

# Flow Allocation in Wireless Networks with Selfish Nodes

A.E. Krzesinski

Department of Mathematical Sciences

Stellenbosch University, 7600 Stellenbosch, South Africa

Email: aek1@cs.sun.ac.za Tel.: +27 21 808 4232 Fax: +27 21 882 9865

**Abstract**—Consider an ad hoc network where packet transmissions occur between the nodes. Optimal flow allocation in such systems can be modelled as a constrained nonlinear optimisation problem. This problem can be solved either by standard methods which assume global knowledge of the system being modelled, or by a distributed algorithm which assumes local knowledge only.

We consider an ad hoc network which contains *selfish* nodes. A selfish node cares only about maximising its own flows and does not care about the utility that any other nodes get. Flow allocation in a network of altruistic and selfish nodes can be modelled as a constrained nonlinear optimisation problem and solved by standard methods. However, in this case a dynamic algorithm to compute the network flows is not available.

We modify the behaviour of the selfish nodes so that a dynamic solution is possible. In this scheme, selfish nodes advertise false (inflated) resource prices to the other nodes. These nodes respond by not routing their flows through the selfish nodes, and the selfish nodes can now use all their resources to transmit their own flows. The flows return to their optimal values if the selfish nodes subsequently advertise the correct prices for their resources.

Altruistic nodes can detect the inflated prices charged by the selfish nodes and respond by advertising false (inflated) prices to the selfish nodes. In this case the flows originating at the selfish nodes are reduced, but the flows do not return to their optimal values. This scheme also has a distributed solution.

**Keywords**—Ad hoc networks, dynamic algorithms, flow allocation, Lagrangian optimisation, selfish users.

## I. INTRODUCTION

Ad hoc networks are self-configuring networks of mobile nodes connected by wireless links. They enable infrastructure-free communication: no fixed equipment is needed, instead each node acts as a router. Although a node benefits from transmitting and receiving its own flows, a node does not benefit from forwarding traffic on behalf of other nodes. However, if the nodes do not act as transit nodes, then the ad hoc network would fail to function. Therefore a crucial question is: how can the nodes be given incentives to act as transit nodes?

This question has received much attention in the literature (see [3] and the references therein). Crowcroft *et al.* [1] present

---

This work is based on research supported in part by the National Research Foundation of South Africa (Grant specific unique reference number (UID) 83965), Nokia-Siemens Networks and Telkom SA Limited. The author thanks the Australian Research Council Centre of Excellence in the Mathematics and Statistics of Complex Systems (MASCOS) for financial support.

such an incentive scheme where each node has a credit balance that determines how much the node can spend on transmission resources in the next time interval. For each node there are two resources: bandwidth and power, each with its own price. The price of each resource increases when the resource is scarce, and decreases when the resource is abundant. The resource prices determine the cost of sending a unit of flow end-to-end along a route, which in turn determines the flow along the route. The flows that originate, terminate and transit at a node determine the price of resources at that node. The interplay between the prices, flow allocations, and credit balances is such that global stability of the system is achieved. Importantly, such a scheme is decentralized: no central controller is needed, and the scheme therefore has favourable scalability properties [2].

The remainder of this paper is organized as follows. Section II presents a simple mathematical model [3] for flow allocation in an ad hoc network. Under the assumption that every node is willing to cooperate, [3] derives the Karush-Kuhn-Tucker equations that define the *socially-optimal flows* on each of the routes. Section III defines a model [3] where a selfish node, the egoist, cares only about maximising its utility, and solves for the optimal flows in this model.

The flow allocation problem in networks of altruistic nodes can be solved by both centralised and by distributed methods. Centralised methods require global knowledge of the system to be solved, distributed methods need only local knowledge. Flow allocation in a network containing both altruistic and selfish nodes can be modelled as a constrained nonlinear optimisation problem and solved by standard methods.

In Section IV we modify the behaviour of the selfish nodes so that a distributed solution is possible. In this scheme the selfish nodes advertise false (inflated) resource prices to the other nodes. As a result, the altruistic nodes do not route their flows through the selfish nodes, and the selfish nodes now can use all their resources to transmit their own flows. The selfish nodes benefit by being able to send larger flows, the altruistic nodes are disadvantaged in that their flows are reduced, and the network experiences a lower total utility.

Suppose the altruistic nodes can detect the inflated prices charged by the selfish nodes. Suppose further that the altruistic nodes respond to the selfish nodes by advertising false (inflated) prices to the selfish nodes while advertising correct prices to the altruistic nodes. This will cause the flows originating from the selfish nodes to be reduced, but the network flows will not necessarily return to their socially-optimal values. This scheme has a distributed solution. Section V demonstrates

the usefulness of the distributed solution approach, where we define a time-varying utility function and solve for the time-varying resource prices. Conclusions are presented in Section VI.

## II. A MODEL FOR FLOW ALLOCATION

Consider a simple model where the positions of all nodes are fixed, and where the path chosen for communication between any origin and destination is also fixed. The model assumes that the nodes communicate with each other by transmitting continuous ‘flows’. In reality, flows are made up of discrete packets. However, for situations where the packet size is small compared to the total volume of data transferred, the continuous model can be appropriate. Refer to [3] for a complete description of the flow allocation model.

The network model consists of  $j \in J$  nodes. Each route  $r \in R$  is defined by an ordered subset of nodes. Route  $r$  has source node  $s(r)$ , destination node  $d(r)$  and a set  $t(r)$  of transit nodes. Let  $R^S(j)$ ,  $R^D(j)$  and  $R^T(j)$  denote the sets of routes for which  $j$  is the source, the destination or a transit node respectively. Let  $\nu_s$ ,  $\nu_d$  and  $\nu_t$  denote the rate at which battery energy is consumed per unit flow at node  $j$  when  $j$  is a source node, a destination node and a transit node respectively. Let  $P_j$  denote the maximum power (the rate per unit time at which battery energy can be used) at node  $j$ .

Let  $U_r(y)$  denote the utility derived by the originating node when a flow  $y$  is transmitted on route  $r$ . The socially-optimal flows  $\mathbf{y} = (y_r)_{r \in R}$  are found by solving the optimisation problem

$$\max_{\mathbf{y}} \sum_{r \in R} U_r(y_r) \quad (1)$$

subject to the constraints  $y_r \geq 0$  for  $r \in R$  (the flows are non-negative) and

$$\nu_s \sum_{r \in R^S(j)} y_r + \nu_d \sum_{r \in R^D(j)} y_r + \nu_t \sum_{r \in R^T(j)} y_r \leq P_j$$

for  $j \in J$  (the flows originating from, terminating at and transiting through node  $j$  cannot use more than  $P_j$  units of power at node  $j$ ).

Eqn. (1) involves the maximization of a strictly concave function over a convex region and so there exists a unique solution.

The Karush-Kuhn-Tucker (KKT) conditions for optimality yield  $2R + J$  equations for the flows  $y_r$ , the Lagrange multipliers  $\eta_r$  for the flow constraints, and the Lagrange multipliers  $\xi_j$  for the power constraints. The  $2R + J$  simultaneous non-linear equations can be solved for example by using the code provided in [6] or by using the NLP package CFSQP [4]. These are centralized solution methods. Alternatively, we can design a dynamic algorithm to solve the KKT equations: this is a distributed solution method.

### A. A distributed solution of the social optimum

A dynamic algorithm can be designed to derive the socially-optimal flows based upon the KKT conditions. This algorithm has the advantage in that it is distributed in the sense that a source node only needs to know the values of the

Lagrange multipliers at the nodes along a route. The algorithm was presented in [3] and was adapted from the distributed dynamic approach of Crowcroft *et al.* [1]. First, compute the power in use at node  $i$  at time  $t - \Delta$

$$p_i(t - \Delta) = \nu_s \sum_{r \in R^S(i)} y_r(t - \Delta) + \nu_d \sum_{r \in R^D(i)} y_r(t - \Delta) + \nu_t \sum_{r \in R^T(i)} y_r(t - \Delta) \quad (2)$$

where  $y_r(t - \Delta)$  is the flow on route  $r$  at time  $t - \Delta$ . Next, the price  $\xi_i(t)$  of power at node  $i$  at time  $t$  is given by the DE

$$\frac{d}{dt} \xi_i(t) = \kappa \xi_i(t) \frac{p_i(t) - P_i}{P_i} \quad (3)$$

where  $\kappa$  is a constant and  $P_i$  is the power available at node  $i$ . A discretised version of the DE (3) yields

$$\xi_i(t) = \xi_i(t - \Delta)(1 - \Delta\kappa(1 - p_i(t - \Delta)/P_i)). \quad (4)$$

The cost of sending a unit of flow along route  $r$  at time  $t$  is

$$\omega_r(t) = \nu_s \xi_{o(r)}(t) + \nu_d \xi_{d(r)}(t) + \nu_t \sum_{j \in t(r)} \xi_j(t). \quad (5)$$

Finally, the flow  $y_r(t)$  on route  $r$  at time  $t$  (given a logarithmic utility function) is

$$y_r(t) = 1/\omega_r(t). \quad (6)$$

Repeated application of Eqns (2) through (6), starting from  $y_r(0) = 0$  for all  $r \in R$  and  $\xi_j(0) = 1$  for all  $j \in J$ , causes the dynamical system  $(\mathbf{y}(t), \boldsymbol{\xi}(t))$  to converge to the solution of the KKT equations, which was demonstrated in [3].

Note that node  $i$  requires a knowledge only of its own power usage to update its price  $\xi_i(t)$  according to Eq. (4) and route  $r$  requires only a knowledge of the prices of power at the nodes along route  $r$  to set its flow according to Eqns. (5) and (6). Thus, the algorithm can be implemented without the nodes having to know global information.

### B. Some examples of the distributed solution

An important property of the dynamical solution is that the system can adapt to changes to the network topology. Consider a network model [1] with ten nodes located at random on a  $100m \times 100m$  plane. Two nodes are connected by a link if they are within  $50m$  of each other. The network has 50 unidirectional links. Each node has a battery of power  $0.5W$ . Simulations [5] were run in which new users were introduced to the network part-way through the simulation. In all cases, it was found that the system successfully adapted and converged to the new optimal solution. Fig. 1 illustrates a simulation in which a new user is introduced to the network after 1000 iterations. The system quickly adapts and converges to the new optimal solution.

The success of the dynamic solution depends on each node being able to communicate its price signal to other nodes. There may be a time delay associated with sharing information within the network. Simulation experiments were run to test the system's sensitivity to delays in communicating the price signals. This was done by updating the flows using the

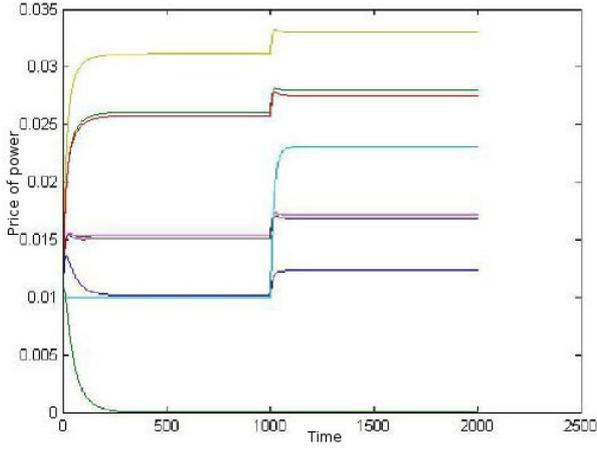


Fig. 1. Price adjustment when a new node arrives to the network [5].

route costs computed  $\tau$  iterations previously so that Eqn. (5) becomes

$$\omega_r(t) = \nu_s \xi_{s(r)}(t - \tau) + \nu_d \xi_{d(r)}(t - \tau) + \nu_t \sum_{j \in t(r)} \xi_j(t - \tau).$$

Fig. 2 shows that the dynamic solution successfully adapts provided the price delay is less than 12 iterations. See [5] for a further investigation of the dynamic solution method.

### III. THE EGOIST

Now consider the case where node  $i$  (the egoist) is selfish and wants to maximise the utility of its flows. Node  $i$  increases the flows on the routes  $r \in R^S(i)$ . Node  $i$  does not care about the utility that any other node gets. However, node  $i$  cannot influence flows that do not use node  $i$  so these flows remain at their socially-optimal values. The egoist flows  $\mathbf{y} = (y_r)_{r \in R^S(i)}$  are found by solving the optimisation problem [3]

$$\max_{\mathbf{y}} \sum_{r \in R^S(i)} U_r(y_r)$$

subject to the constraints  $y_r \geq 0$  for  $r \in \mathcal{R}$  (the flows are non-negative)

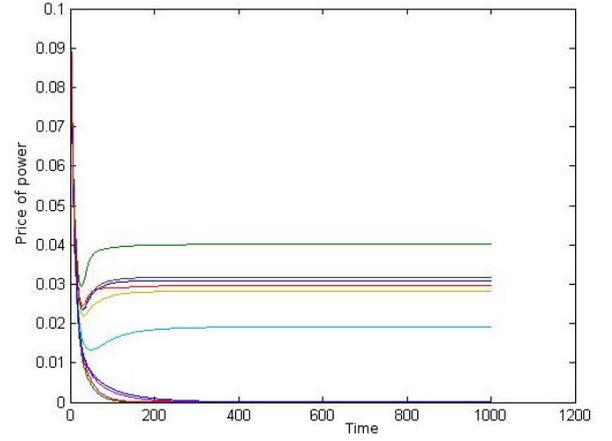
$$\nu_s \sum_{r \in R^S(i)} y_r \leq P_i$$

(the flows originating from the egoist node  $i$  can use at most  $P_i$  units of power at node  $i$ ), and

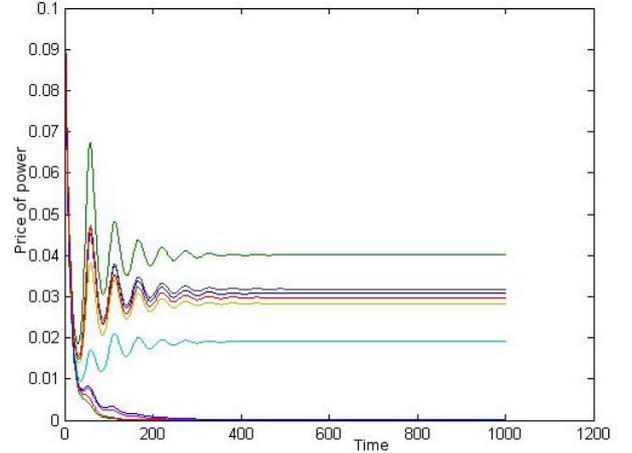
$$\nu_d \sum_{r \in R^S(i) \cap R^D(j)} y_r + \nu_t \sum_{r \in R^S(i) \cap R^T(j)} y_r \leq P_j - C_{ij}$$

for  $j \neq i$  (the flows originating from the egoist node  $i$  can use at most  $P_j - C_{ij}$  units of power at node  $j$ ). The power at node  $j$  available to routes with source node  $i$  is reduced by the socially-optimal power consumption  $C_{ij}$  of routes that do not go through node  $i$ . The egoist  $i$  has no control over these routes.

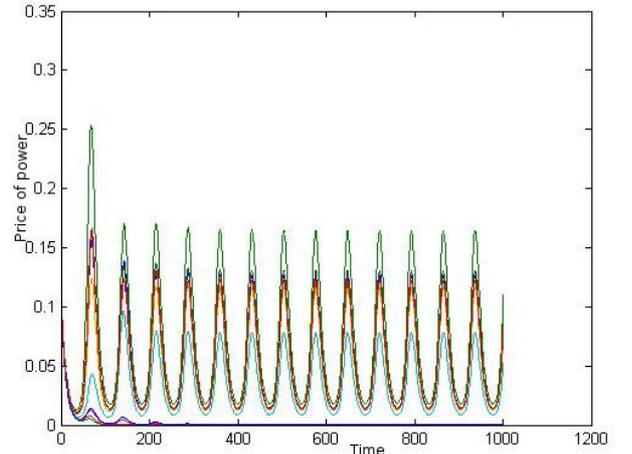
$$C_{ij} = \nu_s \sum_{r \in R^S(j) \setminus R(i)} y_r^{(s)} + \nu_d \sum_{r \in R^D(j) \setminus R(i)} y_r^{(s)} + \nu_t \sum_{r \in R^T(j) \setminus R(i)} y_r^{(s)}$$



(a) Time delay  $\tau = 5$



(b) Time delay  $\tau = 12$

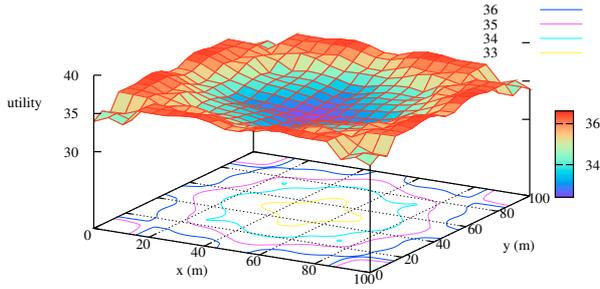


(c) Time delay  $\tau = 17$

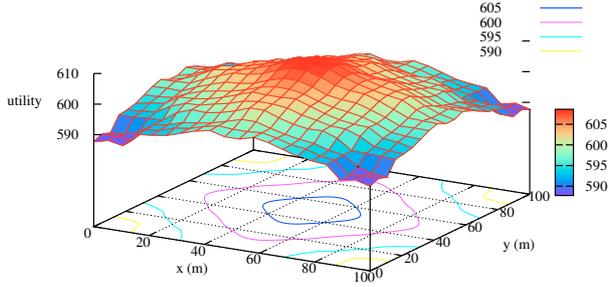
Fig. 2. Price at each node as a function of time [5].

where  $y_r^{(s)}$  is the socially-optimal flow on route  $r$  and  $R(i) \equiv R^S(i) \cup R^D(i) \cup R^T(i)$ .

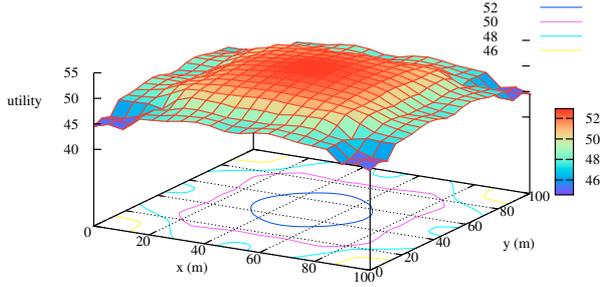
The KKT conditions for optimality yield  $2R + J$  equations where  $R = |R^S(i)|$  for the flows  $y_r$ , the Lagrange multipliers  $\eta_r$  for the flow constraints, and the Lagrange multipliers  $\xi_j$



(a) The socially-optimal utility of the outbound flows at node  $k$  at  $(x, y)$



(b) The total socially-optimal utility when node  $k$  is at  $(x, y)$



(c) The utility of the outbound flows at the egoist node  $k$  at  $(x, y)$

Fig. 3. Utility surfaces for the 16-node network model.

for the power constraints. The  $2R+J$  simultaneous non-linear equations can be solved using the NLP package CFSQP [4] which provides a centralised solution method. We were not able to design a distributed method to compute the egoist flows.

#### A. Utility as a function of location

Consider a network model with 16 nodes. The model assumes a flat terrain of  $100\text{ m} \times 100\text{ m}$  that is partitioned into a  $4 \times 4$  grid of cells. The transmission range is  $D = 50\text{ m}$  and each cell is  $25\text{ m} \times 25\text{ m}$ . One node is placed at the centre of each cell so that the network is connected.

An additional node  $k$  (the tagged node) is successively located at  $(x_k, y_k)$  where  $x_k \in \{0, 5, \dots, 100\}$  and  $y_k \in \{0, 5, \dots, 100\}$ . The routes are fixed. The cost  $C_\ell$  of the link  $\ell$  from node  $i$  to node  $j$  is given by

$$C_\ell = \begin{cases} d_{ij} & d_{ij} \leq D \\ \infty & \text{otherwise} \end{cases}$$

where  $d_{ij}$  is the euclidean distance from node  $i$  to node  $j$ . The utility function is  $U_r(y_r) = \log(y_r)$  where  $y_r$  is the flow on route  $r$ . The other parameters are  $\nu_s = \nu_d = 1$ ,  $\nu_t = 2$  and  $P_j = 0.5$ . The CFSQP NLP solver [4] was used to compute the optimal allocation of the flows.

Fig. 3(a) presents the socially-optimal utility  $\sum_{r \in \mathcal{R}^S(k)} U_r(y_r)$  of the flows originating at node  $k$  when node  $k$  is in position  $(x_k, y_k)$ . The figure also shows the utility contours projected onto the  $(x, y)$  plane.

- The utility of node  $k$  is low when node  $k$  is near a corner of the  $(x, y)$  plane. In this case node  $k$  carries no transit routes. Although the power at node  $k$  is not fully used (the value of the Lagrange multiplier  $\xi_k$  for the power constraint at node  $k$  is zero), it is not optimal to assign large flows to the relatively long routes that originate from node  $k$ .
- The utility of node  $k$  is also low when node  $k$  is near the centre of the  $(x, y)$  plane. In this case node  $k$  carries many transit routes. The power at node  $k$  is fully used (the Lagrange multiplier  $\xi_k > 0$ ) but only a small amount of power is available at node  $k$  for it to send its own flows, most of the power being used to relay flows that transit through node  $k$  and to accept flows that terminate at node  $k$ .

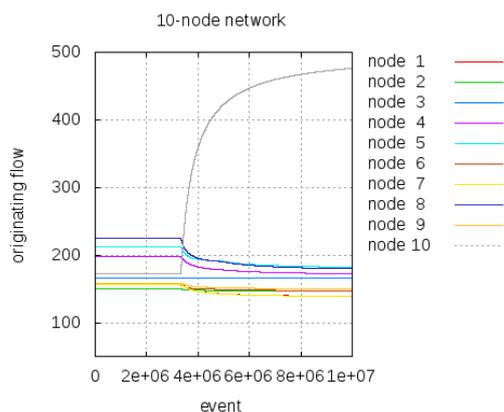
Fig. 3(b) presents the socially-optimal *total* utility  $\sum_{r \in \mathcal{R}} U_r(y_r)$  when node  $k$  is in position  $(x_k, y_k)$

- The total utility is low when node  $k$  is at the corner of the network where it can only originate small flows.
- The total utility is large when node  $k$  is near the centre of the network. In this case node  $k$  provides transit services to many routes thus relieving other nodes near the centre of network from functioning as transit nodes so that they have sufficient power to originate large flows.

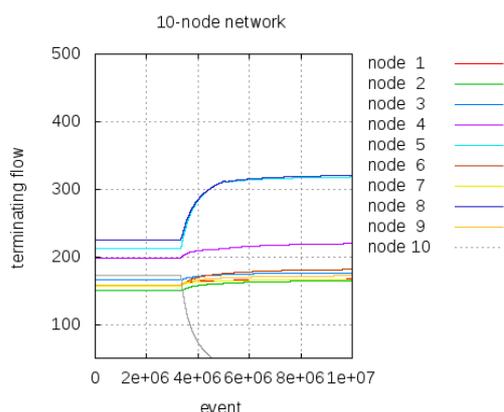
We next examine the case when all nodes are altruistic nodes except node  $k$  which is an egoist node. Fig. 3(c) presents the utility of the outbound flows at the egoist node  $k$ . The utility is large when the egoist node  $k$  is near the centre of the network where it can use all the power at node  $k$  to send large flows on relatively short routes to most destinations.

#### IV. AN EGOIST VARIANT: PRICE MANIPULATION

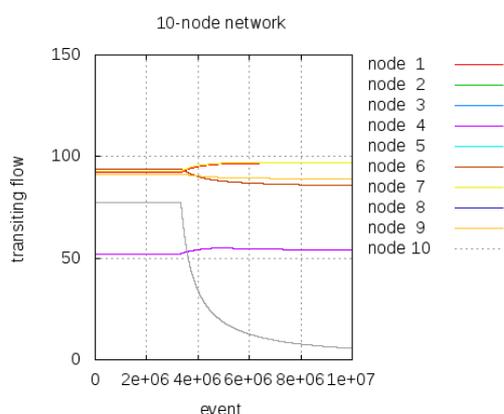
The egoist user can exhibit many different types of selfish behaviour. For example, the egoist could communicate a price of zero to the other nodes. Once the egoist had attracted a large volume of traffic (the extrinsic flows), the egoist could substitute its own flow for its transit flow. Such behavior would leave the prices unaffected at the downstream nodes although the misbehaviour could be detected by downstream nodes observing that the extrinsic flows had been dropped. However, this detection mechanism would not scale as it would require per flow state information be kept at each node. Nodes upstream from the egoist may be unaware that their flows through and to the egoist are being dropped. These flows represent a waste of network resources since they cannot reach their destinations, neither can these resources used by these flows be seized by the egoist.



(a) Originating flows at the egoist node 10



(b) Terminating flows at the egoist node 10



(c) Transit flows at the egoist node 10

Fig. 4. Price manipulation.

#### A. Advertising false prices

The question arises if it is possible for the egoist to disguise its misbehaviour from the other nodes and acquire resources from upstream and downstream nodes. Consider an egoist that communicates an increasingly *false* price to the other nodes in the network so that these nodes are slowly discouraged from using the egoist node. As a consequence of the perceived inflated resource price at the egoist, flows through the egoist node are gradually attenuated as the other nodes are discouraged from using the egoist. The fraud is likely

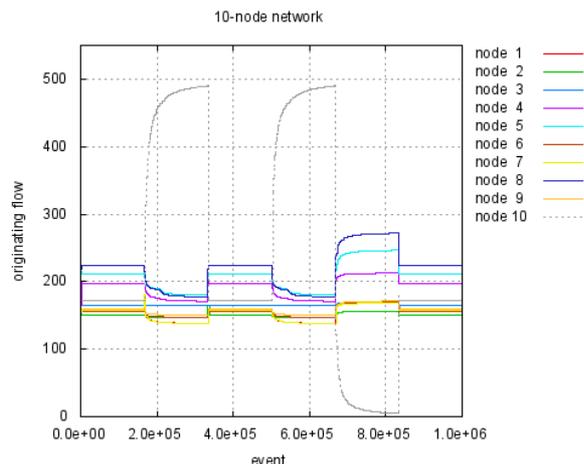


Fig. 5. Price manipulation.

to be undetected. The unused capacity at the egoist node and elsewhere can now be used by the egoist to send its own flows – the egoist uses the correct price when it sends its own flows.

Fig. 4 shows the originating, terminating and transit flows at node 10 which is located at the centre of a 10-node network on a  $100m \times 100m$  plane as introduced above. The utility function is

$$U_r(y_r) = \log(y_r)$$

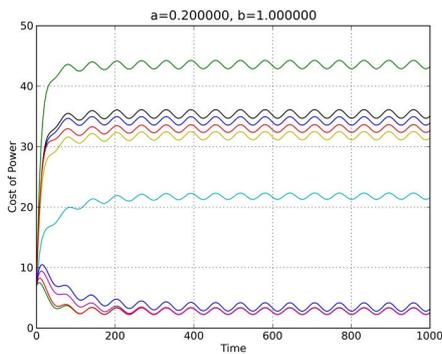
where  $y_r$  is the flow on route  $r$ . The other parameters are  $\nu_s = 1$ ,  $\nu_d = \nu_t = 2$  and  $P_j = 0.5$ . Node 10 initially advertises true resources prices and the flows converge to their socially-optimal values. One third of the way though the simulation node 10 gradually increases its prices and communicates these inflated prices to the rest of the network. Fig. 4(a) shows that the flows originating at the egoist node 10 increase, to the detriment of the originating flows at all the other nodes. Figs. 4(b) and 4(c) show that the flows terminating at and transiting through the egoist node 10 are attenuated.

#### B. Retaliatory pricing strategies

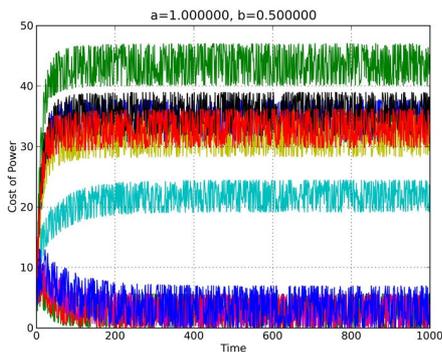
The nodes can deploy reactive strategies to retaliate against nodes that are perceived as misbehaving. Consider the 10-node network introduced above, and let the network move through six stages. Fig. 5 shows that initially when all nodes advertise true resource prices, the flows converge to their socially-optimal values. Next, the egoist node 10 advertises false prices and the egoist benefits to the detriment of the other nodes. Then node 10 advertises true prices and the flows return to their socially-optimal values. Once again, node 10 advertises false prices and the egoist benefits. This time the other nodes retaliate by advertising false (inflated) prices to the egoist. The prices that the altruistic nodes communicate among themselves remain at their true values. The egoist flows are decreased but the flows do not return to their socially-optimal values. Finally all nodes advertise true prices and the flows are restored to their socially-optimal values.

### V. TIME-VARYING PRICES

In this section we investigate the impact of a time varying utility function on the price of the resources in the network.



(a)  $U(y_r(t)) = a \sin(bt) \log(y_r(t))$



(b)  $U(y_r(t)) = (a \sin(bt + 2\pi U(0, 1) + 1) \log(y_r(t))$

Fig. 6. Time dependent utility function.

The 10-node network model of the previous section is used. The flow allocations are computed using the dynamic solution method. The flows could also be computed using the CFSQP NLP solver [4] but the computation would be slow and, depending upon the periodicity of the cyclic variations, might be numerically unstable.

We use a time-varying utility function

$$U(y_r(t)) = (a \sin(bt + 2\pi U(0, 1)) + 1) \log(y_r)$$

where  $U(0, 1)$  is a uniform deviate to introduce phase differences among the flows on the various routes. The phase differences ensure that the route utilities do not all increase or all decrease simultaneously. If  $a = 0$  we have the logarithmic utility function. Increasing the value of  $b$  introduces high frequency oscillations in the utility function.

Fig. 6(a) shows the price of power at the various nodes of the network when  $a = 0.2$  and  $b = 1.0$  with no random phase differences. The flows converge and reflect the sinusoidal variation of the utility function. Fig. 6(b) shows the price of power at the various nodes of the network when  $a = 1.0$  and  $b = 0.5$  with random phase differences. The flows display the effect of the phase-induced jitter that is superimposed upon a sinusoidal variation in the utility function.

## VI. CONCLUSION

Optimal flow allocation in a network of static wireless nodes can be modelled as a constrained nonlinear optimisation problem. This problem can be solved either by standard

methods which assume global knowledge of the system being modelled, or by a dynamic algorithm which assumes local knowledge only. We present such a dynamic algorithm and use it to evaluate the performance of a small 10-node network.

We next consider an ad hoc network which contains *selfish* nodes. A selfish node cares only about maximising its own flows and does not care about the utility that any other nodes get. We modify the behaviour of the selfish nodes so that a dynamic solution is possible. In this scheme, selfish nodes advertise false (inflated) resource prices to the other nodes. These nodes respond by not routing their flows through the selfish nodes, and the selfish nodes can now use all their resources to transmit their own flows. The flows return to their socially-optimal values if the selfish nodes advertise the correct prices for their resources.

Altruistic nodes can detect the inflated prices charged by the selfish nodes and respond by advertising false (inflated) prices to the selfish nodes. In this case the flows originating at the selfish nodes are reduced, but the flows do not return to their socially-optimal values. This scheme also has a distributed solution.

Dynamic solution methods are also valuable in situations where the node positions, their utility functions or other parameters, vary with time. The network adjusts its emitted flows in a decentralised way so that good performance in terms of tracking the overall utility is produced. We finally present several experiments where the dynamic solution method was used to evaluate network models where the utility is a function of time.

## REFERENCES

- [1] J. Crowcroft, R. Gibbens, F. Kelly, and S. Östring. Modelling Incentives for Collaboration in Mobile Ad Hoc Networks. *Performance Evaluation*, 57(4):427–439, 2004.
- [2] J. Göbel, M. Mandjes and A.E. Krzesinski. Incentive-based control of ad hoc networks: a performance study. *Computer Networks*, 53(14), pp 2427–2443, Sep 2009.
- [3] A.E. Krzesinski, G. Latouche and P.G. Taylor. How do we encourage an egoist to act socially in an ad hoc mobile network? *Computer Networks Vol 56 No 15 (Oct 2012)* pp 3499–3510.
- [4] C. Lawrence, J.L. Zhou and A.L. Tits. User's guide for CFSQP version 2.4: a C code for solving constrained nonlinear (minimax) optimization problems generating iterates satisfying all inequality constraints, *Research Report TR-94-16r, Department of Electrical Engineering and the Institute for Systems Research, University of Maryland, College Park, MD 20742 USA (1996)*.
- [5] E. Muir, G. Adams and P. Braunsteins. Localised Analysis of an Ad Hoc Mobile Network. *Technical Report AMSI Summer Vacation Scholarship*, Department of Mathematics and Statistics, University of Melbourne, Australia, Feb 2012.
- [6] W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P. Flannery. *Numerical Recipes: The Art of Scientific Computing*, Third Edition, Cambridge University Press, ISBN-10: 0521880688.

**Anthony Krzesinski** is a Professor of Computer Science at the University of Stellenbosch, South Africa.