It allows gathering mutually interfering links in partially overlapping maximal cliques. The cliques deterministically account for bandwidth capacity and act as resources in the LP problem.

We then propose a distributed low-complexity allocation algorithm that uses the concept of resource *shadow price*, borrowed from the dual LP problem. The novelty is that the algorithm employs an auction mechanism, where flows are bidding for resources. The bids depend both on the flow's utility function and the intrinsically derived shadow prices.

C. Curescu is with Ericsson Research, Torshamnsgatan 23, Kista, 164 83 Stockholm, Sweden. Email: calin.curescu@ericsson.com.

S. Nadjm-Tehrani is with the Department of Computer and Information Science, Linköping University, 58183 Linköping, Sweden. E-mail: simin@ida.liu.se.

We present two versions of the allocation scheme. Focusing on the key concepts, the first version is based on network wide synchronised allocation rounds among clique resources. Besides being hard to achieve, synchronisation also creates periodical bursty control signalling. Thus, we also propose an asynchronous version that shows a utility-based performance similar to the synchronous version.

We study the performance of our scheme at different traffic loads and mobility speeds, and compare our algorithms against two baseline algorithms representing lower and upper performance bounds. The two baselines are: 1) an allocation scheme using hop-based shortest path first routing, followed by maximum possible bandwidth allocation, and 2) an optimal, centralised algorithm that solves the LP form of the problem. The need for global knowledge and a high computational demand makes the LP implementation unsuitable for online allocation. However, it provides an excellent measure of the upper bound on the performance.

Our distributed algorithms are shown to provide an accrued aggregate utility that is very close to the optimum achievable by the centralised LP solution. In our experiments we also isolate the performance gain of using the price-based routing algorithm by comparing it with a hop-based shortest path routing algorithm. The price-based routing algorithm shows its advantage in scenarios with load hotspots.

As a final contribution, we have compared our algorithm to an algorithm proposed by Xue et al. [5], [3], that is representative for a class of state-of-the-art algorithms based on the work by Kelly et al. [5], [3], and which use a distributed gradient projection method (GPA). While both algorithms solve similar allocation problems, they are conceptually different. Our algorithm employs distributed bidding and auctioning, and acts according to an *admission control* paradigm. Xue et al.'s algorithm is a GPA type, and falls into the *congestion control* category. To our knowledge, there is no other class of algorithms that solve a similar allocation problem.

To compare our algorithm to the GPA, we constructed a compatible experimental environment. The comparison focuses on convergence properties and performance measured in terms of accumulated utility for different levels of mobility. The simulation results show that our algorithm adapts better to the changes in a dynamic ad hoc network. Furthermore, our algorithm exhibits far better behaviour when the intervals between allocation points are increased, thus being able to decrease the control signalling overhead with only small decreases in the overall performance.

The paper is organised as follows: Section II discusses related work and Section III presents utility functions, the network model and the LP formulation of the problem. The distributed utility-based allocation scheme is described in Section IV. In Section V we evaluate our allocation scheme in comparison to two other baseline schemes, and also compare its two versions, one with synchronous, the other with asynchronous allocation rounds. Finally, Section VI introduces the gradient projection algorithm and compares it to our allocation algorithm. Section VII concludes the paper and outlines future work.

II. Related work

Work in resource allocation for ad hoc wireless networks has been addressed either at the MAC-level, as an extension to routing, or at an optimisation policy level.

Bandwidth availability in ad hoc networks can be either precomputed [3], [4], [6] or measured at MAC level [3]. Xue and Ganz [6] compute the available bandwidth at a node as the channel bandwidth minus the bandwidth consumed by the traffic at all neighbours. While easy to implement, this is too pessimistic, and better models can be created when interference structures are built based on link interference [3], [4]. In this work, we use the contention model based on maximal cliques of contending links [3]. If no global optimisation is sought, resource allocation can be attempted independently at every node by appropriate MAC layer design. Luo et al. [4] present a packet scheduling approach to ensure a minimum weighted-fair scheduling combined with maximising spatial reuse of channel.

In several earlier works resource allocation/reservation is treated as an extension of the routing protocol. For instance, Chen and Nahrstedt [7] propose an on-demand distributed routing algorithm that aims to avoid flooding the network. They consider delay and bandwidth constrained least cost problems. The feature of the "bandwidth routing" [8] protocol is that link-layer scheduling is directly considered in the protocol. Karaki et al. provide a survey on QoS routing problems [9]. QoS routing is usually not directly aimed at optimal resource allocation but at finding either the shortest path that satisfies at least some minimum QoS requirements, or the path that gives the largest margins for a QoS constraint. In our work, however, the routing algorithm is part of the global allocation optimisation scheme.

A seminal work concerning optimal resource allocation and usage of quantised utility functions is presented by Lee et al [1]. Among others, the authors propose an algorithm that uses the convex hull of the utility functions, and yields good results despite computational simplicity. In our work we adopt the same discrete utility function model. We also build upon a QoS differentiation method proposed in earlier work [10], [2]. By taking into consideration the sensitivity of different application types to resource reallocations, it can consistently treat both real-time and best effort connections.

Several other works describe utility-based approaches to resource allocation in multihop wireless networks [11], [12]. Liao et al [11] provide a utility fair max-min allocation for wireless networks. A distributed allocation scheme is used, and periodical reallocations keep the consistency. We believe, however, that aiming for equal utility can be counterproductive during overloads, as it will degrade all flows to a lowest acceptable level. A system that addresses resource allocation in a wireless/wireline access network is the "TIMELY architecture" proposed by Bharghavan et al [12]. Maximising the revenue based on max-min fairness is one of the criteria used during allocation and adaptation. They employ a 4-tuple revenue model (revenue function, termination credit, adaptation credit and an admission fee), where the same instance of the 4-tuple is used globally. While simplifying allocation, this prevents an accurate differentiation between flows.

During recent years several works have addressed the problem of maximising network utility and have proposed distributed approaches to achieve this [5], [3], [13], [14], [15], [16]. To our knowledge they all derive their solution from a decomposition method presented in the seminal work of Kelly [17] and solved by employing gradient/subgradient projection algorithms. For the reminder of this section we continue discussing characteristics and examples of this class (to which we refer as the GPA class).

Like our approach, these works also use concave utility func-

tions and aim to maximise the aggregated utility of the flows in the network. However, there are some fundamental differences between the two approaches. The GPA class formulation works only with twice differentiable continuous functions while our formulation works with piecewise linear ones. The expressiveness of the latter is important to us as we aim to capture the real user-perceived utility of the flows, while in the GPA class the utility functions are used to enforce a certain rate-fairness criteria.

In our case we allocate the constrained resource in explicit allocation rounds, with flows admitted according to the size of their bid (following an admission control paradigm). The gradient/subgradient projection algorithm on the other hand reacts based on the congestion level of the resource and moves step-wise in the direction of the gradient (following a congestion control paradigm). Thus, if the step-size is large, the allocation will overshoot the optimum and may lead to oscillative behaviour. If the step-size is small the algorithm will converge but many allocation iterations are needed to reach the equilibrium. While this works for flexible flows, it is unacceptable for inflexible flows needing a guaranteed resource level once it is allocated. In addition, networks with a high grade of mobility and irregular flow arrival rates could spend little time in an optimal state, and flows would suffer frequent oscillations in their allocation. In our scheme we attempt to allocate close to the optimum in one try, and (re)allocation is considered only to account for sizeable changes in the network state. Note also that in our case the allocation can change only to clearly defined resource amounts (specified by the pieces of the utility function) and we take into consideration how/if reallocation affects the utility that has already been accrued for the flow.

The work by Xue, Li and Nahrstedt [5], is an example of the GPA class that addresses a formulation that is very similar to ours. Both works use shadow prices of the bandwidth clique resources on the end-to-end path of the flow for steering allocation. They use an iterative algorithm where a) the network adapts to the rate of flows by changing the resource price, b) the flow adapts to the new price by modifying the transmission rate. In a more recent work [3], Xue et al. consider a mobile environment and use AODV as a routing algorithm. AODV routes over the shortest path (in number of hops) and this might overload inner network paths while resources will go unused towards the marginal areas. Our work uses a price-based approach even for routing decisions. A further description of the Xue et al. work together with a comparative study is presented in Section VI.

Other works attempt to directly include the MAC scheduling in the optimisation problem. Eryilmaz and Srikant propose a solution based on a queue-length centralized scheduler and a primal-dual distributed congestion controller [13]. Another solution, where the MAC scheduling is also distributed does rate allocation by changing the transmission persistence probability of links and nodes. However, due to the MAC inclusion only utility functions with enough curvature can be used [14]. An overview of cross-layer optimisation in wireless networks is provided in the tutorial by Lin et al. [18]. In our work we use clique resources, which allow us to decouple our allocation optimisation problem from how MAC scheduling is performed. The disadvantage is that the number of clique resources a link traverses can be considerably higher compared to the number of traversed links, and that a more conservative estimation of capacity is needed (due to possible scheduling interdependencies).

In the context of wireline networks, Lin and Shroff [15] explicitly include the QoS routing part in their utility optimisation problem by allowing a flow to use multiple paths and optimising the routing probability over the different paths. By including flow arrival and service time probabilities the authors optimise for average network conditions thus decreasing the frequency of the control loops. Multipath routing optimisation is also addressed by Han et al.[16] where it includes stability conditions for a generic multi-path TCP implementation. In our work we do not include routing in the optimisation loop. However, we use the results of the allocation optimisation by choosing the route which will use the least congested path. Thus, we expect multipath routing to provide an advantage only for very large long-lived flows. The opportunity for multi-path routing needs also to be traded off against the overhead of maintaining the multiple paths.

A preliminary version of our distributed allocation scheme and a comparison with Xue et al. algorithm have appeared in conference proceedings [19], [20].

III. Problem formulation

In this section we first outline our utility and network model, followed by the LP formulation of the allocation problem. We then present the notion of shadow prices and highlight some properties of the optimal solution that will be used as a guideline when constructing the distributed algorithm.

A. Utility functions

Many types of mobile applications support different QoS levels. For example, multimedia services can decrease audio or picture quality to meet some bandwidth or delay restrictions, while applications like e-mail or file sharing can usually adapt to anything available. The changes in application utility depend on the amount of allocated resource, and can be captured by an associated utility function. By using utility functions in the allocation process a clear quantitative differentiation can be made among competing applications. Thus, the system can optimise QoS by lowering the allocation for the least efficient applications during overload periods, and increasing the allocation of the most efficient ones when resources become available. Moreover, online negotiations are not needed as they are intrinsically built in the utility functions.

In our work we employ a user-centric utility view. Utility functions do not act only as internal parameters for the system policy, but also reflect the “contract” between the user and the service provider. Graphical tools with built-in examples could help the user easily construct such utility functions [21]. As a starting point these tools could suggest to the user values taken from quality assessment studies [22], such as evaluations of video codecs [23]. Note that the unit used for measuring utility is not important, as long as we use the same unit globally for all flows and for all resource prices. A straightforward way to use this utility model in a commercial system is by directly linking the utility of a certain service level to the price the user is ready to pay. For instance, if a user prefers a fixed price rate, a simple, one-step utility function can be used.

Let us denote with $u_i(b_i)$ the momentary utility accrued for an allocated bandwidth of b_i. Furthermore, for ease of representation, and to keep complexity low, we use quantised utility functions similar to the ones used in the QRAM project [1]. Thus, we

can represent the function as a short list of bandwidth-utility pairs, $u_i = \left(\binom{U_i^1}{B_i^1}, ..., \binom{U_i^s}{B_i^s} \right)$, where s is the number of utility levels of flow i, and $u_i(b_i) = U_i^k$ is the accrued utility for an allocated resource $b_i \in [B_i^k, B_i^{k+1})$, where $1 \leq k \leq s$. Utility functions could take any shape, which makes the optimal allocation problem NP-complete even for a single resource case. Nevertheless Lee et al [21] obtained results very close to the optimum when approximating general utility functions with their convex hull frontier. We too use convex hull frontier approximation and our simulations presented in Section V-G confirm that the imperfection in allocation is small even in our setting. The utility functions for the different application types used in our experiments can be found in Section V-A, Figure 6.

Finally we would like to emphasise that utility functions do not inherently reflect application flexibility with respect to resource reallocations. An application that can function at several resource levels may be nevertheless quite sensitive to changes once an initial resource level is chosen and allocated to it. Our system can treat rigid applications (e.g. real-time) differently from flexible applications (e.g. file transfer). This is done by changing the user-provided utility functions at run-time for allocation purposes. For example, a real-time flow's utility function will increase with age, expressing the importance of not losing invested resources. In earlier work we have illustrated the benefits of this differentiation mechanism in a cellular setting [10], [2].

B. Network model

We consider a wireless ad hoc network with n nodes. Two nodes that are in transmission range of each other are regarded as connected by a wireless link. Nodes communicate with each other by means of multi-hop bidirectional end-to-end flows, f_i, between an originator (source) node and a destination node.

In ad hoc wireless networks, we have a location-dependent contention between the transmissions on the wireless links. Transmissions over a link can be bidirectional, thus two links contend with each other if one of the end-nodes of a link is within the transmission range of an end-node of the other link [5], [4]. A link contention graph can be constructed, where vertices represent links, and an edge connects two vertices if the corresponding links contend with each other. Each maximal clique in such a graph represents a distinct maximal set of mutually contending links[1]. A necessary condition for a *feasible* bandwidth allocation is that for each maximal clique the bandwidth allocated over all links forming the clique is less than or equal to the maximum channel capacity. In practice, the choice of transmission scheduling algorithm and additional interference can impose a tighter bound than the channel capacity. That is,

$$\forall j, \quad \sum_{l \in r_j} b^l \leq B_j^{max} \tag{1}$$

where b^l is the allocated bandwidth over wireless link l, and B_j^{max} is the achievable capacity of the maximal clique r_j. Note that B_j^{max} is less than or equal to the wireless channel capacity.

Hence, each maximal clique can be regarded as an independent *clique resource* with capacity B_j^{max}. Since only links close to each other contend for the same bandwidth, local information is sufficient for constructing the cliques that a certain link belongs to[2]. In constructing the cliques we use a similar approach to the one used by Xue et al [5], [3]. Section IV-D details how cliques are constructed and on which nodes the clique-related activities take place.

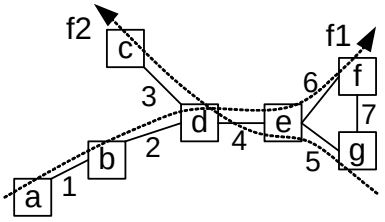


Fig. 1. Network example

In Figure 1 we present an example of a network topology (the mobile nodes are represented as squares) and two ongoing flows using this network. Figure 2 presents the link contention graph, where vertices represent the links (identified by corresponding numbers) of the network in Figure 1. We can identify three maximal cliques representing resources. Note that a single flow can span over several links belonging to the same clique resource.

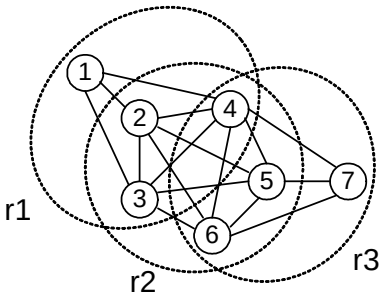


Fig. 2. Link contention graph for network example

Let q_{ij} represent how many links of clique r_j are used by flow f_i. Transmissions on the links belonging to a clique are mutually exclusive, which means that the bandwidth used up by a flow is q_{ij} times the flow's transmission rate. Let m be the total number of flows, and b_i a certain allocation to flow f_i. Then we can rewrite the constraint in Equation 1 in relation to the bandwidth allocated to the m flows in the network:

$$\forall j \quad \sum_{i=1}^{m} q_{ij} \times b_i \leq B_j^{max} \tag{2}$$

Note that when a flow does not traverse a clique we have $q_{ij} = 0$. Table I presents the values of q_{ij} for the example in Figures 1 and 2.

TABLE I
FLOW-RESOURCE USAGE FOR NETWORK EXAMPLE

q_{ij}	r_1	r_2	r_3
f_1	3	3	2
f_2	2	3	2

[1]A maximal clique is a subset of vertices, of which each pair defines an edge, that cannot be enlarged by adding any additional vertex.

[2]We assume that the communication range is the same as the transmission range. Otherwise, bandwidth estimation has to be used, since two nodes could interfere but not be able to communicate.

C. The optimisation problem and its linear programming form

For the wireless multi-hop network, having computed all the clique resources, and assuming that routing has already been done, at any allocation moment we can formulate the following allocation problem. Let u_i be the utility function and x_i the allocation to be determined for flow i (we use x_i to represent the allocation to be found, while b_i represents an allocation example). Let p be the number of clique resources and q_{ij} the usage count of clique resource j by flow i. Then the optimal allocation for all x_i over all cliques j can be obtained from:

$$Maximise \quad \sum_{i=1}^{m} u_i(x_i) \tag{3}$$

$$subject\ to \quad \sum_{i=1}^{m} q_{ij} \times x_i \leq B_j^{max} \tag{4}$$

$$0 \leq x_i \leq b_i^{max} \tag{5}$$

where B_j^{max} is the maximum bandwidth available for clique j, and b_i^{max} is the maximum bandwidth required by flow i.

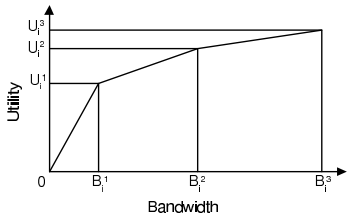


Fig. 3. Linear segments of a convex hull

To make the previous general problem solvable in polynomial time, we first approximate the original utility functions, u_i, with their convex hull frontiers, u_i' which are piece-wise linear and concave. To completely linearise the objective function we conceptually split a flow in several parallel subflows (same source, destination, and path), each corresponding to a linear segment of the utility function. For a subflow k of flow i the allocation is constrained through the utility function levels B_i^k as follows, $b_i^k \leq B_i^k - B_i^{k-1}$. The *utility efficiency* of the subflow (utility/bit) is $\lambda_i^k = \frac{U_i^k - U_i^{k-1}}{B_i^k - B_i^{k-1}}$. In Figure 3 we have an example of a convex hull with 3 linear segments corresponding to 3 subflows. Then, given s segments in the convex hull, for allocations b_i^k to subflows of flow i, we have:

$$u_i'(b_i) = \sum_{k=1}^{s} \lambda_i^k \times b_i^k \tag{6}$$

where $b_i = \sum_{k=1}^{s} b_i^k$. However, not every allocation to the subflows is consistent. In order to use the right side of Equation 6 as a function, the following two constraints must be satisfied:

(C_1) Every k-th subflow has a maximum allocation limit. That is, $b_i^k \leq b_i^{k\ max}$ where $b_i^{k\ max} = B_i^k - B_i^{k-1}$.

(C_2) The order of the segments in the R-U function must be respected when allocating (i.e. "gaps" are not allowed). That is, if $b_i^k > 0$ then for all $l < k$, $b_i^l = b_i^{l\ max}$.

Given that the constraints above are satisfied, we can present the following LP form of the problem. The subflow allocation variables are denoted by x_i^k.

$$Maximise \quad \sum_{i=1,k=1}^{m,s} \lambda_i^k \times x_i^k \tag{7}$$

$$subject\ to \quad \sum_{i=1,k=1}^{m,s} q_{ij} \times x_i^k \leq B_j^{max} \tag{8}$$

$$0 \leq x_i^k \leq b_i^{k\ max} \tag{9}$$

Constraint (C_1) is linear and directly used in the above form. Moreover, an optimal allocation automatically respects constraint (C_2), as the linear segments of the convex hull are ordered highest efficiency first. This is proven by the following lemma.

Lemma 3.1: The results of the maximisation problem in Equations (7) to (9) satisfy constraint (C_2).

Proof: Let's assume the opposite, which means that there are two subflows, l and k, $l < k$, of a flow i where $b_i^l < b_i^{l\ max}$ and $b_i^k > 0$. Let $\gamma = min(b_i^{l\ max} - b_i^l, b_i^k)$. We denote the utility generated by subflows l and k by $U_{l+k} = \lambda_i^l \times b_i^l + \lambda_i^k \times b_i^k$. Since both subflows belong to the same flow, one can imagine subtracting γ from subflow k and allocating it to subflow l. Let $U'_{l+k} = \lambda_i^l \times (b_i^l + \gamma) + \lambda_i^k \times (b_i^k - \gamma)$. Then $U'_{l+k} > U_{l+k}$ because $\lambda_i^l > \lambda_i^k$ for a concave function. Other allocations being equal, this means that b_i^l, b_i^k are not optimal. Contradiction. ■

Obviously, a centralised algorithm that implements the LP solver requires large computational and signalling overheads, making it infeasible for online allocation in an open and dynamic ad hoc network. However, we use it as an upper bound baseline to evaluate the performance of the distributed, low complexity algorithm we propose in Section IV.

D. Worst case error introduced by convex hull approximation

Next we give a theoretical worst case difference between the optimal solution to the original maximisation problem (Equations (3) to (5)), and the solution given by the above LP formulation (Equations (7) to (9)) that uses convex hull approximations when deciding allocation. Note that even though we use the convex hull approximation to reach an allocation decision, say b_i, the utility accounting is done using the original utility functions: $U = \sum_i u_i(b_i) = \sum_i u_i(\sum_k b_i^k)$.

Let U_{opt} denote the optimal attainable utility. Obviously $U \leq U_{opt}$. Let δ_i denote the maximum difference between the convex hull of a utility function and the utility function itself, for any of the flows involved. For instance in Figure 6(a) the maximum difference between the convex hull (the thick dashed line) and the utility function (the continuous line) is just before the 0.5 bandwidth point. Let ξ be the largest δ_i among all the utility functions of the involved flows, $\xi = max_{i=1}^{m} \delta_i$. Let b_i^k be an allocation as obtained by solving the LP problem of Equations (7) to (9), and let p be the number of clique resources. Then we can state:

Lemma 3.2: $U_{opt} - p \times \xi \leq U \leq U_{opt}$.

Proof: First we define an additional variable, an upper bound utility, U_{sup}. It denotes the optimal value for the objective function in Equations (7) to (9), i.e. the utility that would accrue if the convex hulls were the real utility functions, $U_{sup} = u_i'(b_i) = \sum_{k=1}^{s} \lambda_i^k \times b_i^k$. Since $u_i' > u_i$ for all allocation instances, $U_{opt} \leq U_{sup}$. Now, let's see how large the difference between U and U_{opt} can become. Regarding the allocation of subflows, b_i^k, if $b_i^k = 0$ or $b_i^k = b_i^{k\ max}$ the subflow has the same contribution to

U and U_{sup}. So the difference between U and U_{sup} is given by the number of subflows with partial allocation, i.e. where $0 < b_i^k < b_i^{k\,max}$. These partial allocations can result only due to the constraints in Equation (8). Due to Lemma 3.1, for each of the p constraints in Equation (8), at most one segment might result in a partial allocation. Thus $U_{sup} - U \leq p \times \xi$, which leads to $U_{opt} - p \times \xi \leq U \leq U_{opt}$. ■

Note that both U and U_{sup} are easily computable at runtime, and since $U \leq U_{opt} \leq U_{sup}$ we can compute at runtime a "better" worst case bound for a particular system instance. In Section V-G, Table V we present such a worst case runtime result.

E. Dual formulation and characteristics of optimal solution

In the distributed allocation algorithm we use the concept of *shadow prices*. In order to introduce shadow prices, we next present a dual LP formulation together with some useful characteristics of the optimal solution.

The following problem is the dual of the LP problem presented in Equations (7) to (9). It aims to put a price on the resources (i.e y_j and v_i^k) by solving:

$$Minimise \sum_{j=1}^{p} B_j^{max} \times y_j + \sum_{i=1,k=1}^{m,s} b_i^{k\,max} \times v_i^k \tag{10}$$

$$subject\ to \sum_{j=1}^{p} q_{ij} \times y_j + v_i^k \geq \lambda_i^k \tag{11}$$

$$0 \leq y_j,\ 0 \leq v_i^k \tag{12}$$

The shadow price y_j of a resource j, shows the marginal increase of total utility if the resource amount could be increased with one more unit. Conversely it can be interpreted as the marginal decrease in utility (per resource unit) if less resource was available. We can imagine a new subflow requesting a marginal amount of bandwidth along its end-to-end path. In order to make space for the new flow, the system has to ensure a marginal amount of available bandwidth on all the resources the new flow traverses (potentially rejecting other subflows). According to the above definition of shadow prices, the utility/bit lost by making room for the new flow is $pp_i = \sum_j q_{ij} \times y_j$, where pp_i stands for *path-price* of flow i. The utility/bit gained by accepting the new subflow is given by λ_i^k. Intuitively, in order to maximise utility, the network should only accept subflows where $\lambda_i^k > pp_i$, otherwise the total utility of the system will decrease. Note that both λ_i and y_j have the same unit (utility/bit), and that both parameters measure utility, albeit from different perspectives. The first one directly measures the utility of the flow, while the second one measures the utility of the clique resource, generated as a consequence of the flows contending for it.

For a complete definition of the dual problem, we need to model the fact that each subflow is restricted to a maximum allocation of $b_i^{k\,max}$. Thus we introduce artificial resources that are exclusively used by a corresponding subflow, with the only purpose to limit allocation to $b_i^{k\,max}$. The shadow price of these resources is denoted by v_i^k.

Here we present some important characteristics of solutions to the primal and dual LP problems, characteristics that give us a better insight into the allocation problem and help us devise the distributed allocation algorithm. Out of constraints (8) and (11) we can derive the non-negative slack variables w_j and z_i^k. The first, w_j, represents the amount of unused capacity at clique resource j. The second, z_i^k, represents "loss per bit" for subflow f_i^k ($z_i^k = pp_i + v_i^k - \lambda_i^k$, i.e. the difference between the cost of resources on the path and the utility yield of the subflow). Using the slack variables, the inequalities of the primal and dual problem turn into equalities:

$$\sum_{i=1,k=1}^{m,s} q_{ij} \times x_i^k + w_j = B_j^{max} \tag{13}$$

$$\sum_{j=1}^{p} q_{ij} \times y_j + v_i^k - z_i^k = \lambda_i^k \tag{14}$$

According to LP theory [24], the optimal solutions for the primal and dual problems fulfil the following constraints on allocations and resource prices. Constraint (17) is similar to (16), but applies to the above mentioned maximum subflow bandwidth resources.

$$x_i^k \times z_i^k = 0 \tag{15}$$

$$y_j \times w_j = 0 \tag{16}$$

$$v_i^k \times (b_i^{k\,max} - x_i^k) = 0 \tag{17}$$

From Equations 13-17 we can identify the following characteristics of the optimal solution:

- (O_1) Either a resource is underutilised ($w_j > 0$) and then its shadow price is zero ($y_j = 0$), or it is fully contended (with some subflows receiving no allocation) and its price is greater than zero.
- (O_2) For subflows where $z_i^k > 0$, we have $x_i^k = 0$. Note also that since $x_i^k = 0$, we have also $v_i^k = 0$. This is in accordance to $\sum_j q_{ij} \times y_j > \lambda_i^k$, meaning that the path-price (accumulated price of resources along the path) is higher than the subflow utility efficiency, and the subflow is not profitable enough to get an allocation.
- (O_3) For subflows where $z_i^k = 0$, and $v_i^k = 0$, we have $\sum_j q_{ij} \times y_j = \lambda_i^k$. This means that the subflow is at the allocation edge given the resources it uses.
- (O_4) For subflows where $v_i^k > 0$, we have $x_i^k = b_i^{k\,max}$, that in turn implies $z_i^k = 0$. This is in accordance to $\sum_j q_{ij} \times y_j + v_i^k = \lambda_i^k$. Thus, v_i^k represents a "pricing slack", i.e. the amount by which the accumulated prices of the used resources could increase, and the flow would still be profitable.

Note that in the last three cases λ_i^k plays the role of a "pricing budget". As long as the subflow is able to "pay" for all the used resources (by the path-price, pp_i) without overshooting the budget, it is accepted, otherwise it is rejected. For a practical interpretation of λ_i and pp_i as part of a user-to-provider payment mechanism both the following modes are supported: 1) For every admitted flow, the user pays the actual path price, pp_i. The budget, λ_i, signals to the provider the maximum amount the user is willing to pay. In this mode, the optimisation scheme assures the user that the price he is paying is the minimum price given the competition he is facing. To ensure profit for the provider even when the network is underloaded, a minimum price/bit should be demanded from the users. 2) The user always pays the full budget, λ_i, as specified by the utility function. In this mode, the optimisation scheme will ensure that higher paying customers are always accepted first, which also means profit maximisation for the provider.

IV. Distributed resource (re)allocation

The ad hoc network considered in this work is an open dynamic system where resource request and availability are always changing. Therefore our scheme employs periodic reallocations to keep the resource usage optimised. As end-to-end connections span several nodes and clique resources, it is important that (re)allocations are well coordinated along the path. Moreover, reallocations imply a "mode" change for applications so their number should be strictly controlled. In this section we present an algorithm that uses allocation rounds that are synchronised for all clique resources. The use of periodic, synchronised allocation rounds guarantees that flows will enjoy a fixed allocation for at least one period. It also puts a bound on the reallocation rate in the system, even if the rate of events (traffic and topology changes) is much higher. Later, in Section IV-F, we propose a new version of the algorithm that works also when the allocation rounds are not synchronised among the clique resources. Choosing an appropriate period size implies a tradeoff. The shorter the period, the better the system is at keeping the utility optimised, but the larger the computation and signalling overhead.

We will now proceed to describe the synchronised version of the allocation algorithm that will be referred as *adhoc-TARA* henceforth. Assume for now that a route for a flow is already established (we will come back to how this route is found in Section IV-E). Conceptually, at each period the (re)allocation will proceed like this:

- Every flow calculates a bid for all clique resources it traverses, based on their associated shadow prices.
- Each clique resource independently evaluates the bids, proposes a certain bandwidth allocation to the flow and recalculates its shadow price.
- The flow chooses the lowest bandwidth proposal from all the cliques it traverses as the new bandwidth for the new period.

A. Bid construction

As already mentioned, the utility efficiency, λ_i^k, represents the maximum "budget" available for "paying" for the traversed resources. Note that a subflow needs to be accepted at all the traversed resources in order to be established. We assume that the contention level of a resource will not abruptly change from one period to the next, so we start with a preliminary bid equal to the shadow price of the resource in the previous period. Now, if we add all these preliminary bids, we end up with the path price of the previous period, $pp_i = \sum_j q_{ij} \times y_j$. Then, if we subtract the path price from the budget, we can compute a *price slack*, $slk_i^k = \lambda_i^k - pp_i$. So, how should this slack be included in the bids? As we do not make any assumptions on the evolution of the resources' congestion, we divide the slack uniformly among the used resources. The number of resources used by a flow is given by the clique-counter, $cc_i = \sum_j q_{ij}$.

Thus, the bids are created as follows:

$$bid_{ij}^k = y_j + \frac{\lambda_i^k - pp_i}{cc_i} = y_j + \frac{\lambda_i^k - \sum_j q_{ij} \times y_j}{\sum_j q_{ij}} \qquad (18)$$

where bid_{ij}^k is the bid of subflow $\frac{k}{i}$ for resource j. The sum of a subflow's bids always amounts to its maximum budget, λ_i^k. As a simple example, imagine a subflow, f_1^1 with budget $\lambda_1^1 = 10$, that uses three clique resources with the shadow prices (of the previous allocation period) $y_1 = 2$, $y_2 = 2$, $y_3 = 3$. The slack $slk_i^k = 10 - 2 - 2 - 3 = 3$ is divided uniformly for each resource. Thus the new bids are: $bid_{11}^1 = 3$, $bid_{12}^1 = 3$, $bid_{13}^1 = 4$.

During allocation, if all bids are large enough, the subflow is accepted, and corresponds to either category (O_3) or (O_4) in Section III-E.

B. Independent allocation

After all the bids have been placed, every clique resource independently allocates the bandwidth to the subflows in decreasing order of bids until bandwidth is depleted. Then the *new shadow price* of the resource is set to the price of the lowest bid among the accepted subflows. Note that all the bandwidth is reallocated, and some subflows might get this time an allocation different from last period.

If contention at a certain resource is greater than during the previous allocation, its price will increase. If some subflow's bid cannot accommodate this increase, the subflow will be rejected. If contention decreases, the price of a resource will decrease. This means that some subflows that bid less than the previous shadow price (i.e. have a negative price slack) are accepted, bringing the price down accordingly. If a resource does not allocate all its bandwidth, it is underloaded and its shadow price becomes zero (case (O_1) in Section III-E).

C. Discussion

Note that if the real shadow prices (the solutions of the dual problem) were known, perfect bids could be constructed. In such a case, a subflow would consistently be accepted or rejected at all the cliques that it traverses. As we do not know the new shadow price beforehand, we use the shadow price from the last allocation as an estimate. At a certain clique resource, the new price could become higher than the bids of some flows (i.e the new contention level was underestimated at bid construction). In such a case, some flows that with hindsight could have offered a proper bid are rejected. Conversely, overestimating a resource price unnecessarily increases its bid to the detriment of bids for other resources.

As a consequence of over/underestimation, the allocated bandwidth could be different at different clique resources, and the flow can use only the minimum allocation over the end-to-end path. Hence, one could use several consecutive iterations of the algorithm to better balance the bids. Nevertheless, as the algorithm is intended for online allocation we use only one iteration per reallocation period, and any mis-allocated bandwidth will remain unused for that period. Since during an optimal allocation the amount of this mis-allocated bandwidth is zero, in our experiments we use the mis-allocated bandwidth as another measure of how close to optimal allocation the performance of our algorithm is.

Figure 4 presents a pseudo-code of the two parts of the distributed algorithm that run synchronously at every clique resource and at every node respectively. For every clique resource a clique-leader node, which is used to perform the (re)allocation computations, is determined at clique-construction time (see Section IV-D).

The clique-leader gathers information about the flows using the clique resource and then runs the allocation algorithm. Whenever a flow starts/stops using a wireless link, the link's end-node closer to the clique-leader registers/deregisters the flow with the clique-leader. This applies to all cliques containing the given link. The

Allocation algorithm run at every clique-leader j, at every period τ:

Let F_j be the set of flows using resource j
$avb_j = B_j^{max}$ //initialise available bandwidth
$\forall$ subflows $f_i^k \in F_j$
  $bid_{ij}^k = y_j + (\lambda_i^k - pp_i)/cc_i$ //compute bid
while $F_j \neq \emptyset$ //allocate for highest bidder first:
  select $f_i^k \in F_j$ with highest bid
  if $avb_j > q_{ij}^k \times b_i^{k\,max}$
    $x_{ij}^k = b_i^{k\,max}$
    $avb_j = avb_j - q_{ij}^k \times x_{ij}^k$
  else
    $x_{ij}^k = 0$
  $F_j = F_j - f_i^k$
$y_j = min(bid_{ij}^k \mid x_{ij}^k > 0)$ //recompute resource price
$\forall i$ where $f_i \in F_j$
  $x_{ij} = \sum_k x_{ij}^k$
  send x_{ij} and price y_j to source of f_i

Flow adaptation algorithm run at every node n, at every period τ:

$\forall$ flows f_i sourced at node n
  $\forall$ resources j that f_i traverses
    wait for allocation x_{ij} and price y_j
  $x_i = min(x_{ij})$ //set bandwidth of f_i
  $pp_i = \sum_j q_{ij} \times y_j$ //recompute its path price
  $\forall$ resources j that f_i traverses
    send pp_i to clique-leader of resource j

Fig. 4. The distributed allocation algorithm

clique-leader can be chosen such that the distance to the end-nodes of the links belonging to the clique is at most 2 hops away. Therefore, inside-clique signalling could use the MAC layer signalling (e.g. piggyback RTS, CTS, ACK packets).

The natural place for running the flow-adaptation part of the algorithm (i.e. adjusting transmission rate to the new allocation) is at the flow's source node. Note also that the signalling information between clique resources and the source nodes of the flows is sent only along established routes, and thus can be piggybacked on existing packets. Packets belonging to any flow using a link in the clique will pass through a node that is at most 2 hops away from the clique-leader (with which it communicates using in-clique signalling).

The flow's source node must receive the new bandwidth decision from all the clique resources on the end-to-end path of the flow and choose the minimum allocated. The larger the synchronisation error between the clocks of the clique-leaders, the more the source-node has to wait until it can set the new rate of the flow. Thus a flow could set an increased rate some time before another flow finishes its allocation round and decreases its rate, leading to short-lived congestions at certain points. Regarding clock-synchronisation protocols in wireless (sensor) ad hoc networks, Römer et al. [25] give precision results of less than $100\mu sec$ for nodes five hops away. These clock skews are small compared to the envisioned reallocation periods, and thus we assume these congestions to be easily mitigated. Nevertheless, synchronised allocation generates bursty signalling in the network, and in Section IV-F we present modifications to the allocation algorithm to study the potential for asynchronous allocation rounds among the clique resources.

Finally we would like to point out that in this work we assume a collaborative approach. Nevertheless, one direction for accommodating selfish nodes could be to reward the nodes of a clique proportionally to the advertised clique-price and the quantity of the flows traversing the clique, combined with using signature mechanisms when gathering statistics. Thus falsely decreasing the clique prices would decrease the revenue of the clique. Falsely increasing the clique prices would result in the clique being avoided and underutilized (due to flows routing on other, lower-priced paths), which will also result in diminished returns.

D. Mobility and clique construction

Due to mobility, a node might enter or exit the communication range of another one, thus creating a new wireless link, or alternatively breaking one. Discovery of topology changes can be implemented either event-based (using MAC feedback) or periodically (local broadcast of hello messages).

As mentioned previously, only local information is needed to construct the maximal cliques. We know that only links adjacent to nodes that are at most 3 hops away might contend with each other (see Section III-B). Thus, if all nodes send their neighbourhood list 3 hops away, every node will be able to identify all the cliques containing any adjacent link [3].

Addressing detected neighbourhood changes at every node n
if the neighbour set changed
  notify sources of broken flows to reroute
  two-hop broadcast the *new* and *broken* links

Clique construction at every node n
if notification of *new* and/or *broken* links received
  update local topology view
  reconstruct clique resources
$\forall$ *newly identified* clique resources r_j
  identify the clique-leader node
$\forall\ r_j$ where n is clique-leader
  establish initial price, y_j
  $\forall$ flows f_i traversing r_j
    send q_{ij} and y_j to source of flow f_i

Flow adaptation to topology at every node n
$\forall$ flows f_i sourced at node n
  if f_i is broken
    reroute
  if q_{ij} and y_j from a clique r_j is received
    recompute path price, $pp_i = \sum_j q_{ij} \times y_j$
    recompute its clique-counter, $cc_i = \sum_j q_{ij}$
    send pp_i and cc_i to clique-leader of r_j

Fig. 5. Clique-construction and topology-related algorithms

Information from packets that traverse any link belonging to a clique should reach the clique-leader node. To minimise the in-clique signalling, the clique-leader is chosen as the node closest to all the links in the clique (i.e. the aggregated hop-count to the closest end-nodes of the links in the clique is minimal). Ties can be broken using node addresses. Note also that the "peripheral" nodes (i.e. those that are adjacent to only one link belonging to the clique) are neither candidates for clique-leader nor the nodes that communicate with the clique-leader on behalf of the link (i.e. the node closer to the clique-leader). Thus, for a proper clique construction it suffices to send neighbourhood information only 2 hops away. Similar to Xue et al [3], we use the Bierstone

algorithm [26] for clique identification. The Bierstone algorithm has a complexity of $O(l_l^2)$, where l_l is the number of links in the local analysed subgraph, which contains only nodes at most 2 hops away from the computing node.

In conclusion, for clique (re)computation, every node should only broadcast adjacent new and broken links (i.e. changes to its neighbourhood set) to all nodes as far as 2 hops away. Thus, after network initialisation, the signalling overhead involved in clique computation greatly depends on network mobility.

If a link breaks, all the flows that use this link should be rerouted. Some old clique resources will disappear and some new ones will be created. To set an appropriate starting price on the new cliques, we perform a "dry allocation" at creation time (no bandwidth is actually reallocated) based on the inherited flows. After a topology change all the affected flows must update their path price for the next allocation round.

Figure 5 shows the three independent algorithms used at each node for adapting to topology changes. No related synchronisation amongst nodes is needed. In the second algorithm, the "clique-leader" node is determined as the node in the clique that has the largest address, and is adjacent to at least two links belonging to the clique.

E. QoS routing

Traditional QoS routing algorithms typically use either shortest path (respecting minimum constraints), or widest path (allowing a better QoS for that flow). However, these are two extreme cases and do not optimise global utility. Shortest path might overload some routes. Widest path may produce too long routes, increasing total network load. Therefore, as part of adhoc-TARA we propose a new routing algorithm based on the shadow price of resources introduced above. Given the allocation algorithm presented in Section IV-C the best chance for a flow to receive the highest QoS is by routing it along a path with the lowest path price. Thus, we use an on-demand, shortest path first (SPF) routing algorithm that uses the path price as distance metric (i.e. it chooses a path that yields a minimal $pp_i = \sum_j q_{ij} \times y_j$)). This lowest path price routing algorithm takes into account both less contended links (lower link prices), and shorter topological paths (lower number of links).

Once chosen, keeping a route fixed is important for deterministic resource allocation. For this we employ the source-routing principle. In source routing the source specifies all the intermediate relays, and routing tables are not needed. Besides providing load balancing capabilities, source-routing prevents route and load oscillations when, for instance, the best route changes.

We perform flow rerouting only when a link in the end-to-end path breaks due to mobility. Topology events are independent of network load, and there might be enough bandwidth available to reroute the flow. On the other hand, rerouting should not be used in the case of a decrease in allocation. In this situation the network is most likely too loaded to accommodate that flow, so an alternative route will not help. Moreover, rerouting in such a case would create an oscillating allocation pattern where flows constantly chase a better route. In our experiments in Section V-H we compare the performance of our price-based routing to a hop-based SPF routing in both a uniformly loaded network and a network with hotspots.

F. Extending adhoc-TARA for asynchronous allocation

The version of adhoc-TARA presented in the previous sections relies on synchronous allocation at all clique resources. Synchrony is required since the source of the flow has to receive all new allocations (from all traversed clique resources) before it can choose the lowest allocation as the new rate for the upcoming period. Implementing this in an open ad hoc environment is problematic due to the following two factors. 1) Network-wide synchronisation of allocation rounds is needed due to the interweaving of flows and resources. This implies a network-wide consensus on allocation time points, good clock synchronisation, and fast and reliable transmission of the packets containing the new allocations and clique-prices from the clique leader to the source nodes of the flows. 2) Bursty control traffic must be dealt with. The information exchange between flow source node and clique-leaders can be seen in Figure 4. The source transmits λ_i and pp_i to all the traversed cliques. This information is the same for all cliques, and can be naturally piggybacked on the flow's packets. The dissemination is not time-critical (as it has the entire period to reach the relevant clique-leaders). In the other direction, however, the clique-leaders transmit all the new allocations x_{ij} and clique-prices y_j to all the source nodes of traversing flows (a much larger information size). This transmission is time-critical and might thus need special prioritised packets. Since all the clique-leaders transmit this information synchronised, the network has to deal with bursts of control information.

By using an asynchronous allocation scheme we could eliminate both the network-wide synchronisation maintenance and the bursty control traffic. To this end we propose the following solution. At every source node of a flow, we maintain a list of all the proposed allocations from all the clique resources that the flow is traversing. Whenever the flow's source is receiving a new bandwidth allocation, we update the value in the list that corresponds to the originating clique. Then we check if the minimum value of all the allocations in the list has changed. If yes, then we set it as the new rate of the flow. Note also that with a new allocation, a new clique-price is received too. Whenever a new clique-price is received, the path-price pp_i is immediately recomputed by the source and disseminated with the flow.

Using the previous mechanism, a flow's source need not wait for results from other cliques to arrive in order to decide if a rate change is opportune. Thus, the allocation moments for different clique resources may occur asynchronously. Note that care has to be taken for new and rerouted connections. We have to wait until all the traversed clique resources have decided on the first allocation (i.e. wait for the duration of one allocation round), before setting a flow's first transmission rate. That is, we accept a subflow only if it accepted by all clique resources along the path. Note that for a particular clique resource we still have a fixed period between two reallocations. As we assume the allocation time points of the different clique resources to be uniformly spread, control traffic will also be uniformly spread in time, in contrast to the periodic bursts of control traffic in the synchronised approach. In Section V-I we compare the synchronous and asynchronous versions of the scheme.

V. Evaluation

A. Evaluation setup

To evaluate the behaviour of our resource allocation scheme we use a traffic mix representative for future mobile communication

networks [27], [2]. Table II summarises the traffic characteristics. To create a diverse traffic mix, the maximum required bandwidth follows a geometric distribution with the given minimum, maximum and mean values.

TABLE II
TRAFFIC MIX USED IN THE EXPERIMENTS

Applic. Group	Max. Bandwidth Requirement (Kbps)			Connection Duration (sec)			Examples	Class	Utility Scaling Factor
	min	Max	avg	min	max	avg			
1	30	30	30	60	600	180	Voice Service & Audio Phone	I	1
2	64	256	128	60	1800	300	Video-phone & Video-conference	II	1/3
3	200	1000	500	300	7200	600	Interact. Multimedia & Video on Demand	II	1/10
4	10	30	20	10	120	30	E-Mail, Paging, & Fax	III	1/2
5	64	512	256	30	7200	180	Remote Login & Data on Demand	III	1/5
6	128	2000	512	30	1200	120	File Transfer & Retrieval Service	III	1/7

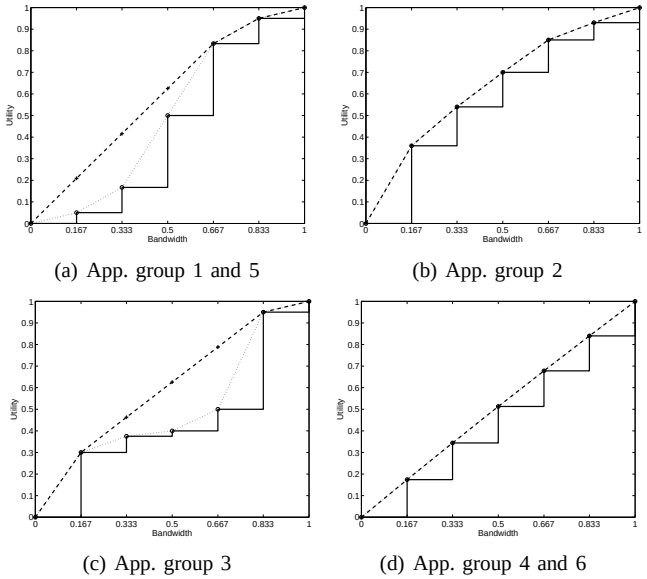
The second column from the right presents the flexibility class, which shows the flexibility of the applications to bandwidth reallocations.

- Class I represents rigid applications, e.g., for a real-time application once the "mode" is set by the initial allocation, any allocation increase is useless, and any decrease fatal. That is, the utility accumulated in time for this flow is completely lost if resources are decreased.
- Class II is semi-rigid, where the lowest allocation point is used to compute the utility for the whole duration (i.e. if resource is decreased a proportional chunk of already accumulated utility will be lost). Examples could be sensor flows with different accuracy, or sensitive multimedia.
- Class III represents fully flexible applications that have no problem to adapt (for every new allocation period the accumulated utility of the flow grows with the utility of the given allocation). Examples are non-real-time data transfers (FTP, email).

The allocation scheme accommodates the class I and II types by online modifications to the utility functions, prior to their usage by the allocation algorithm. These modifications are orthogonal to the allocation scheme presented here, and are presented in earlier work [10], [2].

The shapes of the utility functions for the different application groups distinguished in the first column in Table II are presented in Figure 6. On the x-axis we plot the normalised bandwidth with respect to the maximum requested by the application (Column 2 of Table II). On the y-axis we have the normalised utility with respect to the maximum utility, which is obtained by multiplying the maximum requested bandwidth with the *utility scaling factor* from the rightmost column of Table II. The utility scaling factor is important, as it provides a means to determine importance among application types. It represents the utility per bit associated with the maximum required bandwidth, and scales the shape of the utility function accordingly. For example, even though one might be ready to pay roughly three times more for a video-phone conversation (bandwidth demand of 256 Kbps), the utility per bit is almost three times higher for an audio-phone application (which requires only 30 Kbps).

Assigning utility values is always a subjective problem, so we chose some common sense values and also consulted with



(a) App. group 1 and 5 (b) App. group 2

(c) App. group 3 (d) App. group 4 and 6

Fig. 6. Utility functions for the six application groups

Ruben et al. [28] who performed a study at Ericsson Cyberlab in Singapore and had access to conceivable business models. In the graphs in Figure 6 the solid line corresponds to the original utility function (the one that defines the "contract" between user and provider), while the dashed line is the convex hull frontier, used for the allocation decision.

A simulator was built on top of the J-sim component platform [29], and packet level simulation was not considered at this stage. The experiments use $1500 \times 1500m^2$ area where 60 mobile stations are uniformly, randomly deployed. The communication range is $250m$ and considered equal to the interference range. Environmental perturbations are not considered, and every clique resource has the $4Mb/s$ channel bandwidth at their disposition.

Mobility is implemented using a modified random way-point model (RWP), with a random speed between zero and a maximum speed. As pointed out by Yoon et al. [30] RWP mobility models have deficiencies, and real applications may indeed show a different behaviour (see the work by Kuiper and Nadjm-Tehrani [31] for example). Thus, to avoid the slowdown of the original RWP, in our model we change the destination waypoint periodically (regardless if the destination waypoint has been reached or not). Moreover the nodes have no pause time. This provides for an average speed that is constant in time. To ensure a good connectivity and stop the mobile nodes from clumping together, we make nodes move away from each other when they come closer than a third of the communication distance. We believe this is a reasonable model in urban connectivity. The inter-arrival time of new flows follows an exponentially distribution, and all the six application groups arrive with equal probability. To solve the linear programming part, we have used a Java package from the operation research objects collection (OR-Objects) [32].

B. Comparison of allocation schemes

In this section's experiments we compare the behaviour of the following routing and allocation schemes for different load and mobility scenarios.

- As a baseline algorithm we use a non-utility routing and allocation scheme denoted by *simple* in the experiments. The

routing is on-demand shortest path first (hop-based). After a route is chosen, the minimum of the bandwidth available at all clique resources on the end-to-end path is allocated to the flow. Actually, if enough bandwidth is not available to accommodate the minimum then the flow is rejected. If the path breaks, the flow is rerouted, and new bandwidth allocated. If a clique resource becomes overloaded due to mobility, flows will be dropped on a last-accepted first-rejected basis.

- To represent best possible solution, we use a *LP* solver to optimally solve the global allocation problem as defined in Section III-C. The formulation of the LP problem does not include routing, so we use the price-based routing algorithm described in Section IV-E. This serves to compare our distributed allocation algorithm with the optimal allocation.
- Next we show the results of runs for the *adhoc-TARA* scheme. It uses the distributed allocation algorithm described in Sections IV-A to IV-C and the routing algorithm presented in Section IV-E.
- Finally, we compare with a variant of our distributed allocation scheme, denoted *altbid*, where a different formula is used to construct the bids. In this alternative the budget, λ_i^k, is proportionally divided based on shadow prices. Thus, $bid_{ij}^k = \frac{y_j \times \lambda_i^k}{\sum_j q_{ij} \times y_j}$. The intuition is that resources with higher shadow prices are more disputed, and thus have a higher chance of getting even more disputed. However, bids for low priced resources become very small, and in the case of a zero priced resource, all bids become null. In this case, ties are broken based on $slk_i^k = \lambda_i^k - pp_i$.

For all the four schemes, the main evaluation metric is the accumulated utility. Note that irrespective of how allocations are determined, utility accounting is performed using the original utility functions and the extended utility model briefly described in Section V-A. According to the model, a class I flow is dropped if the *initial* bandwidth cannot be maintained. A class II flow is dropped if the *minimum* bandwidth cannot be maintained. A class III flow is never dropped (unless there is a network partition). If a flow is dropped, no utility is gained for the flow, and the bandwidth invested during its lifetime is wasted.

C. Network-wide utility and mobility influence

As utility is our main performance metric we will first show how the four schemes behave when subjected to scenarios with different mobility. Thus in Figures 7 and 8, on the X-axis we have the average speed of the nodes (m/s), and on the Y-axis the time-accumulated system utility. Every plotted point represents an average of 8 different experiments. Each experiment was run over a period of $600sec$, with a (re)allocation period of 2 seconds. All the experiments were run with a moderate overload (average inter-arrival rate of $1/200s^{-1}$).

The experiments in the two figures are differentiated by the type of the applications used. In Figure 7 ("rt-mix" scenario) we consider a mix of rigid and flexible application groups as presented in Table II. In Figure 8 we consider that all the 6 application groups are fully flexible (their class is set to class III). In this case no flows will be dropped due to zero allocation.

We can see that the results of adhoc-TARA come surprisingly close to the optimal LP allocation. Even at the lowest point the distributed allocation algorithm is at almost 90% of the optimal

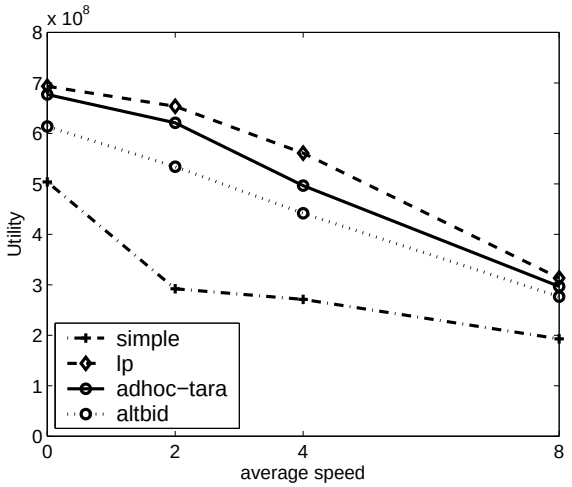


Fig. 7. Utility for rt-mix traffic

allocation (recall that LP uses the same routing as adhoc-TARA). Both the "flexible" and the "rt-mix" scenarios suffer from mobility in a similar way. The simple scheme cannot properly differentiate between flows and is trailing at around half of the utility of the LP algorithm. The altbid algorithm performs constantly below adhoc-TARA (at worst 72% of the LP). This is because the bid is too biased towards high-priced resources, while low priced resources can also quickly become more congested. Adhoc-TARA, on the other hand, creates a more balanced bid.

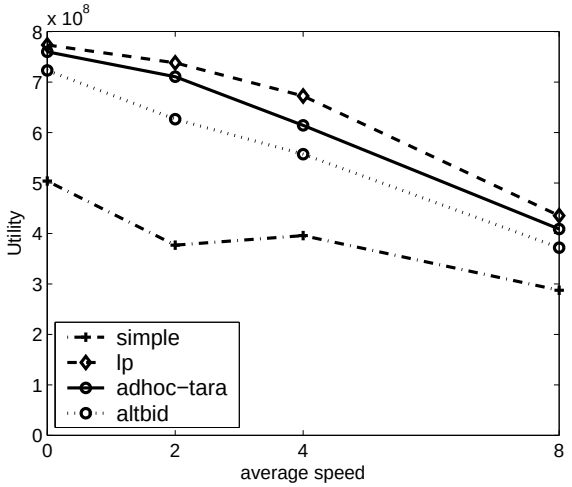


Fig. 8. Utility for flexible traffic

Figure 8 shows the achieved utility as an average of eight different topologies/traffic scenarios for every plotted point. Since measuring confidence intervals for utilities achieved over different topologies would be meaningless, we compute them on a different basis. That is, we compute a relative performance by dividing the results of *adhoc-TARA* and *simple* to the optimal results of *LP* that are achieved on the same traffic/topology setup. Only then we go on and compute statistical confidence. As exemplified in Figure 9 for "flexible" traffic, the 90% confidence intervals have the largest size of ± 0.015 (around a 0.93 point) for *adhoc-TARA*, while being roughly double, max ± 0.035 (around a 0.63 point), for *simple*. Though not shown as a graph, the confidence intervals for

Figure 7 are comparable; i.e. with the 90% confidence intervals for *adhoc-TARA* having max ± 0.04 (around a 0.97 point), and for *simple* max ± 0.05 (around a 0.61 point).

D. Network-wide utility and offered load dependency

The next set of experiments, presented in Figure 10, show how utility depends on the offered load. In all experiments the average speed is $4m/s$, we plot results for light, moderate and heavy offered load, and each point is an average of 8 experiments. Both "flexible" and "rt-mix" scenarios have similar trends. We can observe that the utility of all 3 schemes increases almost proportionally with load, preserving the superior performance of adhoc-TARA. The 90% confidence intervals for the relative utility performance of *adhoc-TARA* have max ± 0.06 (around a 0.94 point), and of *simple* have max ± 0.06 (around a 0.60 point).

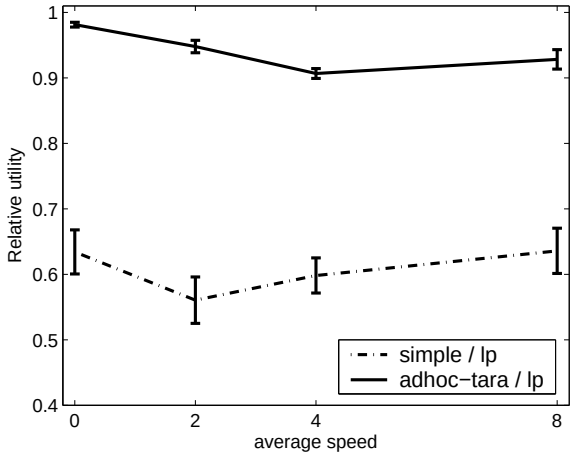


Fig. 9. The 90% confidence intervals for relative utility performance in the "flexible" traffic scenarios

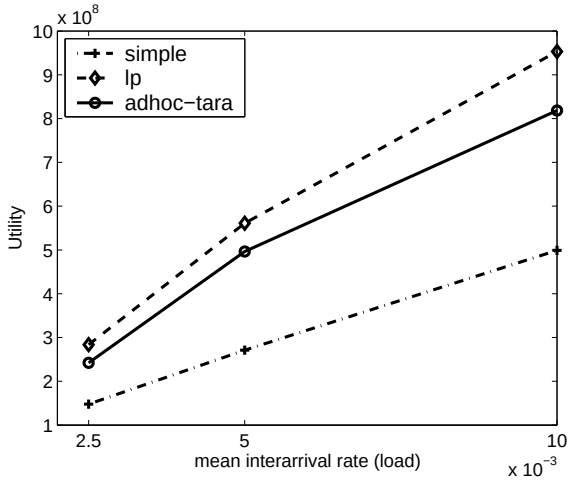


Fig. 10. Utility as function of traffic load for "rt-mix" traffic

E. Bandwidth mis-allocation in adhoc-TARA

In Table III we present the bandwidth utilisation of adhoc-TARA as compared to the utilisation of the LP algorithm. The mobility row shows the average speed of nodes (m/s). Our distributed algorithm independently allocates bandwidth at the clique resources along the flow's path. If allocations are different, some bandwidth is wasted. The LP solution is using global knowledge and thus has no such problem. Nevertheless, the difference in bandwidth usage (as an average over all clique resources) between LP and adhoc-TARA is only around 15% as presented in Table III.

TABLE III
BANDWIDTH USAGE OF ADHOC-TARA COMPARED TO LP

mobility	0	2	4	8
usage(%)	92	84	80	87

F. Running time comparison

Besides signalling overhead, computational complexity is a big drawback of the LP solution and one of the strengths of adhoc-TARA. Table IV gives a comparison of the average time (seconds, on a 1GHz P III) needed to reach an allocation decision, as offered load is increased. The running times of the experiments give a good indication of the relative merits of adhoc-TARA compared to optimal LP.

TABLE IV
AVERAGE RUNNING TIME OF ALLOCATION DECISION

interarrival rate	1/400	1/200	1/100
LP	54.1	173.8	890.9
adhoc-TARA	0.179	0.265	0.476

G. Run-time bound for convex hull related error

In Figures 7 and 8, we have presented the LP solution as optimal. Nevertheless, it uses the convex hull frontier approximation of the utility functions in its decision making, which may introduce an error compared to the true optimum, as shown by Lemma 3.2, in Section III-D. In this section we calculate a runtime bound of how large this error could be.

In Section III-D we defined U, U_{sup}, and U_{opt} and showed that $U \leq U_{opt} \leq U_{sup}$. As we can easily compute U and U_{sup} for every allocation round, we can compute a run-time difference bound between U and U_{opt}. Let's denote it as $\Delta U_{opt} = U_{sup} - U \geq U_{opt} - U$.

During an experimental run, 300 allocation rounds are performed in a simulated half hour. We experiment for different offered loads, as shown in Table V. We show the average number of connections in the system, together with the average value of $\Delta U_{opt}/U$ during the 300 allocation rounds. We can observe that the potential error introduced by using the convex hull frontier in the allocation decision is not larger than 5%, even though the flows used in our experiments are quite large compared to the capacity of the wireless channel. In networks populated with a large number of small sized flows we expect even better bounds.

TABLE V
APPROXIMATION BOUND FOR LP ALGORITHM

interarrival rate	1/400	1/200	1/100
avg. nr. of conns	14	29	60
avg. $\Delta U_{opt}/U$	0.043	0.047	0.040

H. Isolating the performance of price-based routing

In this section we measure the utility improvement due to the price-based routing algorithm presented in Section IV-E (labelled as *price* in the forthcoming curves) instead of a hop-based shortest path first routing algorithm (labelled as *spf*). We simulate two different load scenarios, one in which the load in the network is uniformly distributed, and the other one containing a hotspot region. The hotspot has a radius of $150m$ and is created in the middle of the $1500 \times 1500m^2$ simulation area. Contention in the hotspot is increased by setting the capacity of the cliques in the hotspot range to $1Mb/s$ instead of the $4Mb/s$ of the rest. The experiments with the hotspots scenarios are labelled with a *hs* suffix on the curves. In both cases the allocation is performed by adhoc-TARA, only the routing is different.

Figure 11 presents the network-wide utility and its variation with increasing mobility for a moderate overload (inter-arrival rate $1/200s^{-1}$). The results for of the scenario with uniformly distributed load show only marginal difference between price-based and hop-based SPF routing. This is expected, as in a uniformly loaded network, the shadow prices are similar and thus the path length largely determines path price. Basically in this case price-based routing degenerates to a hop-based SPF routing.

The hotspot scenario on the other hand shows a clear differentiation among the routing algorithms. In this case there is a real advantage for some flows to avoid the hotspot and take a longer route. This is visible in the mobility-independent 11% utility gained by using the price-based routing algorithm.

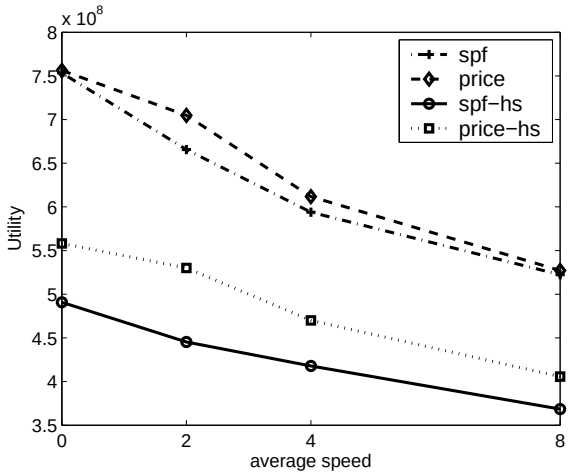


Fig. 11. Utility with different routing algorithms as function of mobility

Figure 12 shows the performance dependency on offered load, at an average mobility of $4m/s$. When the network is only lightly overloaded (inter-arrival rate $1/400s^{-1}$) best paths are the shortest paths and the difference is marginal. As already presented, under moderate overload the hotspot scenario shows the benefits of price-based routing. Note also that for heavy overloads (inter-arrival rate $1/100s^{-1}$), even for the scenario with no hotspots there is a difference between price-based and hop-based routing. This is due to the topology, that makes more connections to route through the centre of the area in the hop-based routing case.

For both Figure 11 and 12 each point is an average of 8 experiments with different initial topology/traffic. We computed confidence intervals in a similar manner to Section V-C. The 90% confidence intervals of the relative utility performance of

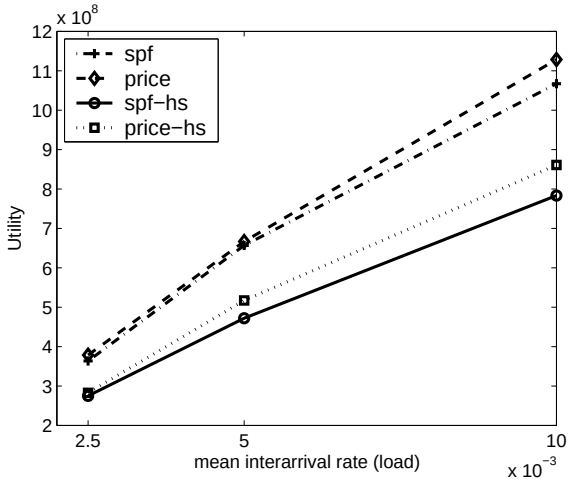


Fig. 12. Utility with different routing algorithms as function of load

spf divided by *price* are small; they have max ± 0.02 (around a 0.98 point), and of *spf-hs* divided by *price-hs* have max ± 0.045 (around a 0.86 point).

I. Evaluating asynchronous allocation

In the previous sections we evaluated a synchronous version of adhoc-TARA. As presented in Section IV-F, however, implementing an asynchronous version has several advantages. Thus, in Figure 13 we present the comparison of both the synchronous and asynchronous versions of the algorithm, as a function of both mobility and offered load. The suffixes of the labels show the interarrival rate of new connections. Intuitively, we have no reasons to expect a different achieved utility by the asynchronous allocation algorithm, and this is confirmed by the experiments. As future work we intend to theoretically prove that there is no difference among the two versions with respect to their convergence towards optimality. Each point is an average of 8 different experiments, and the confidence intervals further confirm the similarity of *async* and *synch*. For any of the mobility/load combinations the 90% confidence intervals of *async* divided by *synch* have max ± 0.013 (around a 1.02 point).

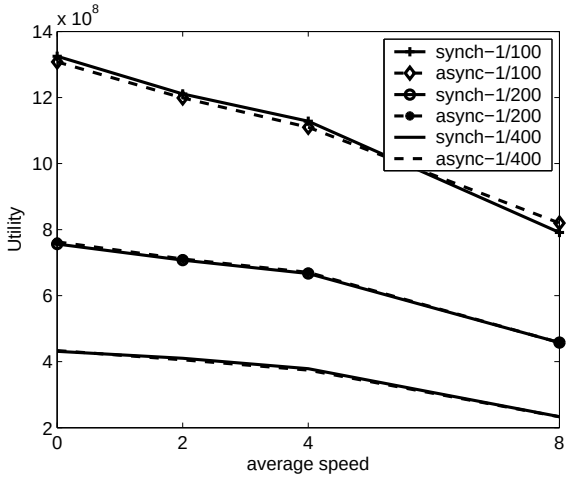


Fig. 13. Utility comparison between the synchronous and asynchronous allocations

VI. Comparing adhoc-TARA to the gradient projection algorithm

In this section we compare adhoc-TARA to another utility/price based resource allocation algorithm proposed by Xue et al. [5], [3]. As presented in the introduction, this algorithm belongs to a class based on the seminal formulation by Kelly et al. [33], [17], and uses gradient projection method to reach optimal allocation.

The algorithm, to which we refer to as the gradient projection allocation (GPA), addresses a maximisation problem that is very similar to ours, as presented by Equations (3)-(5). It also employs an iterative allocation algorithm that estimates the shadow prices of clique resources. The concepts of link contention and clique resources are identical in both works. Nevertheless, the solutions expose fundamentally different concepts, that also require utility functions with different characteristics. While concavity is needed in both cases, the GPA needs continuous, twice differentiable utility functions as an input. In the next two subsections we introduce the GPA algorithm and present how we constructed the utility functions in order to make both schemes comparable. Then we compare the two allocation methods focusing on convergence properties and performance measured in terms of accumulated utility.

A. The gradient projection algorithm

Starting from the optimisation problem presented in Equations (3)-(5) we introduce a set of multipliers y_j and relax the constraints to obtain the Lagrangian objective function:

$$\begin{aligned} L(x_i, y_j) &= \sum_{i=1}^{m} u_i(x_i) - \sum_{j=1}^{p} y_j(\sum_{i=1}^{m} q_{ij} \times x_i - B_j^{max}) \\ &= \sum_{i=1}^{m} u_i(x_i) - \sum_{i=1}^{m} x_i(\sum_{j=1}^{p} q_{ij} \times y_j) + \sum_{j=1}^{p} y_j \times B_j^{max} \end{aligned} \tag{19}$$

Now we can define the Lagrangian dual function:

$$DL(y_j) = \max_{0 \le x_i \le b_i^{max}} L(x_i, y_j) \tag{20}$$

and the dual problem accordingly:

$$\min_{0 \le y_j} DL(y_j). \tag{21}$$

The multiplier y_j can be interpreted as the price a flow has to pay for accessing clique j. Remember, the quantity $pp_i = \sum_{j=1}^{p} q_{ij} \times y_j$ in (19) corresponds to the accumulated price of all resources a flow i uses, and is referred as the flow's path-price. We observe that the last term in (19) is constant for given y_j and does not influence the optimal allocation solution. We can therefore neglect it without changing the problem. For clarity, we restate the Lagrangian dual function in its new form:

$$\begin{aligned} DL(y_j) &= \max_{0 \le x_i \le b_i^{max}} L(x_i, y_j) \\ &= \max_{0 \le x_i \le b_i^{max}} \sum_{i=1}^{m} u_i(x_i) - \sum_{i=1}^{m} pp_i \times x_i \end{aligned} \tag{22}$$

Thus, the optimisation problem is decomposed into two separate problems, the Subproblem (22), which aims at finding an optimal allocation given the clique prices y_j and the Subproblem (21) for finding the optimal prices for the cliques. From optimisation theory it is known that if the utility functions are concave, then an optimal solution y_j^* to (22), and the optimal solution x_i^* to the main problem (3)-(5) satisfy:

$$x_i^* \in \arg \max_{0 \le x_i \le b_i^{max}} L(x_i, y_j^*) \tag{23}$$

In other words we can obtain a solution to our initial problem (3)-(5) by solving the problems (21) and (22). Furthermore, y_j^* represents the *shadow price* of clique j.

Xue, Li and Nahrstedt [3] proposed the application of the GPA for utility optimised allocation. To use this method the utility functions must be twice differentiable. It is further assumed that the utility functions are strictly concave, and hence the problem has a unique optimal solution. The gradient projection method is an iterative method to find an extreme point of a constrained function. It approaches an extreme point by taking from the current position a step with a fixed length γ, in the direction of the (negative) gradient. If outside, the obtained point is projected back onto the feasible region.

In GPA, a gradient of Subproblem (22) for every resource dimension j is given by $\sum_{i=1}^{m} q_{ij} \times x_i - B_j^{max}$, and thus each component can be calculated separately on a given clique j, requiring only knowledge of the flows traversing the clique. Similarly, the allocated bandwidth can be determined by the source nodes, given the prices of the cliques that the flow traverses (since for all other cliques $q_{ij} = 0$). Hence, like for adhoc-TARA, the problem can be solved in a distributed manner without resorting to any global information about the network. It is shown in [3] that for a given set of prices, a unique optimal solution is obtained by letting $x_i = [\frac{d}{dx} u_i]^{-1}(pp_i)$. The algorithm in Figure 14 summarises the basic steps for the rate allocation (by $[...]^+$ we denote that negative values are set to 0, so the prices y_j remain in the feasible region). The clique part calculates new resource prices given the bandwidth allocations, and the flow part sets new bandwidth allocations as a function of traversed resource prices. By iteratively executing the two parts, both the allocation and prices converge towards the optimum, given that the right γ is chosen, and both topology and traffic configuration is static [33], [3]. As we will see in the experiments section, in a mobile environment the period between two iterations will play a big role with respect to the performance.

Price update algorithm run at leader node of every clique j
update price, $y_j = [y_j - \gamma(B_j^{max} - \sum_{i=0}^{m} q_{ij} \times x_i)]^+$
send y_j to all flows traversing clique j

Bandwidth allocation algorithm run at source of every flow i
calculate path price, $pp_i = \sum_{j=1}^{p} q_{ij} \times y_j$
calculate allocation, $x_i = [\frac{d}{dx} u_i]^{-1}(pp_i)$

Fig. 14. The gradient projection algorithm (GPA)

On a conceptual level we can differentiate GPA and adhoc-TARA by noting that the GPA algorithm is a *congestion control* algorithm. We can observe that the price of a resource increases (and subsequently the rate of the flow decreases) only as a consequence of an overload. Moreover, as all the flows take independent rate change decisions, an underload in a resource triggers a rate increase of all the traversing flows, which can lead then to an overload, and so on. Adhoc-TARA on the other hand can be labelled as an *admission control* algorithm, since every clique employs explicit allocation rounds. Thus flows do not adapt independently, and resources do not get overloaded.

Another difference is that the GPA can adapt only in small steps, so in the case of severe over/underload the algorithm needs several iterations to allocate the real amount of resource. In comparison, adhoc-TARA always attempts to fully allocate according to the right resource capacity.

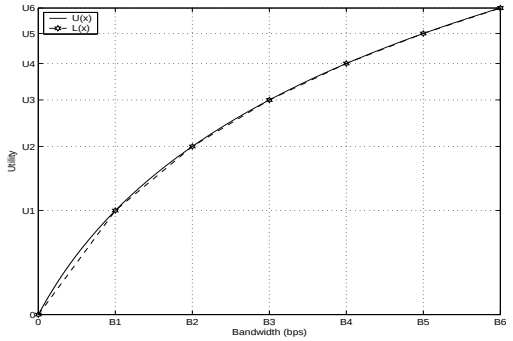


Fig. 15. A logarithmic utility function is interpolated to obtain a piecewise linear function

B. Utility functions and accounting

The gradient projection method requires the utility functions to be continuous and differentiable while for adhoc-TARA they have to be piece-wise linear. We adopt the following strategy in order to perform comparable evaluation. For GPA we use utility functions that are specified as $U(x) = a\log(bx + c)$, where a, b, c are arbitrary parameters that control its shape. For adhoc-TARA these functions are linearly interpolated at 7 equidistant points so as to get a piecewise linear utility function consisting of 6 segments, say $L(x)$ (cf. Figure 15).

The utility accounting is based on each algorithm's utility function. More precisely, given a period length of τ, the utility for allocation x_i^t for period $[t, t+\tau]$ is given by $u_i^t = \tau L(x_i^t)$ for adhoc-TARA and $u_i^t = \tau U(x_i^t)$ for GPA. The system utility is then simply $\sum_t \sum_{i=0}^{m} u_i^t$. Although $L(x)$ is slightly lower than $U(x)$ our experiments showed that this difference is negligible.

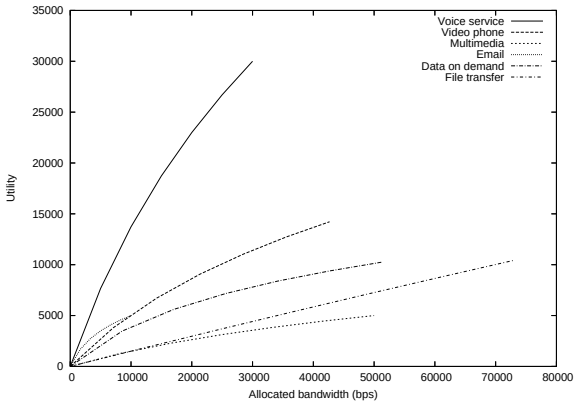


Fig. 16. The traffic mix used for subsequent experiments

To run the comparative analysis we have modified the utility functions of the six application groups in the traffic mix (see Section V-A) to the ones depicted in Figure 16. Projection of each curve's end point on the x-axis shows the average bandwidth that is used by a connection of this class.

The GPA algorithm cannot accommodate rigid and semi-rigid flows (as the ones presented in Section V-A), and for a fair comparison, we treat all six application classes as fully flexible connections for both algorithms.

Furthermore, during overload situations, the GPA might allocate more bandwidth than it is available at a resource. In real networks this would lead to packet drop and retransmission. We simulate the packet drop by granting all connections the allocated bandwidth until the resources at a clique are exhausted. The remaining connections will then be allocated zero bandwidth. The possible overhead that frequent retransmission induces in real networks is not accounted for.

For the experiments presented in the next sections we extended our J-sim-based simulator to perform allocation using the GPA algorithm. We used the synchronous version of the simulation environment. For routing we used the on-demand shortest path algorithm where the length of a path corresponds to the number of hops. Packet-level simulation is not implemented and thus packet-level overhead are not studied in this context.

C. Convergence properties

To illustrate the basic functioning of the algorithms we apply them to the simple, static scenario depicted in Figure 17, with a logarithmic utility function $U(x) = \ln(x)$. It is linearly interpolated at 7 points to obtain a piecewise linear function for adhoc-TARA, as shown in Figure 15. The required bandwidth of both flows is $3Mbps$. The resource capacity B_j^{max} is set to $4Mbps$.

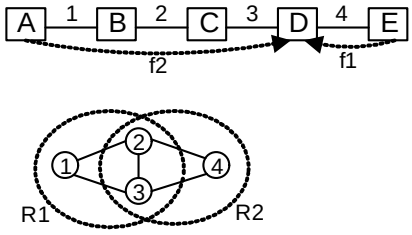
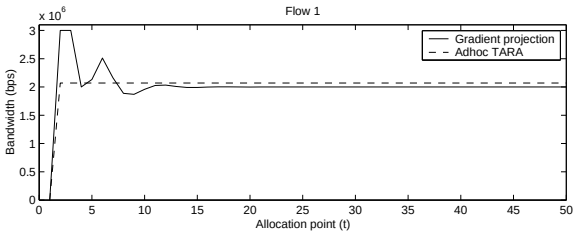
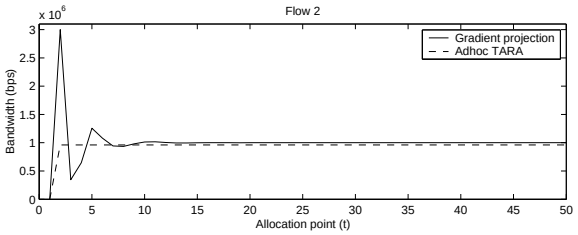


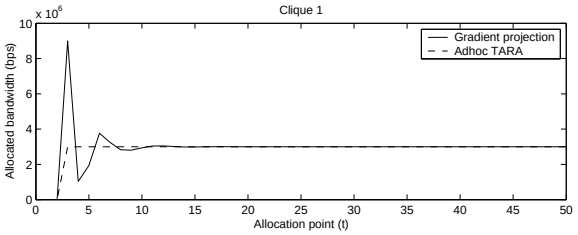
Fig. 17. A simple network configuration and its contention graph

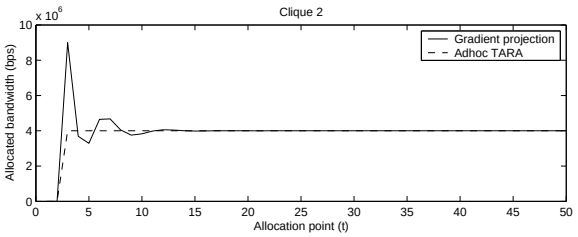
Figure 18 illustrates how the two algorithms allocate bandwidth to cliques and flows in this situation. We observe that once both algorithms have converged, the allocation is quite similar, but not exactly the same. The reason is that adhoc-TARA can only allocate at a discrete number of points. We also see that GPA needs a few iterations to converge to the optimal solution, while in this simple scenario adhoc-TARA is able to reach near-optimal allocation at once. In GPA the allocation oscillations are a consequence of the flows adapting independently to the available bandwidth. Right in the beginning both flows observe an empty resource and move to occupy it, then they observe the overload, and so on. In Figure 18(b), where the bandwidth allocation at the two cliques is depicted, we also observe that in the initial phase the GPA allocates more bandwidth than is actually available, until it converges to a feasible allocation. In real networks this would lead to packet loss and retransmission. Figure 18(a) shows that in an optimal solution the allocated bandwidth for flow 2 is twice as small as for flow 1 despite the fact that both flows have the same utility function. This allocation makes sense, as flow 2 uses two links in clique $R1$ while flow 1 uses only one.





(a) Connection bandwidth





(b) Clique bandwidth

Fig. 18. Bandwidth allocation for the flows and cliques in Figure 17

D. Performance comparison and mobility dependency

For the experiments that follow we use the same simulation parameters as the ones presented in Section V-A, with two differences. First, the comparable utility functions presented in Section VI-B, Figure 16 are used. Second, we introduce the step-length parameter γ, which is particular to the GPA, and unless otherwise specified has the following value, γ =1e-10.

We proceed by studying the influence of mobility on the accumulated utility. Nodes join and leave the interference range of other nodes, hence changing the clique's load abruptly. Moreover, a route can be lost forcing the connection to take another route. As already discussed, it may take quite a few iterations for GPA to adapt when facing such abrupt changes, whereas adhoc-TARA's allocation reflects this change immediately. We therefore expect that the performance of GPA deteriorates quicker than adhoc-TARA. Figure 19 clearly confirms our expectation. Similar to the results in Section V-C we can observe that the performance of adhoc-TARA is very close to the optimal LP-based algorithm. There are two important factors that influence the performance of the GPA in highly dynamic environments. These are the period between iterations, τ, and the step-length γ, and we look closer at them in the next sessions. Note that in this graph's experiments, we used $\tau = 0.02$ s and γ =1e-10. For every point we performed 4 experiments with different initial topology/traffic. We computed the relative performance of both *adhoc-TARA* and *GPA* divided by *LP* as we did Section V-C. The 90% confidence intervals are small, for *adhoc-TARA* they have max ± 0.013 (around a 0.98 point), and for *GPA* max ± 0.035 (around a 0.94 point).

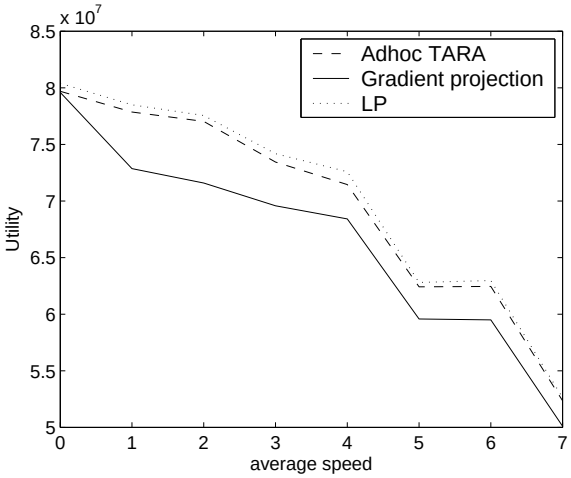


Fig. 19. Influence of mobility on the performance

E. Influence of period length on the performance

As we have already mentioned, the period time (τ) in our mobile simulation is of crucial importance for GPA. The larger the period, the longer the GPA stays in suboptimal allocation after a traffic or topology change. And these changes occur quite often in a mobile scenario, with important consequences. Thus, it is clear that the accumulated utility is monotonically increasing as τ moves closer to zero. Figure 20 shows how the accumulated utility is affected by the interallocation period length. Each point is an average of 4 experiments, and the relative performance of *GPA* divided by *adhoc-TARA* stable. The 90% confidence intervals have max ± 0.013 (around a 0.98 point).

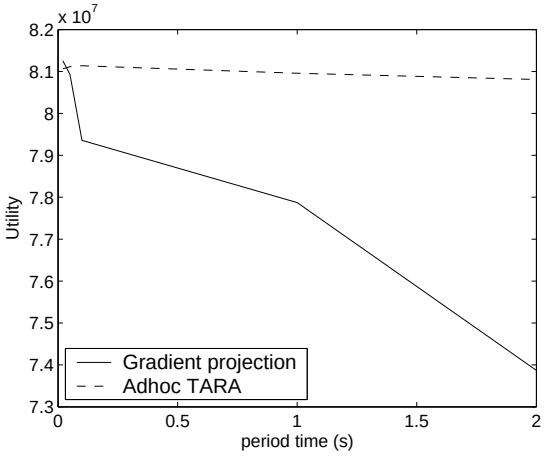


Fig. 20. Influence of the period time τ on the performance

We observe that a short period time is vital for GPA to show good performance, while adhoc-TARA is less sensitive to longer dissemination times. The less sensitive behaviour of adhoc-TARA is explained by two facts. First, in adhoc-TARA the flows do not adapt independently, and a clique never allocates more bandwidth

than available. Second, if a change in the network demands it, adhoc-TARA can operate large changes in a flow's allocation (from max to zero, or vice versa). In GPA on the other hand, flows need to adapt in steps, the size of which depends on γ.

In a real network, the period is lower-bounded by the end-to-end transmission time of the packets, and regulates how often the control information is transmitted. Thus, there is a trade-off between signalling overhead and the time GPA needs to converge to an optimal solution.

F. Influence of step-length on GPA

Choosing the right step-length, γ, is crucial for the performance of GPA. Xue et al. [3] show that convergence is guaranteed if γ satisfies $0 < \gamma < 2/\overline{\kappa}\overline{Y}\overline{Z}$, where, informally speaking, $\overline{\kappa}$ is a bound on the curvature of the utility functions, $\overline{Y}$ the length of the longest path for a flow and $\overline{Z}$ the number of sub-flows at the most congested clique. In our experiments we note that $\overline{\kappa}$ is the dominating factor, and it turns out to be of the same order of magnitude as the requested bandwidth of a connection. If we have a good idea about the peak traffic in our network, the above formula helps us to choose the step-length small enough. Unfortunately such information is rarely available.

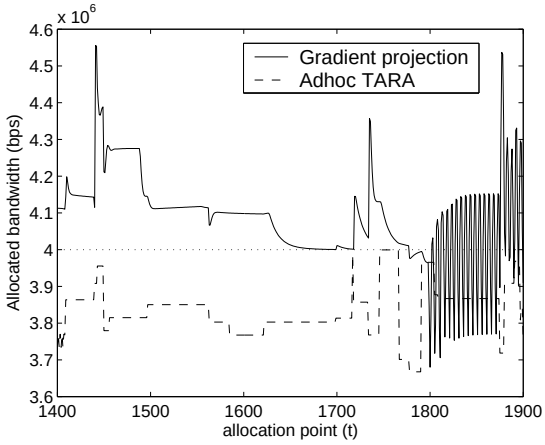


Fig. 21. Allocated bandwidth at a sample clique (inter-arrival rate $1/200s^{-1}$)

Choosing a too large step length can result in oscillating allocation behaviour, as shown in Figure 21, starting around allocation point 1800. This figure shows the bandwidth allocated by GPA to an arbitrary clique with γ =1e-10. Being conservative and choosing the step-length too small will ensure convergence. This, however, is done at the cost of convergence speed, and thus the system spends more time in suboptimal states.

Optimal step-length depends on the traffic type, as illustrated in Figure 22. The simulations were performed for two different traffic models, namely the mixed traffic presented in Section V-A with an inter-arrival rate of $1/600s^{-1}$, and a traffic setup where all the connections are of "file transfer" type (see Figure 16). The inter-arrival rate in the latter case is $1/1000s^{-1}$.

VII. Conclusions and Future Work

In this paper we present a novel utility/price-based bandwidth allocation scheme for wireless networks, together with a compatible price-based routing algorithm. We first show that we can use discrete utility functions together with linear programming for optimising resource allocation in multihop ad hoc networks.

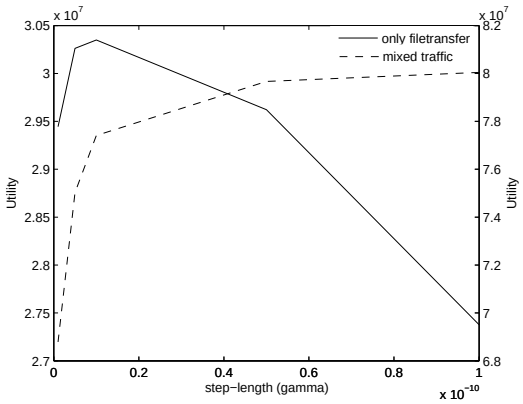


Fig. 22. Influence of step-length in GPA on the accumulated utility for two different traffic mixes

We then propose adhoc-TARA, a distributed allocation algorithm that bids for resources depending on their *shadow prices*, and the *utility efficiency* of the flows. Simulations show a very good performance of the distributed allocation algorithm, comparable to an optimal LP based global allocation, and with a much lower overhead. Furthermore, in hotspot scenarios price-based routing shows its benefits as compared to hop-based SPF routing.

Since synchronous allocation might be hard to implement in an ad hoc setting, we then present an asynchronous version of the algorithm and show that its performance is not affected by this change.

Finally, we compare adhoc-TARA to another type of distributed price-based allocation algorithm, which is based on the gradient projection method. The simulations show that adhoc-TARA is much more robust with respect to both mobility and length of the allocation period. On the other hand, the GPA has been theoretically proven to converge towards the optimum [17], [3].

As a future work we aim to study convergence conditions and properties of adhoc-TARA, and theoretically prove that it converges towards the optimum. Current work includes the implementation of needed additions and modifications throughout the protocol stack of an ad hoc network, to test it using detailed packet-level simulations. We aim to study and compare the packet-level overheads introduced by our allocation algorithm. Complementary simulation studies are needed for testing the resilience of the algorithm to loss of control packets, yielding guidelines on how we can better trade-off signalling overhead against control accuracy.

Wireless networks face a paradigm shift. They intend to complement the Internet with its different services and applications, with much less available resources. Thus, we argue that without a quantitative measure for the importance of the flows, the network cannot provide resource assurance and allocation flexibility at overloads. Under these conditions, combining utility functions with a lightweight distributed implementation could provide a very strong argument to get rid of the old performance metrics and optimise the QoS as perceived by the user.

Acknowledgement

This work was supported by the Swedish National Graduate School in Computer Science (CUGS). We would like to thank Marcel Lüthi for his essential contribution in comparing adhoc-TARA to GPS. The second author gratefully acknowledges the

support of University of Luxembourg during the preparation of this manuscript.

REFERENCES

[1] 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.

[2] C. Curescu and S. Nadjm-Tehrani, "Time-aware utility-based resource allocation in wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 7, pp. 624–636, July 2005.

[3] Y. Xue, B. Li, and K. Nahrstedt, "Optimal resource allocation in wireless ad hoc networks: A price-based approach," *IEEE Transactions on Mobile Computing*, vol. 5, no. 4, pp. 347–364, 2006.

[4] H. Luo, S. Lu, V. Bharghavan, J. Cheng, and G. Zhong, "A packet scheduling approach to QoS support in multihop wireless networks," *Mobile Networks and Applications*, vol. 9, no. 3, pp. 193–206, 2004.

[5] Y. Xue, B. Li, and K. Nahrstedt, "Price-based resource allocation in wireless ad hoc networks," in *Proceedings. of the 11th International Workshop on Quality of Service (IWQoS), also in Lecture Notes in Computer Science*, vol. 2707. ACM Springer-Verlag, jun 2003, pp. 79–96.

[6] Q. Xue and A. Ganz, "Ad hoc QoS on-demand routing (AQOR) in mobile ad hoc networks," *Journal of Parallel and Distributed Computing*, vol. 63, no. 2, pp. 154–165, 2003.

[7] S. Chen and K. Nahrstedt, "Distributed quality-of-service routing in ad-hoc networks," *IEEE Journal on Special Areas in Communications*, vol. 17, no. 8, pp. 1–18, aug 1999.

[8] C. R. Lin and J.-S. Liu, "QoS routing in ad hoc wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1426–38, 1999.

[9] J. N. Al-Karaki and A. Kamal, *Quality of Service Routing in Mobile Ad hoc Networks: Current and Future Trends*. CRC Publishers, 2004, ch. Mobile Computing Handbook.

[10] C. Curescu and S. Nadjm-Tehrani, "Time-aware utility-based QoS optimisation," in *Proceedings of the 15th Euromicro Conference on Real-time Systems*. IEEE Computer Society, July 2003, pp. 83–93.

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

[12] V. Bharghavan, K.-W. Lee, S. Lu, S. Ha, J.-R. Li, and D. Dwyer, "The TIMELY adaptive resource management architecture," *Personal Communications, IEEE*, vol. 5, no. 4, pp. 20–31, aug 1998.

[13] A. Eryilmaz and R. Srikant, "Joint congestion control, routing and MAC for stability and fairness in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1514–1524, Aug. 2006.

[14] J. W. Lee, M. Chiang, and R. A. Calderbank, "Jointly optimal congestion and contention control in wireless ad hoc networks," in *Proceedings of the IEEE Vehicular Technology Conference*, May 2006, Melbourne, Australia.

[15] X. Lin and N. B. Shroff, "An optimization-based approach for QoS routing in high-bandwidth networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 6, pp. 1348–1361, 2006.

[16] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Multi-path tcp: A joint congestion control and routing scheme to exploit path diversity on the internet," *IEEE/ACM Transactions on Networking*, vol. 14, no. 6, pp. 1260–1271, Dec. 2006.

[17] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, 1998.

[18] X. Lin, N. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452–1463, 2006.

[19] C. Curescu and S. Nadjm-Tehrani, "Price/utility-based optimized resource allocation in wireless ad hoc networks," in *Proceedings of the 2nd Conference on Sensor and Ad Hoc Communications and Networks (Secon)*. IEEE Communications Society, Sept. 2005.

[20] M. Luethi, S. Nadjm-Tehrani, and C. Curescu, "Comparative study of price-based resource allocation algorithms for ad hoc networks," in *Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE Computer Society, April 2006.

[21] C. Lee, "On quality of service management," Ph.D. dissertation, Carnegie Mellon University, Aug. 1999, technical report CMU-CS-99-165.

[22] ITU, "ITU-R Recommendation BT.500-10: "Methodology for the subjective assessment of the quality of television pictures."," 2000, Geneva, Switzerland.

[23] EBU Project Group B/VIM Video In Multimedia, "SAMVIQ - subjective assessment methodology for video quality," BPN 056 report, May 2003, http://www.ebu.ch/trev_301-samviq.pdf.

[24] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*. Boston: Kluwer Academic Publishers, 1996.

[25] K. Römer, P. Blum, and L. Meier, "Time synchronization and calibration in wireless sensor networks," in *Handbook of Sensor Networks: Algorithms and Architectures*, I. Stojmenovic, Ed. John Wiley, Sept. 2005.

[26] J. G. Augustson and J. Minker, "An analysis of some graph theoretical cluster techniques," *J. ACM*, vol. 17, no. 4, pp. 571–588, 1970.

[27] 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*, vol. 16, pp. 858–878, Aug. 1998.

[28] F. Ruben and F. Edlund, "Design of a mobile payment intermediary," Master's thesis, Linköping University, LiTH-IDA-Ex-02/108, Dec. 2002.

[29] http://www.j sim.org/, "J-sim homepage."

[30] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *Proceedings of IEEE INFOCOM*, Apr. 2003, pp. 1312–1321.

[31] E. Kuiper and S. Nadjm-Tehrani, "Mobility models for UAV group reconnaissance applications," in *Proceedings of International Conference on Wireless and Mobile Communications*. IEEE Computer Society, July 2006.

[32] http://opsresearch.com/OR Objects/index.html, "Or-Objects homepage."

[33] F. P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, Jan. 1997.

Calin Curescu received his BSc degree in computer engineering from Politehnica University, Timisoara, Romania in 1999 and his PhD degree in computer science from Linköping University, Sweden in 2005. He is currently with Ericsson Research in Stockholm, Sweden. His research interests include new service paradigms in conjunction with network convergence, quality of service in dynamic distributed systems, wireless networks, and real-time systems.

Simin Nadjm-Tehrani is a full professor at Linköping university (LiU), Sweden, and acts as a professor in dependable real-time systems also at University of Luxembourg. She has led the real-time systems laboratory (RTSLAB) at the department of Computer and Information Science, LiU, since year 2000. She obtained her BSc from Manchester University, England, and her PhD from Linköping University in 1994. Her current research interest is in the area of dependable distributed systems and networks, including analysis of safety and fault tolerance, reliable communication and measuring availability in presence of failures, attacks and overloads.