Dynamics of a localized reputation-based network protocol

George Karakostas
McMaster University
karakos@mcmaster.edu

Raminder Kharaud
McMaster University
kharaur@mcmaster.ca

Anastasios Viglas
University of Sydney
taso.viglas@sydney.edu.au

Abstract—**We consider a type of game theoretic dynamics in a network model where all nodes act selfishly and will forward packets only if it is to their benefit. The model we present assumes that each node receives utility from successfully sending its own flow to its destination(s) and from receiving flow, while it pays a cost (e.g., battery energy) for its transmissions. Each node has to decide whether to relay flow as an intermediate node from other sources, as relaying incurs only costs. To induce nodes into acting as intermediaries, the model implements a reputation-based mechanism which punishes non-cooperative nodes by cutting off links to them, a decision that is made in a very local fashion. In our setting, the nodes know only the state of the network in their local neighborhood, and can only decide on the amount of the flow on their outgoing edges, unlike the previously considered models where users have full knowledge of the network and can also decide the routing of flow originating from them. Given the opportunistic nature of the nodes and their very limited knowledge of the network, our simulations show the rather surprising fact that a non-negligible amount of non-trivial flow (flow over at least two hops) is successfully transmitted.**

I. Introduction

In wireless networks such as sensor networks, a major concern is the battery life of each node. Each node is transmitting bits using its battery energy, but these transmitted bits have different intrinsic value for the node: bits that are transmitted because the node wants to communicate with another node should count as *gain*, while bits that the node forwards in order to facilitate the communication of other nodes should count as *loss*, since the energy spent for this purpose doesn't serve the *self-interest* of the node. Usually the nodes of a network are seen as part of a greater scheme that is not concerned with their self-interest. In this work we study networks of nodes that act as *selfish agents*, i.e., there is no central coordinator that imposes a certain behavior on the nodes, but each node has its own *decision variables* that decide the nature of its participation in the network, and its own *utility function* which is a measure of the network service to its self-interest (gain or loss). The main motivation for building such networks with selfish nodes is the apparent fact of life that certain agents are willing to participate in a greater scheme (the network) only if they see a reward for their effort. For example, such a network protocol may allow the build-up of a wireless ad-hoc network in a city using citizens' smartphones; unlike a sensor network where the sensors are machines dedicated to a single purpose (and therefore have no self-interests), a smartphone network cannot be enforced upon the phone users; still, it could possible that users who are far away from a WiFi hotspot can use other smartphones to connect to, say, the Internet, through a user who is close to a hotspot, if a network of selfish agents can be built. Unfortunately, the latter may not be possible at all, a situation unlike that of the usual wireless networks with selfless nodes which can always be enforced into being (provided the nodes do not fail, etc.). The possibility of existence for wireless networks with selfish nodes and their performance is the subject of this study.

Although energy concerns are a good motivation for the study of networks with selfish nodes, they may not be the only ones. Therefore we cast the problem we study in a more general framework of network flows. The network itself can be modeled as a directed unweighted graph, where the nodes are the agents and the edges are the *potential* links between agents. Note that for this work, the terms *node* and *agent* are interchangeable. As will become apparent in the following, we call these links *potential* because they may or may not be present in the network, depending on the behavior of the agents adjacent to the link. In this network we have a number of designated Origin-Destination (O-D) pairs of agents; each O-D pair s, t is associated with a positive parameter $d_{s,t}$ which is the transmission rate of the connection that s and t want to establish. We formulate this transmission requirement as a network flow commodity, with source s, sink t and flow demand $d_{s,t}$. Note that each node may be the source and/or the sink of more than one commodities. For simplicity, we will assume that there is only a single commodity associated with a particular O-D pair.

Game-theoretical concepts are suitable for capturing the selfish nature of the agents. A node receives utility from all flow successfully delivered and for which the node is either the origin or the destination. All nodes need to pay a cost which is proportional to the amount of flow they need to transmit, whether they are the origin of that flow, or they simply relay someone else's commodity. In fact, we assume that a node needs to pay this cost even for the amount of flow it attempts to transmit, but never reaches its destination; this captures the fact that transmission (whether successful or not) always requires resources. The total gain (or loss) of a node is its *total utility* (or payoff). Each node controls a set of decision variables that determine its behavior, and therefore there is a range of potential values for these variables that the node can chose; every permissible combination of values the node can chose is a *strategy*. Apparently, each node is going to switch between strategies in its strategy space if such a switch is beneficial, i.e., if its total utility increases. In particular, a strategy that maximizes this utility if we assume that no other node switches at the same time, is called a *best response*. Eventually, the system may reach a state where no such switches, done unilaterally (i.e., assuming that nobody else switches strategies at the same time), can be beneficial; in this case the system has reached a *Nash equilibrium*. This process of reaching such a stable state (i.e., the *dynamics* of the game that models our network problem) is the main subject of this work.

Initially it seems unlikely that even a single node may be willing to collaborate by forwarding flow. Obviously, if an O-D pair for a commodity is connected by a link, then it is to the benefit of both the origin and the destination to go ahead with this communication. If, however, there is no direct connection, then an intermediate node, or several intermediate nodes must be used as relays, or forwarding nodes, and such nodes have no a priori reasons to cooperate (after all, a commodity that they only forward means only additional costs without any benefits). There are two general methods for inducing these intermediate nodes into doing at least some forwarding: (i) they can be rewarded by the O-D pairs they help with some kind of credit, that they in turn can use in order to pay nodes that relay their own commodities, or (ii) they can be punished by their neighbors for not being cooperative enough. In this work we adopt the second approach. Namely, we follow in the line of research of [1] and [2], and we propose a protocol that gives a node the power to essentially cut its link to a neighbor that is not cooperative enough, i.e., blocks more flow than the node can tolerate. We depart from that work by simplifying the strategy space for the nodes: Up to now, each node could decide the routing of the flow for which it was the origin. In order to do that, it had to know the current topology of the whole network (i.e., which links are currently instantiated), which is rather unrealistic, because it requires the continuous gathering of network information by every node all the time. In this work we propose a TCP/IP-like model, in which the nodes collect initially information for the network topology (as before), but they use this topology in order to calculate the shortest-path (in number of hops) for each commodity. These shortest paths are used to populate a routing table for every node that maps commodities to neighbors and indicates which link should be used by the node in order to forward the flow of a particular commodity. After this initial stage of setting up the routing tables, the (game) protocol starts running for each node separately: each node has access only to *local* information, and decides only *local* variables, i.e., it decides only what happens to the edges incident to it, and to the flow it sends/forwards/ on these edges. This localized approach greatly simplifies the protocol, is more realistic, and makes the computation of the best response by a node much simpler. The price we pay for simplicity and practicality is the lack of necessary and sufficient conditions for the existence of *non-trivial* equilibria, i.e., equilibria other than the equilibrium where the only commodities that are transmitted are the ones with origin and destination both incident to a link; [1] and [2] stated and took advantage of such conditions to study the existence of equilibria.

Hence, with the lack of theoretical results for the existence of non-trivial equilibria, we turn to the dynamics of the game. We study experimentally the convergence of a best response update rule for the nodes towards an equilibrium; in particular, we are interested in the likelihood of reaching a non-trivial equilibrium as well as in the efficiency of such an equilibrium, measured by the percentage of the original total demand that is eventually transmitted.

The paper is organized as follows. We start with an overview of related work and background required in section II. We proceed in section III with a formal definition of the network flow problem. We present a simplified version of the model and describe the the theoretical results derives for this simplified model in section IV. Then we give an experimental evaluation of this model in section V, concluding in section VI.

II. Background and related work

Selfish behavior is not new in multi-hop networks; it is rather a frequent and reasonable assumption that captures the behavior of self-interested entities that need to coexist and possibly cooperate in a common environment. Selfish behaviour has been studied using game theoretic techniques in many different areas and problem settings, including wireless ad-hoc multi-hop networks; as already mentioned, this work is a continuation of the work in [1], [2], but the latter was inspired by the CONFIDANT protocol studied in [3]. Naturally the following question arises: does there exist an equlibrium, where nodes do relay flow for others (and therefore do pay the cost for someone else's flow) which which relieve the network from the trivial equilibrium situation mentioned above? Several papers [1], [4], [5], [6] show that, indeed, node strategies that lead to non-trivial equilibria do exist for relatively natural network relay models. There are cases where it is to the benefit of *everyone* involved, or at least most, to relay traffic, because this will lead to a better utility outcome for themselves [1], [2]. Two ways to avoid the trivial solution of zero-relaying, which is a form of the well-known "tragedy of the commons", were mentioned above. Payment schemes is a common way to provide incentives to intermediate nodes to relay packets. Reputation-based protocols are based on keeping records of the past actions of neighbors: each node keeps track of the amount of traffic its neighbors has forwarded in the past and follows a specific protocol to decide the amount of traffic it will route in each round. The decisions can be local [3], [7], [8] (each node decides according to its own private information about the past actions of its neighbors) or centralized (a central authority collects all information as a central repository, and decisions are based on the statistics from the entire network) [9], [5].

In this work we study a reputation-based protocol. Nash equilibria are also explored experimentally in [10], where the relay game is also a reputation-based protocol, and it is modeled as a repeated game. Conditions for the existence of equilibrium solutions are established theoretically, and the authors also present experiments to establish the probability that a random network will indeed allow an equilibrium relay solution. Their experimental results are used mainly to explore particular strategies for the repeated game, or check whether specific conditions hold, while we are concerned with the dynamics of a single stage game and the particular strategies we consider are best response ones.

III. Definition of the model

The protocol we describe here is significantly simpler to that presented previously in [1], or even in [2] (which was itself a simplified version of [1]). We will explain the main differences in more detail further on. Let $G = (V, E)$ be a directed graph, representing a connected network that consists of nodes that are elements of the set V, the set of agents. If there can be a direct link that allows node $u \in V$ to transmit to node $v \in V$, then there is a directed edge $(u, v) \in E$.[1] We are also given a set of O-D pairs $(s_i, t_i) \in V$ for commodities $i = 1 \dots k$, and flow demand $d_i \geq 0$ for each commodity i; commodity i would like to send its flow demand d_i from the origin s_i to the destination t_i. The set of all paths between s_i and t_i is denoted by P_i.

In [1], [2] each commodity is routed by its origin, which can choose to split the commodity demand along any number of paths from P_i. In what is a significant departure, the model in this work allows only for *unsplittable* flows, i.e., each commodity is routed through a *single* path. This path is not a decision variable of the origin node anymore (therefore the latter doesn't have the flexibility of choosing the currently most beneficial routing, but doesn't need to know the current state of the whole network, either). From now on, we assume that the flow paths have been determined[2], and every node is aware of which of its outgoing edges it should use to forward or transmit a particular commodity through, by using a *routing table*. Obviously there must be a distributed protocol that runs before our own, that will populate these routing tables with the correct information; since such a protocol is not a study object for this work, we will assume that the routing tables are given.

Following the general model in [1], we allow intermediate nodes to decide to drop a certain amount of flow (i.e., decide not to relay). Therefore, on a connection $e = (u, v)$, the node v might transmit a certain amount of flow f_e^i, but node u may decide to only retransmit, for example, half of it on the next edge towards it destination. In this case there an amount of flow is not *successful*: it is transmitted by a source node, but it never reaches its destination. One of the theoretical results in [1] is that some networks have equilibrium flow solutions that use only successful flows, i.e., a node never transmits more flow that is actually relayed to its destination. [2] focused on this particular case for their experiments. Since in this work we are interested only in the dynamics of a set of strategies defined for every node that allows only partial information about the network to be used, the distinction between successful and unsuccessful flows doesn't play a significant role for us and we will not assume (as in [2]) that our flows belong only to one of these categories.

For every edge $e = (u, v)$ there are two 'strategic' parameters associated with the decision of how much flow the edge should carry. The receiving node v needs to decide how much flow from this edge it is willing to relay further. Note that an edge $e = (u, v)$ will carry "through flow" (flow that needs to be relayed by v to some destination) and "arriving flow" (flow with destination v). The amount of flow β_e^v that goes through e and v doesn't block (i.e., it allows it to go through v further down its path) is a decision variable of v. On the same edge, u also has a threshold THR_e^u decision variable for the amount of flow it sends to v through e and v blocks. If v is blocking a lot of flow (the amount that v relays β_e^v is so low that the flow dropped becomes more than u's threshold THR_e^u) then u 'cuts off' (i.e., blocks all flow that goes through) edge e. Note that this mechanism can hurt v, because edge e also carries flow with destination v, which v values. This cutting off of an edge is the *punishment* part of the protocol definition, it provides the only incentive for a node to forward third-party commodities, and its mechanism is based on a node's 'reputation' as a 'good citizen', measured by the amount of flow it blocks, i.e., it is a reputation-based protocol. Note that if α_e^v is the amount of flow from e that v blocks, there are two ways of cutting an edge (or, equivalently imposing the inequality $THR_e^u < \alpha_e^v$): either u decreases its threshold THR_e^u, or v blocks more flow by increasing α_e^v.

To summarize, each player (node v) needs to decide on its own strategy, i.e., a set of values for the following variables:

- THR_e^v: The threshold of v for the amount of flow blocked by the other side x of its outgoing edge $e = (v, x)$.
- β_e^v: The total amount of through flow that v receives from its incoming edge $e = (x, v)$ and it forwards. The rest is blocked, with the blocked flow distributed equally (percentage-wise) to all commodities comprising the through flow.
- f^{vx}: The amount of flow v transmits for the commodity with v, x as its O-D pair.

The strategy profile σ_v of a node v contains values for all these variables.

The utility function for each node is a simple generalization of the one in [1]. A node u gets utility from receiving flow (that has destination the node u), and by the fact that flow with origin u actually arrives to its destination. On the other hand, a node u will incur cost (negative utility) whenever it needs to transmit flow (its own or relayed traffic). We do not explicitly associate cost with receiving flow because this

[1]Note that the special case where G is undirected is often used to model wireless ad-hoc networks, when communication links are bi-directional. Clearly, each undirected edge can be replaced by two directed edges of opposing directions.

[2]In our implementations, we use shortest paths (in terms of hops) as a natural path selection.

can be easily modelled in the utility function by choosing the weights appropriately. We define the utility of v as follows:

$$U(v) = w \cdot \left(\begin{array}{c} \text{flow sent by } v \text{ and reached its} \\ \text{destination} \end{array} \right) + \left(\begin{array}{c} \text{flow received} \\ \text{by } v \end{array} \right) - \left(\begin{array}{c} \text{flow forwarded} \\ \text{by } v \end{array} \right) - \left(\begin{array}{c} \text{flow sent by } v \text{ and didn't} \\ \text{reach its destination} \end{array} \right). \quad (1)$$

The parameter w is used in order to capture the trade-off between the utility received by successful flow and the cost of transmission. Note that $w \geq 1$ implies that there is non-negative utility derived from sending flow to a destination after subtracting the cost of transmission; if $w < 1$ then it does not make any sense to transmit any flow.

Each node will choose its strategy in a way that maximizes its own utility, as defined by equation (1). A *Nash equilibrium* in this network game is a complete set of strategies (or *strategy profile*) $\sigma = (\sigma_{v_1}, \sigma_{v_2}, \ldots, \sigma_{v_n})$ (one for each node v_i) such that no node has a unilateral incentive to change its own strategy. In other words, assuming that the nodes are using the strategies in σ, no node v can increase its own utility by changing only its own parameters σ_v in the strategy profile σ.

As mentioned above, it is easy to see that if the source node u of a O-D pair u, v is directly linked to destination v, then it is always beneficial for both nodes if u sends all d_{uv} demand to v. Therefore there is a trivial equilibrium solution, where only neighbor demands are routed in the network and no other flow is sent. We will call this the *trivial Nash equilibrium* or simply the trivial solution.

IV. Definition of a network dynamics

In this section we define a process by which a network of selfish agents such as the one described above evolves over time. This process is determined by the rules each node follows in order to update its strategy (i.e., the values for its decision variables), given some (incomplete) information about the current state of the network. This myopic character of the nodes implies that our dynamics may not necessary converge to a Nash equilibrium as defined above, because the latter presupposes full knowledge of the strategies of all other players. The advantage of this approach is that it is more realistic: no network node can know the whole network and have this information available in the blink of an eye. Hence we are trading some of the theoretically descriptive power of Nash equilibria (which allows, for example, to prove the existence of non-trivial Nash equilibria with nice connectivity properties in [1]) for a more realistic setting for our experiments.

In order to define the network dynamics we need to define two elements: the *input* to a node, and the set of update rules.

Node input: Each node is aware of the complete strategy of each one of its neighbors in the network; it is also aware of how much of the flow it sends to a destination actually arrives (say, through ACK replies). Note that we have already assumed that the routing table for each node is already set up, and that a node has no (direct) information about the strategies of the nodes beyond its neighborhood, or which edges are currently cut off.

Strategy update: We would like to define the broadest set of candidate strategies that its input allows to a node, and pick out of them the best response strategy, i.e., the strategy that maximizes the node's utility. We assume that no other node(s) update their strategies at the same time. While this assumption may seem not realistic, it doesn't affect the implementation (but only the theoretical analysis), and it will not be far from reality if the update algorithm for the node is fast. Then the issue that arises is how to check for the best strategy, given the fact that the set of potential strategies is infinite due to the real (continuous) values that some of the node decision variables take. The crucial observation that resolves this problem is the following:

Theorem 1: The best response strategy is a member of a finite set of strategies with size $O(2^d)$, where d is the degree (together the in- and out-degrees) of the node in G.

Proof: Suppose that s^* is the currently best response strategy for node u, and U^* its node utility. Then we can modify this strategy, without changing u's utility as follows: First look at all incoming links $e = (x, u)$ to u. If e is cut-off by s^* (i.e., $THR_e^x < \alpha_e^u$), then $b_e^u = 0$, i.e., u doesn't forward anything coming from x, because otherwise u could set $b_e^u := 0$ without changing the network, and thus increasing its utility, a contradiction. Similarly, if e is 'alive' (i.e., $THR_e^x \geq \alpha_e^u$), then u has no reason to forward more than $b_e^u = THR_e^x$ amount of flow from x, since otherwise it could reduce its b_e^u without changing the network and thus increasing its utility, a contradiction. Then look at all outgoing links $e = (u, x)$ from u. If e is cut-off by s^* (i.e., $THR_e^u < \alpha_e^x$), then $f^{ut} = 0$, for all destinations t whose path uses e, i.e., u doesn't send anything through x, because otherwise u could set $f^{ut} := 0$ for some t without changing the network, and thus increasing its utility (since it doesn't pay to transmit flow that never arrives at its destination), a contradiction. Similarly, if e is 'alive' (i.e., $THR_e^u \geq \alpha_e^x$), then it can reduce its threshold to $THR_e^u := \alpha_e^x$ (a tiny amount more actually, so e is not automatically cut off) without affecting the network or its utility. From the amount of successful flow it can send through e (which it knows due to the ACK signals), the amount of flow for every commodity that x forwards (which it also knows), and its own b^u values (that have already been uniquely determined), it can determine uniquely[3] the optimal value for its f^{ut} decision variables, together with the exact amount of each commodity that it blocks in its incoming edges, in order to maintain the network topology. In other words, we have just proven that once we know which exactly of the edges incident to u are cut off by s^*, we can determine values for

[3]Actually there may be more than one ways for doing this. We break these ties by favoring the commodities in a lexicographic order when determining how much flow to send.

u's decision variables that achieve utility U^* and respect the network topology imposed by s^*. Hence, if we go over all $O(2^d)$ possibilities of edges incident to u being 'alive' or not and do the calculations above for each one of them, at least one of these strategies will achieve utility U^*; the set of these strategies is the set of the theorem statement. ■

Note that even if the size of the set of candidate best response strategies depends exponentially on the node degree, in practice one doesn't expect to encounter high-degree nodes often. Therefore, the update rule based on Theorem 1 and described in Algorihtm 1 should run very fast in typical cases.

Algorithm 1 Strategy update for a node u

```
1: boolean $F[\text{degree}(u)]$
2: $current\_best\_s := \emptyset$
3: $current\_best\_util := -\infty$
4: for all combinations of $F$ do
5:     for all $e = (x, u) : F[e] = 0$ do
6:         $b_e^u := 0$
7:     end for
8:     for all $e = (x, u) : F[e] = 1$ do
9:         $b_e^u := THR_e^x$
10:    end for
11:    for all $e = (u, x) : F[e] = 0$ do
12:        $f^{ut} = 0, \ \forall t$ s.t. $(u, t)$-path uses $e$
13:    end for
14:    for all $e = (u, x) : F[e] = 1$ do
15:        Compute $f^{ut}$, flows blocked by $u$ as in Theorem 1
16:    end for
17:    $current\_s := (f, b, THR)$
18:    if $U(current\_s) > current\_best\_util$ then
19:        $current\_best\_s := current\_s$
20:        $current\_best\_util := U(current\_s)$
21:    end if
22: end for
```

V. Evaluation

In this section, we present a set of experiments to evaluate the forwarding model presented above. Our main goals are the following: The theoretical results indicate that equilibria solutions may exist for some networks, and that localized dynamics may actually lead to an equilibrium solution. We run experiments to establish that following the dynamics of this game will lead to non-trivial equilibrium solutions for random graphs. These equilibrium solutions route non-zero fraction of the available flow potentially from every commodity. We establish experimentally how the equilibrium solutions reached by local, greedy, and selfish dynamics, depends on various paramteres of the graph. We analyze random families of graphs with randomly chosen source-destination pairs and demands. For these random graphs we run a simulation of the game dynamics described above. Graphs are generated according to the Erdös-Renýi model (G_{nm}), and the Barabasi-Albert powerlaw model. Various edge densities are used. Self loops and multi-edges are not allowed. Commodity pairs are chosen

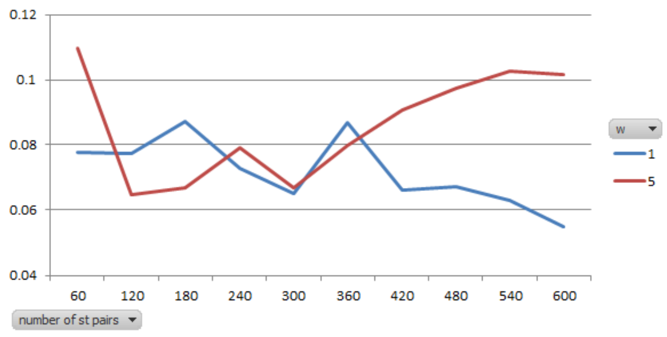

Fig. 1. Percent of available flow routed at equilibrium reached by network dynamics for Erdös-Renýi graphs as a function of commodity density

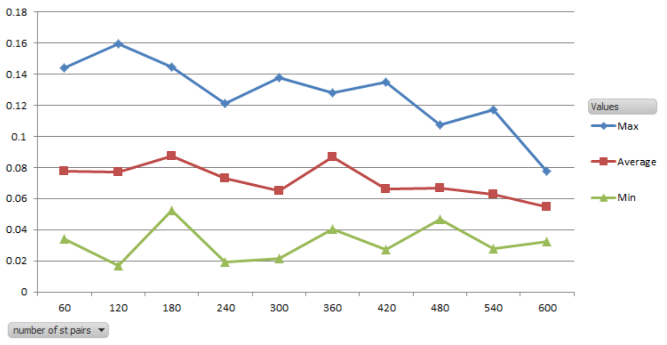

Fig. 2. Minimum, maximum, and average percentage of available flow routed at equilibrium as a function of commodity density, for $w = 1$

uniformly at random. Flow demands are chosen uniformly at random between the predetermined minimum and maximum values. The parameter w quantifies the utility of flow in the network as a function of the amount of flow. We experiment with various values of this parameter.

Given a randomly generated instance (graph plus commodities) we run the game dynamics simulation until it converges to an equilibrium, and compare the total routed flow at equilibrium with the best possible solution. The optimal solution is the case that all flow is routed (complete cooperation). Note that trivial flow, which is the flow between source-destination pairs that happen to be neighbors, will always be routed so when we compute the ratio of equilibrium flow versus optimal, we subtract the trivial flow. In order to understand the efficiency of the equilibrium solution, we define the *equilibrium flow ratio*, to be (equilibrium flow - trivial flow) divided by (optimal flow - trivial flow). This ratio will be always between 0 and 1 (or 0 and 100% if expressed as a precentage). Our experiments explore the equilibrium flow ratio for varying values of commodity density, and the parameter w, the utility to cost ratio of routed flow.

Figure 1 shows two series of values, for Erdos-Renyi graphs, for $w = 1$ and $w = 5$. The figure shows how the percentage of flow routed at the dynamic equilibrium depends on the density of commidities in the graph (number of origin-destination pairs). The figure shows the ratio of flow that is routed at equilibrium reached by network dynamics, averaged over 10

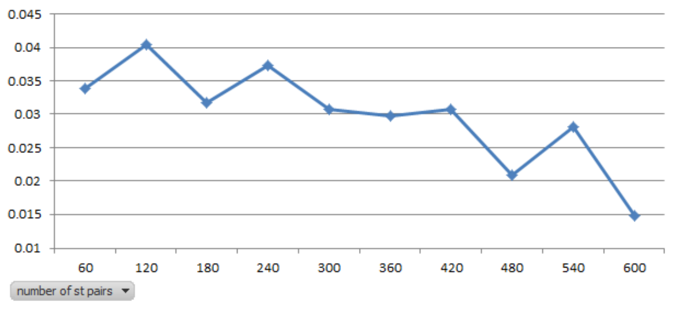

Fig. 3. Standard deviation of the average percentage of flow routed at equilibrium as a function of commodity density, for $w = 1$

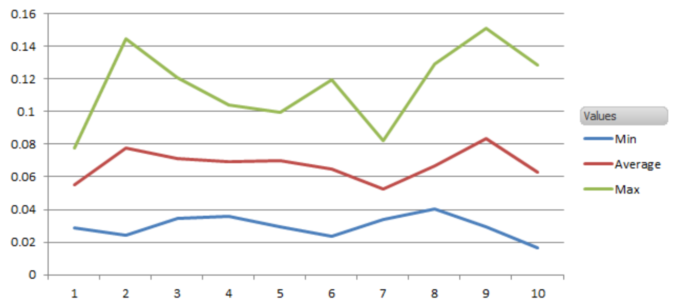

Fig. 4. Minimum, maximum, and average percentage of available flow routed at equilibrium as a function of the parameter w, for demands chosen in the interval $[1, 10]$

runs. Figure 2 at the minimum, average, and maximum values achieved for the flow routed at equilibrium over these runs.

We see that the gap between the minimum and maximum values closes as the number of commodities increases. Indeed, looking at Figure 3 the standard deviation of the equilibrium flow ratio shows the same clear trend. This is an interesting point about the network dynamics. As the number of origin destination pairs increases, each user in the network finds it slightly more difficult to find paths for all the destination. There is slight decrease in overall routed flow as the number of commodities increases. Note however, that these experiments assign random demand values chosen from 1 to 10. This means that there is a large difference in the amount of flow each origin destination pair wants to send. An st pair demand may require 10 times higher amount of flow than other pairs. We can restrict this to see how it affect the dynamics. In the next figures, we compare the equilibrium achieved when choosing demands in the interval $[1, 10]$, or in $[8, 10]$. In the second case all demands are much more balanced in terms of amount of flow they would like to route. The trend is still similar. The spread of the values also decreases as the number of commodities increases.

Figures 5 and 4 show the minimum, average and maximum ratio of equilibrium flow over maximum possible, for minimum demand 1 and minimum demand 8 (maximum demand is always 10).

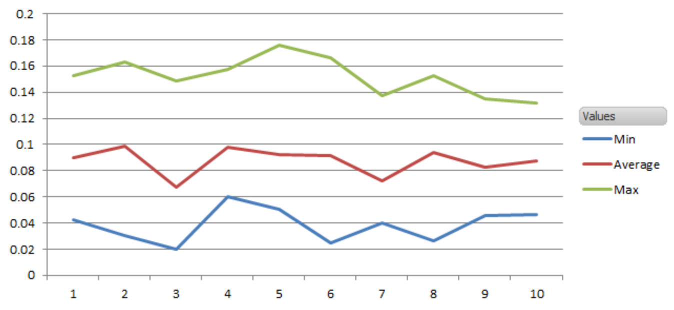

Fig. 5. Minimum, maximum, and average percentage of available flow routed at equilibrium as a function of the parameter w, for demands chosen in the interval $[8, 10]$

VI. Conclusion

We considered a natural relaying problem modelled as a game theoretic problem. We focused on a recent game theoretic model that established the existence of equilibrium flows that are based on natural incentives for relaying as opposed to payments. We experimentally established that these equilibrium solutions can carry a significant fraction of the available flow. Our experiments show that the efficiency of the equilibrium solution increases with edge density and demand density in the network.

References

[1] G. Karakostas and E. Markou, “Emergency Connectivity in Ad-Hoc Networks with Selfish Nodes,” *Algorithmica*, August 2012.

[2] G. Karakostas and A. Viglas, “Analysis of a forwarding game without payments,” in 13^{th} *Int. Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, 2012, pp. 691–696.

[3] S. Buchegger and J.-Y. Le Boudec, “Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes : Fairness In Dynamic Ad-hoc NeTworks),” in *Components*, ser. Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), vol. 48, no. 3, 2002, pp. 226–236.

[4] A. Urpi, M. Bonuccelli, and S. Giordano, “Modelling cooperation in mobile ad hoc networks: a formal description of selfishness,” in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2003.

[5] V. Srinivasan, P. Nuggehalli, C. F. Chiasserini, and R. R. Rao, “Cooperation in Wireless Ad Hoc Wireless Networks,” in *IEEE Infocom*, 2003.

[6] P. Michiardi and R. Molva, “Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks,” in *Conference on Communications and Multimedia Security*, vol. IFIP 228, no. December, 2002, pp. 107–121.

[7] Q. He, D. Wu, and P. Khosla, “SORI: a secure and objective reputation-based incentive scheme for ad-hoc networks,” pp. 825–830 Vol.2, 2004.

[8] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, “Sustaining Cooperation in Multi-Hop Wireless Networks,” *Symposium A Quarterly Journal In Modern Foreign Literatures*, vol. pages, pp. 231–244, 2005.

[9] F. Milan, J. J. Jaramillo, and R. Srikant, “Achieving cooperation in multihop wireless networks of selfish nodes,” *Proceeding from the 2006 workshop on Game theory for communications and networks GameNets 06*, p. 3, 2006.

[10] M. Felegyhazi, J. P. Hubaux, and L. Buttyan, “Nash equilibria of packet forwarding strategies in wireless ad hoc networks,” pp. 463–476, 2006.