Comparative Study of Price-based Resource Allocation Algorithms for Ad Hoc Networks

Marcel Lüthi, Simin Nadjm-Tehrani†, Calin Curescu

Department of Computer and Information Science
Linköping University
{g-marlu,simin,calcu}@ida.liu.se

Abstract

As mobile ad hoc networks provide a wide range of possibly critical services, providing quality of service guarantees becomes an essential element. Yet there is a limited understanding of the performance characteristics of different resource allocation algorithms. In particular, there is little work that comparatively studies different algorithms in the same traffic environment. Therefore we study two algorithms, adhoc-TARA and an algorithm based on the gradient projection method, for optimised bandwidth allocation in ad hoc networks under overload situations. The focus is on convergence properties and performance measured in terms of accumulated utility. The simulation results show that the gradient projection algorithm converges to an optimal solution even in large, dynamic networks, but that in such dynamic environments the convergence time can significantly influence the overall performance. In comparison, the near-optimal algorithm adhoc-TARA, which quickly adapts to changes in the state of the network, can exhibit superior performance. Further we illustrate how different parameter settings influence the performance of the algorithms. We conclude that finding an optimal allocation comes at a high price in the rapidly changing environments of ad hoc networks and that near-optimal allocation can be an ample alternative.

1. 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 a source to a destination. Mobile ad hoc networks aim to provide a wide range of services in which soft real-time (multimedia), and high priority critical data, should seamlessly integrate. As society becomes dependable on provision of such services, their availability under overloads becomes a critical issue. If not adequately dealt with, this problem may become a serious threat to critical infrastructures of the future.

Due to the nature of ad hoc wireless networks, resource allocation and routing that respects quality of service (QoS) parameters should be adaptive to the rapidly changing environment, work in a distributed manner, and have light computational and signalling overhead. While CPU utilisation and power are important resources to consider, here we focus on allocation of bandwidth as a resource.

Although the need for resource allocation in ad hoc networks has clearly been recognised and many allocation-algorithms were proposed (e.g. [11, 5, 4, 12, 13, 6]), their comparative performance has, to the best of our knowledge, not yet been studied. In the related field of ad hoc routing however, steps in this direction have already been taken (e.g. [2, 3, 8]) and a picture of the properties of different algorithms begins to emerge.

In this paper we compare two algorithms for optimised resource allocation in ad hoc networks. Both algorithms attempt to maximise the accumulated utility, which is given as the sum of utility experienced by the users for given bandwidth allocations. The first algorithm, proposed by Xue, Li and Nahrstedt [12, 13], uses the gradient projection method on the Lagrangian dual of the optimisation problem to find an optimal allocation (here referred to as GPA). The second algorithm, Time Aware Resource Allocation (adhoc-TARA), proposed by Curescu and Nadjm-Tehrani [6], is also based on Lagrangian duality but uses a bidding scheme to allocate resources.

†Corresponding author

1-4244-0054-6/06/$20.00

Xue et al. [13] proved that their allocation scheme converges to the optimal solution and provided experimental results for small to moderate sized networks and light load. Also, some experiments with mobility for small networks have been performed. Adhoc-TARA has been evaluated in an experimental setting with mixed traffic, mobility, and highly overloaded scenarios. Our motivation for this comparison is to get a deeper understanding of both algorithms and characterise their respective strengths in a common experimental platform.

It is worthwhile to assess GPA's performance for larger networks under overload situations. Thereby, for different connection interarrival times and under mobility, we can study the time that the system remains in a non-optimal state. We conjecture that this time may be significant in scenarios where the state of the network is frequently changing. Thus, it is interesting to compare GPA with adhoc-TARA, which has so far shown promising results in such environments.

Time awareness in adhoc-TARA was designed to deal efficiently with rigid traffic. For rigid traffic a minimal allocated bandwidth has to be guaranteed during the whole lifetime of the connection. If the connection is dropped due to overloads, its accumulated utility will be lost (examples are voice or video phone applications). As GPA does not incorporate special mechanisms to handle rigid traffic, this paper is only based on flexible traffic and hence not exploiting the full potential of adhoc-TARA in this respect.

Earlier studies of GPA have been performed based on detailed simulations at packet level. To overcome the challenge of running both algorithms on a common platform we have made an abstraction that ignores the packet level and hence cannot quantify overhead aspects only measurable in such environments. Being aware of this deficiency, we still believe that this comparative study has a value and provides new insights for both algorithms. Our simulation results show that even in large networks under mobility and highly dynamic traffic patterns, the gradient projection algorithm converges to an optimal solution. However, the experiments support our conjecture that in such dynamic environments the convergence time for the gradient projection algorithm can significantly influence the overall performance and that a near-optimal algorithm, such as adhoc-TARA can show superior performance. We investigate the influence of the parameter step-length in GPA and show that the optimal value for the step-length depends on the traffic characteristics. Further, the experiments show that GPA's performance is very dependent on quick dissemination of the price information whereas adhoc-TARA is more robust in this respect.

The paper is organised as follows: In section 2 we introduce the mathematical model underlying both algorithms and formulate a general optimisation problem to be solved. Thereafter we describe how the two algorithms solve this optimisation problem. Section 3 discusses and illustrates convergence properties of the algorithms. After a discussion of our simulation environment and methodology we present in section 4 the results of our experimental study under varying traffic and mobility conditions.

2. Background

Both algorithms we compare in this paper assume that each flow comes with a resource-utility function. The function indicates the utility that is accrued if the flow is allocated a given resource level. Both algorithms also apply the idea of price-based resource allocation. The basic concept is to set prices on mutually contending links based on their congestion. The goal is to allocate the bandwidth in such a way that the network's utility is maximised.

A distributed algorithm is obtained by considering the Lagrangian dual of the optimisation problem and hence decomposing the problem. A thorough discussion and motivation of using Lagrangian duality for rate control in networks is given in the seminal paper of Kelly et al. [10].

We first present the general mathematical model underlying both algorithms before proceeding with the details of the algorithms.

2.1. Mathematical model

This section is essentially a condensed version of the mathematical model as described in [12]. Throughout this document boldface symbols represent vectors. When a vector is used to refer to elements of a set, we use the set elements as index, e.g. v_f refers to the component corresponding to element f of set F.

As usual the ad hoc network is modelled as a graph $G(V, E)$, where V is a set of nodes (i.e. wireless devices) and E a set of bidirectional links. Each node $v \in V$ has transmission range d_{tx} and interference range d_{int}. Two nodes $v_i, v_j \in V$ are connected by an edge $\{v_i, v_j\} \in E$ if they are within each other's transmission range.

We define F to be a set of end to end flows, where each flow $f \in F$ can span multiple hops, and denote the set of links comprising flow f as $E(f)$. If for a flow either the source or the destination of a single hop is within the interference range d_{int} of another flow, the

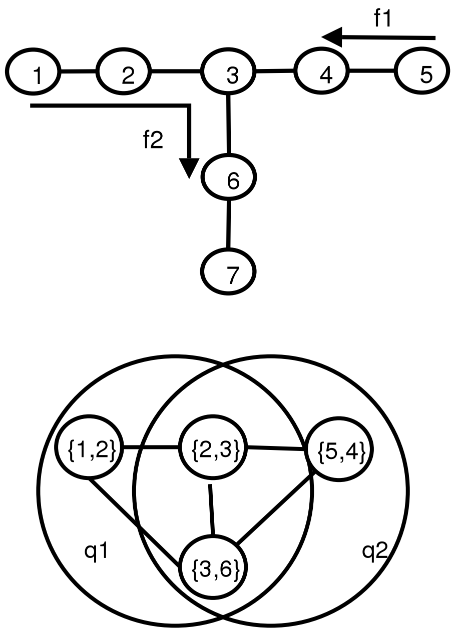


Figure 1. A simple network configuration and its contention graph

flows are said to contend. Given the characteristics of the medium, it is clear that in such a case only one of the flows can send at a given time. The contentions at a specific time can be modelled by a link contention graph $G_c(V_c, E_c)$ where the vertices correspond to the wireless links (i.e. $V_c = E$) and there exists an edge between two vertices if there are flows which contend for these links (i.e. they cannot transmit simultaneously). The *cliques* in the contention graph then represent mutual contending flows that "share" the medium (and hence the available bandwidth). It is therefore natural to regard maximum cliques (i.e. cliques that are not contained in any other clique) as the fundamental resource for pricing in the network. The set of maximal cliques q is denoted by Q.

We define $\boldsymbol{R} = (r_{qf})$ to be the $|Q| \times |F|$ matrix consisting of the elements $r_{qf} = |V_c(q) \cap E(f)|$, that is the number of links flow f uses within clique q.

To illustrate these concepts consider Figure 1, which depicts a simple network configuration and its contention graph. In this example the matrix R becomes

$$\boldsymbol{R} = \begin{bmatrix} 0 & 3 \\ 1 & 2 \end{bmatrix}.$$

Let C_q be the available bandwidth of clique q. A bandwidth allocation $(x_f | x_f \geq 0, f \in F)$ is said to be feasible if and only if

$$\sum_{f \in F} r_{qf} x_f \leq C_q, \ \forall q \tag{1}$$

To quantify the "satisfaction" gained from an allocation x_f, utility functions $U_f : \mathbb{R}_0^+ \to \mathbb{R}$ are defined. That is, given an allocated bandwidth x_f, the flow f accrues a utility corresponding to $U_f(x_f)$. The utility functions are used as inputs to the algorithms in this paper, and also for evaluating results for each algorithm. Then the system utility is the sum of the individual flow's utilities. We are thus able to formulate the optimisation problem

$$\begin{aligned} z^* := \max_{\boldsymbol{x}} & \sum_{f \in F} U_f(x_f) \\ \text{s.t.} & \sum_{f \in F} r_{qf} x_f \leq C_q, \ \forall q \\ & 0 \leq x_f \leq M_f, \ \forall f \end{aligned} \tag{2}$$

where M_f is the maximal bandwidth needed by flow f.

We introduce the vector of multipliers $\boldsymbol{\mu}$ and relax the constraints (1) to obtain the Lagrangian function

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{\mu}) &= \sum_{f \in F} U_f(x_f) - \sum_{q \in Q} \mu_q (\sum_{f \in F} r_{qf} x_f - C_q) \\ &= \sum_{f \in F} U_f(x_f) - \sum_{f \in F} x_f (\sum_{q \in Q} \mu_q r_{qf}) + \sum_{q \in Q} \mu_q C_q \end{aligned} \tag{3}$$

We let $\boldsymbol{X} = \{\boldsymbol{x} | \boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{M}\}$ and define the Lagrangian dual function

$$\theta(\boldsymbol{\mu}) = \max_{\boldsymbol{x} \in \boldsymbol{X}} L(\boldsymbol{x}, \boldsymbol{\mu}) \tag{4}$$

and the dual problem accordingly

$$\theta^* = \min_{\boldsymbol{\mu} \geq \boldsymbol{0}} \theta(\boldsymbol{\mu}). \tag{5}$$

The multiplier μ_q can be interpreted as the price a flow has to pay for accessing clique q. Consequently the quantity $\sum_{q \in Q} \mu_q r_{qf}$ in (3) corresponds to the accumulated price of all resources a flow f uses. It is therefore referred to as its *path-price* and denoted by λ_f.

We observe that the last term in (3) is constant for a given $\boldsymbol{\mu}$ and does not influence the optimal solution $\boldsymbol{x}^*$. We can therefore neglect it without changing the problem. For clarity, we restate the Lagrangian dual function in its new form:

$$\theta(\boldsymbol{\mu}) = \max_{\boldsymbol{x} \in \boldsymbol{X}} L(\boldsymbol{x}, \boldsymbol{\mu}) = \max_{\boldsymbol{x} \in \boldsymbol{X}} \sum_{f \in F} U_f(x_f) - \sum_{f \in F} x_f \lambda_f \tag{6}$$

Thus, the problem is decomposed into two separate problems, the *subproblem* (6), which aims at finding an optimal allocation given the current clique prices $\boldsymbol{\mu}$ and the Lagrangian dual problem (5) for finding the optimal prices for the cliques. From optimisation theory it is known that if the utility functions are concave, then it holds that at an optimal solution $\boldsymbol{\mu}^*$ to (6), $\theta^* = z^*$ and the optimal solution $\boldsymbol{x}^*$ to (2) satisfies $\boldsymbol{x}^* \in \arg\max_{\boldsymbol{x} \in \boldsymbol{X}} L(\boldsymbol{x}, \boldsymbol{\mu}^*)$. In other words we can obtain a solution to our initial problem (2) by solving the problems (5) and (6) . Furthermore, μ_q^* is the *shadow price* of clique q, which is defined as the increase in system utility if we were allowed to increase its available bandwidth by one unit.

2.2. The gradient projection algorithm

Xue, Li and Nahrstedt [12] proposed the application of the gradient projection algorithm (GPA) to the Lagrangian dual function. 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 our case, a gradient at $\theta(\mu)$ is given by $\sum_{f \in F} r_{qf} x_f - C_q, q \in Q$ and thus each component can be calculated separately on a given clique q, 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 $r_{qf} = 0$). Hence the problem can be solved in a distributed manner without resorting to any global information about the network.

It is shown in [12] that for a given set of prices, a unique optimal solution is obtained by letting $x_f = [\frac{d}{dx} U_f]^{-1}(\lambda_f)$. Algorithm 1 summarises the basic steps for the rate allocation.

Algorithm 1 The gradient projection allocation algorithm (GPA)

{At clique q and time t}
$\mu_q^t \leftarrow [\mu_q^{(t-1)} - \gamma(C_q - \sum_{f|E(f) \cup V(q) \neq \{\}} x_f^t r_{qf})]^+$

{At source node of flow f and time t }
$\lambda_f \leftarrow \sum_{q \in Q} \mu_q^t r_{qf}$ {Calculate path-price}
$x_f^t \leftarrow [\frac{d}{dx} U_f]^{-1}(\lambda_f)$ {Calculate rate allocation}

2.3. The adhoc-TARA algorithm

In contrast to the previously discussed (continuous) utility functions, in adhoc-TARA it is assumed that the utility function is discrete; the user specifies the conceived utility using a (small) fixed set of possible allocation points. This corresponds to business models that allow differentiated rates for different quality of service, and charge the user in accordance with the QoS delivered.

To make the problem computationally tractable, the algorithm uses the piece-wise linear, concave utility function given by the convex hull of these discrete points, say $L(x)$. It is even possible to linearise the problem completely. Let the kinks of $L(x)$ have coordinates $\{(B_1, U1), (B_2, U2), \ldots, (B_n, U_n)\}$ (cf. Figure 2). Instead of considering one flow with utility function $L(x)$ we can decompose a flow into $n-1$ *subflows* where each sub-flow corresponds to a linear segment. That is, sub-flow i has a linear utility function $U(x) = B_i + ux$, where

$$u = \frac{U_{i+1} - U_i}{B_{i+1} - B_i} \tag{7}$$

is the slope of the ith segment. In context of our optimisation problem, the sub-flows are treated as independent flows. We have thus obtained a problem formulation where all the flows are linear. The optimisation problem (2) therefore becomes an instance of the linear programming problem (LP) and the Lagrangian dual problem (5) corresponds then to its LP-dual. It is easy to see that due to concavity of $L(x)$, any optimal solution to the linearised problem is also an optimal solution to the original problem [6].

Adhoc-TARA's optimisation method is motivated by the concept of shadow prices as defined in section 2.1. The idea is to approximate the shadow price at a clique by letting the flows bid for using its resources. Such an "auction" should come close to the true value of the clique (its shadow price), which in turn is known to be the solution to the LP-dual.

In adhoc-TARA, the flows construct the bids as follows:

$$\text{bid}_{qf} = \mu_q + \frac{u_f - \lambda_f}{\sum_{q \in Q} r_{qf}} \tag{8}$$

where u_f is the slope of the linear utility function (see Equation 7).

A clique q considers all the bids and allocates bandwidth to the flows in a highest bid order, until the capacity is exhausted. The bandwidth allocation to a sub-flow is then the minimal allocation of all the cliques it traverses. The price of a clique is set to the lowest bid that was accepted. Through this bidding scheme we achieve an approximation of the shadow price and optimise allocation in a completely distributed way.

Note that the algorithm allocates either the maximum bandwidth demanded by a sub-flow or none, as the utility is specified only at these allocation points. This might however lead to some residual bandwidth of a clique remaining unused.

A thorough description of adhoc-TARA is given by Curescu and Nadjm-Tehrani [6]. Algorithm 2 summarises the computations performed in adhoc-TARA.

Algorithm 2 Allocation Algorithm adhoc-TARA

{at clique q and time t}
for all $f \in F$ **do**
  {Calculate bid for flow f}
  $\text{bid}_f = \mu_q^{(t-1)} + \frac{u_f - \lambda_f^{(t-1)}}{\sum_{q \in Q} r_{qf}}$
end for
avBw $\leftarrow C_q$ {Initialise the avail. bandwidth}
{Process flows in order of increasing bids}
$F^s \leftarrow \text{sort}(F)$
for all $f \in F^s$ **do**
  {Allocate until bandwidth exhausted}
  if avBw $> M_f r_{qf}$ **then**
    $x_{fq} \leftarrow M_f$
    avBw $\leftarrow$ avBw $- M_f r_{qf}$
  else
    break
  end if
end for
$\mu_q^t \leftarrow \min_{\{f \in F\}} \{\text{bid}_f | x_{fq} > 0\}$
send μ_q^t and x_{fq} to source node of flows $f \in F$

{At source node of flow f and time t}
$x_f^t \leftarrow \min_{\{q \in Q\}} \{x_{fq}\}$
$\lambda_f^t \leftarrow \sum_{q \in Q} \mu_q^{(t-1)} r_{qf}$ {Calculate path-price}
send λ_f^t to cliques flow f traverses

3. Convergence properties

Although the above algorithms are similar in their goals there are some crucial differences between them. For example, adhoc-TARA considers only the bids at the clique when prioritising among the bidders. Further it always allocates bandwidth until the clique capacity is exhausted. A sub-flow might therefore be dropped at a clique, with an abrupt consequence for the involved connection. On the other hand, the bandwidth for GPA is adapted smoothly. This is also the case when a rapid change occurs, as for example a new connection is added or a mobile node enters a new clique and hence contributing to a possible congestion. As a result, a clique can be under- or overutilised until the optimum is reached.

In GPA the allocated rate of a connection is a strictly decreasing function of the path price. Hence changes in the path price always cause changes in the allocated bandwidth for a connection. On the other hand, adhoc-TARA's bidding mechanism makes it less sensitive to price changes. Changes in the path-price do not necessarily imply changes in the allocation. This is in particular an advantage when dealing with connections that are sensitive to fluctuations of the allocated bandwidth.

To illustrate the basic functioning of the algorithms we apply them to the simple scenario depicted in Figure 1. As a utility function we use the logarithmic 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 2.

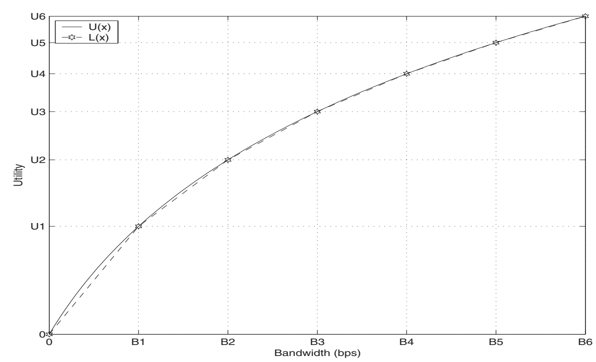
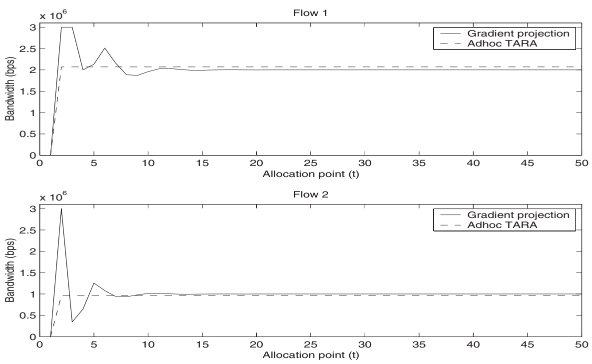


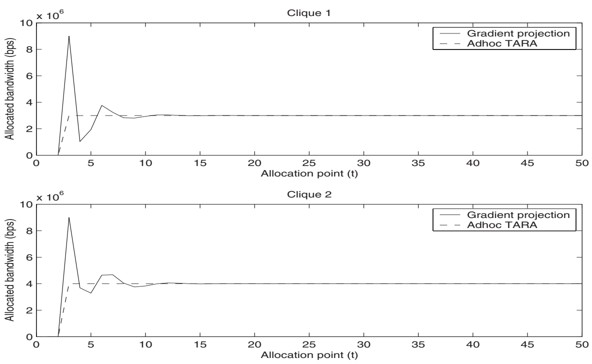
Figure 2. A logarithmic utility function is interpolated to obtain a piecewise linear function

The required bandwidth of both flows is $3Mbps$. The resource capacity C_q is set to $4Mbps$. As the flows contend for the link $(3, 4)$ there is no feasible allocation which respects both flow's required bandwidth.

Figure 3 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. It is known that GPA converges to the (unique) optimal solution, but that adhoc-TARA's allocation is slightly suboptimal. On the other hand, we see that GPA needs a few iterations to converge to the optimal solution, while TARA reaches the near-optimal allocation at once. Figure 3(a) illustrates that in an optimal solution the allocated bandwidth for flow 2 is much smaller,



(a) Connection bandwidth



(b) Clique bandwidth

Figure 3. Bandwidth allocation for the flows and cliques in Figure 1

despite the fact that both flows have the same utility function. This allocation intuitively makes sense, as flow 1 uses fewer resources of the network per unit of bandwidth and hence a larger utility can be achieved by favouring it. Mathematically, this is reflected in the lower path-price of the flow, leading to a higher allocation. In Figure 3(b) the bandwidth allocation at the two cliques is depicted. We 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.

It is clear that in any static network where the connections are sufficiently long living the outcome will be similar to this simple case, and GPA will eventually perform somewhat better than adhoc-TARA.

4. Overloads and Mobility

We have seen that GPA's allocation converges to the optimal solution. However, the above scenario is very static and does not represent a typical adhoc network. In the following we consider a larger network of 60 nodes under different load levels and mobility. Our goal is to see how the convergence time of GPA influences the results in cases where the state of the network is frequently changing. In particular we are interested to know how GPA compares to adhoc-TARA which converges much faster. Further, we investigate how the dissemination time influences the results and study the influence of the step-length in GPA as these are our main parameters to tune the algorithms.

4.1. Simulation setup

We perform the experiments using a simulator based on J-Sim [1]. The simulation environment is synchronous, that is, the bandwidth allocation is performed after a fixed time interval, in which all necessary information for making an allocation decision is computed. For routing we use an on-demand shortest path algorithm, where the length of a path corresponds to the number of hops. Connections are represented as flows, rather than at packet-level. This approach was chosen due to ease of extension of J-Sim for comparative purposes. Thus, some characteristics like packet-level overhead are not studied in this context and the obtained results should be interpreted accordingly.

4.1.1 Baseline

To set the results in context, we compare them to two other algorithms, a simple greedy algorithm and an optimal algorithm having global information. Both algorithms solve the linearised formulation discussed in section 2.3.

The greedy algorithm is obtained as follows: The bid construction in adhoc-TARA could be simplified so that the bid is set to the slope of the flow's utility function divided by the number of links of the clique it uses, i.e. $\text{bid}_{qf} = u_f / R_{qf}$. This corresponds to a simple greedy strategy, where at each clique the flows that use the granted bandwidth most "efficiently" are accepted. Note that no path-prices need to be calculated. Thus, the bid is independent of congestion at other cliques in the flow's path. Another immediate consequence is that the length of the flow path will play no role in the allocation although longer flows consume more resources.

For the global optimal algorithm, we solve the linear program (2) with the LP-solver GLPK [9]. This algorithm is in sequel referred to as "global".

4.1.2 Utility functions and accounting

The gradient projection method requires the utility functions to be strictly concave and differentiable while for adhoc-TARA they have to be linear. We adopt the following strategy to make the results comparable. The utility function is specified as $U(x) = a\log(bx + c)$, where a, b, c are arbitrary parameters that control its shape. This function is linearly interpolated at 7 equidistant points so as to get a piecewise linear utility function consisting of 6 segments, say $L(x)$ (cf. Figure 2).

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

For GPA it happens during overload situations, that more bandwidth is allocated than resources are available. In real networks this would lead to packet drop and probably retransmission. We simulate 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 network is not accounted for.

4.1.3 Traffic model

In the following we will use a traffic mix which combines connections with different bandwidth demands and utility functions. We define six different traffic classes as depicted in Figure 4. 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 aim of having these classes is to mimic typical requirements of real world applications (refer to [7] for a more detailed discussion).

New connections are created at random, according to an exponential distribution with given interarrival rate, while the connection classes are chosen uniformly at random. In a realistic setting the utility function for some of the connections (for example voice service) would be rigid, meaning that zero utility would be accounted if its allocation was ever decreased below a given threshold. For simplicity we treat them in our simulation as flexible connections and apply the utility accounting described above.

4.1.4 Simulation Parameters

Unless otherwise indicated, the following parameters have been used for all the simulations.

Parameter	Value
Period time, τ	0.02 s
Simulation duration	600 s
Connection interarrival rate (per node)	$1/600\ s^{-1}$
Mobility model	Pseudo random waypoint
Simulation area	1500 m $\times$ 1500 m
Transmission range	250 m
Traffic model	see Section 4.1.3
Step-length γ (GPA)	1e-10

To ensure high connectivity and keep the mobile nodes from clumping together, we modified the random waypoint model such that nodes move away from each other when they come closer than a third of their transmission range.

For all our experiments we use the initial network topography of 60 nodes whose location is chosen uniformely at random.

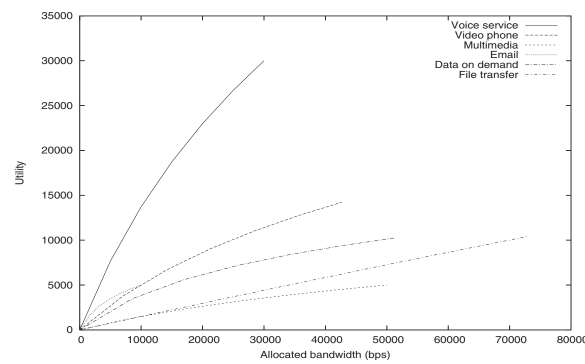


Figure 4. The traffic mix used for subsequent experiments

4.2. Results

The following experiments illustrate how increasing load and mobility affects the accumulated utility. Further the dependence on the dissemination time and step-length (for GPA) is illustrated. We use the accumulated utility as the main metric in all our experiments.

4.2.1 Network load

Figure 5 shows the accumulated utility under the given traffic model under different network loads for a static topology. Higher loads are generated by increasing the arrival rate of the connections. Although the differences are not large, it can be seen that while for light load GPA yields very good results, its performance deteriorates under higher loads. This is explained by the fact that the shorter the interarrival time, the larger the fraction of time GPA is in a non-optimal state. Surprisingly, adhoc-TARA's performance is even under higher loads close to the optimum.

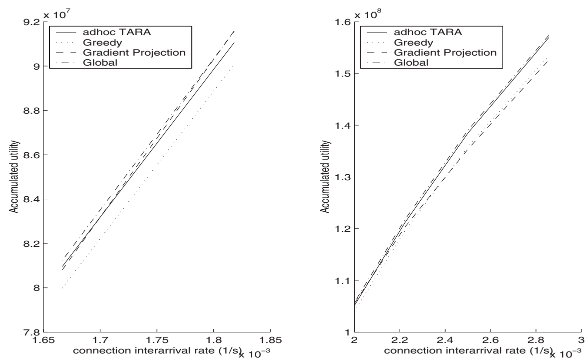


Figure 5. The accumulated utility as a function of the network load

We can gain more insight by looking closer at how the different algorithms allocate the bandwidth. Figure 6 shows the allocated resources at an arbitrary clique during some time period. We observe that the allocation for GPA is generally close to the optimal (global) allocation. However, until it converges it uses more resources than are actually available, leading to suboptimal allocation as some connection's allocated bandwidth is dropped (compare section 4.1.2). We can see in the same figure, that at around allocation point 1800 GPA starts to oscillate. This is an indicator that we did not choose the step-length small enough to ensure convergence for so many flows. This is easily remedied by decreasing the step-length, however at the expense of slowing down convergence under lighter load. Adhoc-TARA is also generally close to the optimal allocation. Yet it can be seen that between allocation point 1000 and 1200 significantly more bandwidth is used at this clique, at the expense of another clique. Note that this gives us no information about how the the accrued utilities differ.

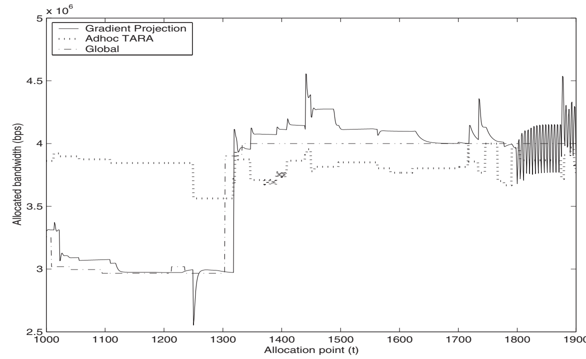


Figure 6. Allocated bandwidth at a sample clique (interarrival rate $1/200s^{-1}$)

4.2.2 Mobility

We proceed by studying the influence of mobility on the accumulated utility. One of the difficulties under mobility is that nodes join and leave the interference range of other nodes and 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 7 clearly confirms our expectation.

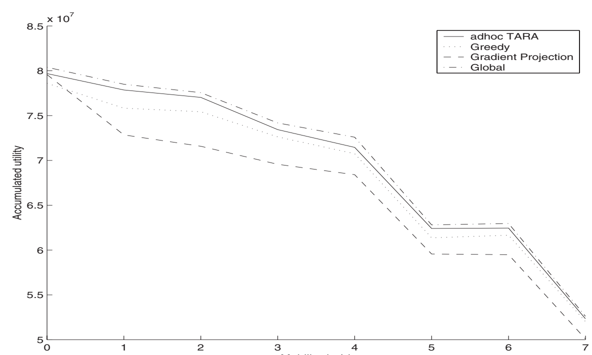


Figure 7. The effect of mobility on the accumulated utility

4.2.3 Period time

As we have already mentioned above, the period time (τ) in our synchronous simulation is of crucial importance for GPA. In fact it is clear that the accumulated utility is monotonically increasing as τ goes to zero. Figure 8 shows how the accumulated utility is affected by the period-time. 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. In a real network, the period depends on the transmission time of the packets and 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.

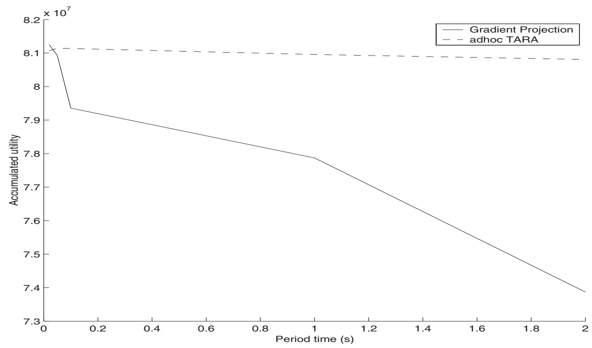


Figure 8. Influence of the period time on the performance

4.2.4 The influence of the step-length

Choosing the right step-length (γ) is crucial for the performance of GPA. Xue et al. [13] 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 case $\overline{\kappa}$ 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. 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. The optimal step-length depends on the traffic type, as illustrated in Figure 9. The simulation was performed with two different traffic models, namely mixed traffic as discussed in section 4.1.3 with an interarrival rate of 1/600, and simulations where the connections are restricted to allow only file transfers (see Figure 4). The interarrival rate in the latter case is 1/1000.

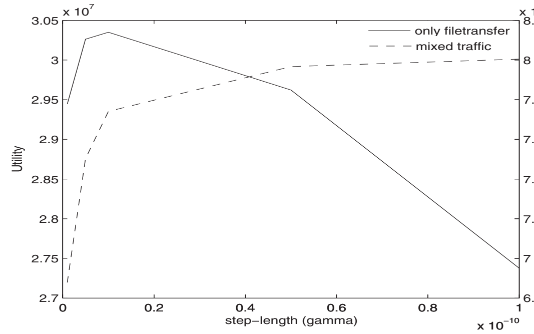


Figure 9. Influence of step-length in GPA on the accumulated utility for two different traffic mixes

5. Conclusions and future work

We have performed experiments to compare two algorithms for optimised resource allocation. Our experiments confirm that even in large networks the gradient projection algorithm converges to an optimal solution, given the step-length is chosen appropriately, whereas for adhoc-TARA the allocation is sometimes non-optimal. The experiments also showed that the accumulated network utility is not necessarily higher for the gradient projection algorithm as it goes through a sequence of suboptimal allocations before converging to the optimum.

Due to the shared transport medium and mobility of the nodes, abrupt changes in load are more frequent in ad hoc networks than in fixed infrastructure networks. Therefore situations where the flows have to be adapted are many and it is crucial that a near-optimal state is quickly reached. The number of iterations required for the gradient projection algorithm to converge to an optimal solution is greatly influenced by the chosen step-length. It was however illustrated that if the traffic-pattern is not known in advance, it is hardly possible to tune the step-length to ensure optimal convergence in all situations. Therefore the period time, that is the time needed to propagate prices of the cliques to the source nodes of the flow, becomes crucial. The smaller this delay is, the better the algorithm's performance. It is clear that fast propagation time is traded-off against signalling overhead.

In contrast, adhoc-TARA's performance is less dependent on finely tuned parameters and network characteristics. Therefore in highly dynamic, mobile networks adhoc-TARA is an adequate algorithm. Although the allocations are not optimal, its performance is in our tests comparable to GPA under low overload and mobility. Under high overload or mobility adhoc-TARA performs often better than GPA. This is mainly due to the fact that at any allocation point a solution close to the optimal is attained immediately.

The above studies have provided interesting insights in the behaviour of both algorithms under dynamic network conditions. However, the conclusions are only valid within the abstract setting of "above packet level" simulations. Both algorithms may exhibit very different behaviour if the overhead in terms of signalling and lost package influence are taken account of. A deeper study of such overheads is therefore an interesting area for future works.

6. Acknowledgements

The work has been supported by the Swedish National Graduate School in Computer Science (CUGS) and Center for Industrial Information Technology (CENIIT) at Linköping University. We would also like to thank Aleksandra Tesanovic for her careful reading of the manuscript and the insightful comments.

References

[1] J-Sim Homepage *http://www.jsim.org*.

[2] A. Boukerche. Performance evaluation of routing protocols for ad hoc wireless networks. *Mobile Networks and Applications*, 9(4):333–342, Aug 2004.

[3] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *MobiCom '98: Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, pages 85–97, New York, NY, USA, 1998. ACM Press.

[4] Z. Cai, M. Lu, and X. Wang. An end-to-end bandwidth allocation algorithm for ad hoc networks. *Telecommunication Systems*, 22(1–4):281–297, Jan 2003.

[5] M. Chiang, D. Oneill, D. Julian, and S. Boyd. Resource Allocation for QoS Provisioning in Wireless Ad Hoc Networks. In *Global Telecommunications Conference,*, volume 5, pages 2911–2915, 2001.

[6] C. Curescu and S. Nadjm-Tehrani. Price/utility-based optimized resource allocation in wireless ad hoc networks. In *Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, Sept. 2005.

[7] C. Curescu and S. Nadjm-Tehrani. Time-aware Utility-based Resource Allocation in Wireless Networks. *IEEE Transactions on Parallel and Distributed Systems*, 16(7):624–636, 2005.

[8] S. R. Das, C. E. Perkins, E. M. Royer, and M. K. Marina. Performance comparison of two on-demand routing protocols for adhoc networks. *Personal Communications, IEEE*, 8(1):16–28, Feb 2001.

[9] GLPK - GNU Linear Programming Kit *http://www.gnu.org/software/glpk*.

[10] F. Kelly, A. Maulloo, and D. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[11] Y. Qiu and P. Marbach. Bandwidth allocation in ad hoc networks: a price-based approach. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, 2003.

[12] Y. Xue, B. Li, and K. Nahrstedt. Price-based resource allocation in wireless ad hoc networks. In *Eleventh International Workshop on Quality of Service (IWQoS 2003)*, volume 2707/2003 of *Lecture Notes in Computer Science*, pages 79–96, 2003.

[13] Y. Xue, B. Li, and K. Nahrstedt. Optimal resource allocation in wireless ad hoc networks: A price-based approach. To appear in *IEEE Transactions on Mobile Computing*, 2005.