Equilibrium flows and path dilation for a network forwarding game

George Karakostas
McMaster University
karakos@mcmaster.edu

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

May 5, 2013

Abstract

In network routing problems, assuming that packet transmission incurs costs, the optimal solution would chose to route all flow along shortest paths. Intermediate nodes will need to relay the flow in most cases and therefore will require some sort of payment, if the flow they are asked to relay does not bring any utility to them. We consider such a forwarding game on directed graphs where nodes need to send certain amount of flow (packets) to specific destinations, possibly through several relay nodes. All nodes in the network act selfishly and will forward packets only if it is to their benefit. The model assumes that each node receives some utility from sending its flow to the predetermined destinations and from receiving flow. However each node has to decide whether to relay flow as an intermediate node from other sources, as relaying has an associated cost. This model assumes that there is no payment scheme. Somewhat surprisingly, this game has possibly several strategies that allow a significant amount of the flow to be routed while all nodes have a positive outcome, which suggests that in this model the nodes have indeed incentives to relay flow even if payments are not explicitly allocated. Previous theoretical work establishes the existence of these strategies (Nash equilibrium solutions). In this work we simplify the original network model, and provide the first experimental evaluation of these equilibria for different classes of graphs. We provide clear evidence that these equilibrium solutions are indeed significant and establish how these equilibria depend on various properties of the network such as average degrees and flow demand density. Our main results establish experimental bounds for the path dilation in this model: the average ratio of the routed flow cost at equilibrium over the cost of the optimal routing which would involve shortest path routing in our model.

1 Introduction

Network routing typically requires transmissions to go through intermediate nodes that relay traffic from an origin to a destination. In many cases intermediate nodes receive no utility from relaying traffic for an origin-destination pair: the reason the node is asked to relay the traffic can be simply because it happens to lie on the shortest path from the origin to the destination node. However retransmission may incur a cost, especially in wireless networks for example. This can be addressed by designing a scheme of payments. A payment can provide an incentive for intermediate nodes to realy the required traffic. However, we can see that there are different incentives, already present in the network for relaying traffic: if all nodes decide not to relay anyone else's traffic, then only neighbors will be able to communicate. Therefore, assuming that all nodes participate in the network because they require to communicate with several other nodes, they would prefer to have a functioning network, so that their own flow is routed to its destination, and also receive the flow that is addressed to them. In other words, each node has an incentive for other nodes to keep relaying traffic for its own benefit. If a node y decides to stop relaying

incoming traffic of a neighbor x, then the neighbor, naturally, will be unhappy and may choose to punish the node by stopping all traffic towards it, including traffic that was trying to reach node y itself. Therefore, the decision of x not to relay traffic results in x losing out on traffic addressed to itself. In that case the node has a clear incentive to relay traffic up to a point, in order to avoid punishment from its unhappy neighbors. Such a model, does not involve payments in order to achieve successful relaying of traffic, but it does involve the possibility of punishment of a non-relaying node. The incentives are different, but still aligned towards getting some of the nodes to decide to cooperate.

This model can be formalized as follows. We consider the scenario of a connected network, modelled as a directed unweighted graph. In this network we have a number of designated origin-destination pairs s, t, each associated with a positive parameter $d_{s,t}$. These origin-destination pairs describe the network flow demands in the network: source node s wants to send an amount of flow $d_{s,t}$ to its target node t. Each node might be a designated source (and therefore would like to send flow to specific destinations), or a destination (and therefore would like to receive flow from predetermined source nodes), or, in most cases, it is both a source and a destination. Nodes receive utility from all flow successfully delivered or received. All nodes need to pay a cost which is proportional to the amount of traffic they need to transmit. If a source node can communicate directly with its target then of course it is to the benefit of both to have 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. These nodes can decide to relay traffic and therefore pay the cost of transmission themselves for someone else's traffic. Although this seems counterintuitive, it has been established very recently that there exist cases where it is the node's overall benefit to relay someone else's traffic, although not necessarily all of the traffic requests. Therefore, assuming that each node plays strategically, there exist cases where it is the benefit of every node in the network to relay traffic for others, even though everyone is selfish (tries to maximize its own utility) and there are no payments allocated.

The recent work of Karakostas et al. [4] establishes theoretically that these solutions exist in instances of this network traffic problem, however those results do not give any insight of whether such solutions exist often, or whether they are realistic. For example, it is possible that only an infinitesimal fraction of the network instances actually have such solutions. Or it might be the case, that in the only such solutions, only an infinitesimal amount of the original flow finds its way to its destination.

In this paper we provide an experimental investigation of this model. We show that these solutions are realistic, in the sense that random network instances have non-trivial solutions with high probability, and those flow solutions carry a significant fraction of the total flow. We also establish how these equilibrium solutions depend on various network parameters. Our work includes minor modifications and simplifications to the model used in previous work [4]. Furthermore, our main results involve the analysis of the price of having an incentive based routing in this forwarding model. In general, the price of anarchy refers to the ratio of the cost achieved at the equilibrium solution divided by the cost of the optimal solution. In our model, the cost is the energy required for relaying and all transmission. Therefore this cost is directly related to the length of the chosen routing paths. The optimal solution would route all flow along the shortest path connecting the origin to the destination node. However, at equilibrium, flow may be routed along longer paths that include nodes that have the right incentives. Therefore, quantifying the price of anarchy involves lookings at *path dilation*, i.e., the length of the equilibrium routing paths compared to the shortest paths. Our experiments show that path dilation at equilibrium is, in most cases, less than 2, i.e. the eqiulibrium paths are no more than twice the length of the shortest path, and often the dilation is about 1.4, i.e., the equilibrium flows are about 40% longer than the shortest paths.

The main contributions of this work include establishing the fact that the equilibrium strategies exist very often in practice, carry a significant amount of flow, and lead to a moderate path price of anarchy. Our experimental results show how the strategies are

affected by different parameters of the network: (1) the average degree, (2) the demand density, and (3) the type or structure of network connectivity.

The paper is organized as follows. We start with an overview of related work and background required in section 2. We proceed in section 3 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 4. Then we give an experimental evaluation of this model in section 5, concluding in section 6.

2 Background and related work

In multi-hop networks, selfish behaviour is a frequent and reasonable assumption that captures the behaviour of self-interested entities that need to coexist and possibly cooperate in a common environment. A selfish node in a network will choose an action that maximizes its own utility (or payoff) without any concern about the result of its decisions to the rest of the nodes. Selfish behaviour has been studied using game theoretic techniques in many different areas and problem settings, including wireless ad-hoc multi-hop networks [1]. For a wireless sensor network for example, every node needs to preserve its battery life, as it is usually a scarce resource. However, if nodes choose to refuse to relay traffic, the network will cease to function. This will lead to no flow being delivered to its destination and all utilities being equal to zero for all nodes. Note that here we ignore flow demands between source-destination pairs that can communicate directly. These flows will always be successful in our model, and form what we will refer to as a trivial solution, or trivial equilibrium. If nodes decide not to relay any flow, the network is in a worst-case equilibrium (a standstill) in the network. The strategy of a node not relaying for anyone, also results in their own flow not being relayed. Naturally the following question arises: do there exist strategies, where nodes do relay flow for others (and therefore do pay the cost for someone else's flow) which relieve the network from the trivial, no-flow standstill situation mentioned above? Several recent papers [4, 9, 8, 6] show that indeed these strategic solutions do exist for relatively natural network relay models. There are cases where it is to the benefit of *everyone* involved to relay traffic, because this will lead to a better utility outcome for themselves [4]. There are many ways to avoid the trivial solution of zero-relaying, which is a form of the well-known "tragedy of the commons". 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 [1, 3, 5] (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) [7, 8].

We focus on the work related to connectivity in such networks based on reputation systems, following the analysis of [4]. The main result we focus on, shows that equilibrium forwarding strategies do exist, without any payment or actual explicit reputation system. In fact the main theoretical results shows that such equilibrium forwarding strategies exist that route a non-zero fraction from *every* source-destination pair. This is a surprising and interesting result that raises many immediate questions about the practical properties of such equilibrium strategies. Note that the result does not guarantee that such equilibrium strategies exist for every network instance. Far from that, there are several simple networks that certainly do not have any such equilibrium strategies except trivial ones (the ones that connect only neighboring source-destination pairs and therefore there is no relaying involved). The natural question to ask is how often do these networks have such equilibrium flows, and how significant these solutions are. The theoretical results guarantee that if an equilibrium exists, a non-zero amount of flow is routed for every source-destination pair, however it may be possible that the fraction of the flow routed is insignificant. In this work we answer both these questions, giving positive answers that show the practical im-

portance of this model as well as the already established theoretical one.

Similar Nash equilibrium solutions for relay games are also explored experimentally in by Félegyházi et al. [2]. The relay game in that work is not using payments or any other explicit incentive for relaying nodes, but it is based on reputation. The game is modelled as a repeated game and conditions for the existence of equilibrium solutions are established theoretically. The authors also present experiments to establish the probability that a random network will indeed allow an equilibrium relay solution. However the experimental results are used mainly to explore particular strategies for the repeated game, or check whether specific conditions hold.

3 Definitions

We describe a model that is significantly simplified compared to that presented previously in [4] but still captures an important sub-class of the forwarding game, where only successful flows are routed in the network. We will explain this distinction 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. If node $u \in V$ can communicate directly by sending data to node $v \in V$, then there is a directed edge $(u, v) \in E$. The special case were G is undirected is a reasonable model for wireless ad-hoc networks, when communication links are undirected. We are also given a set of source-destination pairs $(s_i, t_i) \in V$ for $i = 1..k$, and flow demands $d_i \geq 0$ for each pair. We will call each such pair a commodity. The i-th commodity, would like to send its flow demand d_i from the source s_i to the target t_i. Each commodity can choose to split the flow along any number of paths from s_i to t_i. The set of all paths between s_i and t_i is denoted by P_i. Note that we describe the model using these sets of paths as it is more intuitive and easier to formulate our results. This formulation would be exponential in size and would not be useful in practice, and it is often used in related literature. Later on in this work, we show how to formulate this model in terms of the graph edges so that all formulations are polynomial in size. The amount of flow that commodity i assigns on edge $e = (u, v)$ is denoted by f_e^i or f_{uv}^i. The general model in [4] allows intermediate nodes to decide to drop a certain amount of flow (decide not to relay). Therefore on a connection $e = (u, v)$ the node u might transmit a certain amount of flow f_e^i, but node v may decide to only retransmit, say, half of it on the next edge towards its destination. In this case there an amount of flow that is not successful: it is transmitted by a source node, but it never reaches its destination. One of the theoretical results in [4] is that some networks have equilibrium flow solutions that use only *successful flows*. That is a node never transmits more flow that is actually relayed to its destination. We focus on this particular case for our experiments. There are also equilibrium solutions with unsuccessful flows according to [4] but those are more complex and less practical to work with as it is NP-hard to compute them, or to check whether they exist. Note that the model we define here is different and significantly simplified compared to the one in [4]. However there is no loss in generality of the model for the particular case we are interested in (equilibrium solutions with successful flows only). 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 will 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 maximum amount of flow of the edge $e = (u, v)$ that v is going to tolerate is denoted by β_e. This means that v is simply not going to relay anything more than this limit. The limit β_e needs to be decided by node v and is one of the strategic variables in the model. On the same edge, u also has a threshold that shows its own tolerance of dropped flow from v. If v is dropping a lot of flow (β_e is too low) then u might decide not to forward any flow to v, by cutting off the edge e. Note that this is an important decision that can hurt v because the edge e also carries flow with destination v. In other words, u will send two kinds of flow to v, flow to relay further, and flow with destination v. If v

does not relay enough then u can cut off the edge, and v will lose the flow with destination v it received through that edge. For every edge $e = (u, v)$, the node u has a strategic variable α_e that denotes the minimum amount of through-flow that v is expected to relay. If v relays less, then the edge e is automatically cutoff. So, whenever $\beta_e < \alpha_e$ the edge e will be cut off. Therefore a_e plays the role of the threshold referred to before. Cutting off an edge this way is part of the model definition. An edge $e = (u, v)$ can be cut off either by u, by increasing its expectation α_e above v's limit, or it can be cut off by v lowering the flow it relays (reducing β_e below u's threshold). This completes the description of the parameters and decision variables of the network.

Now we need to define the utility function for the players (network nodes). The utility function we introduce here is a simple generalization of the one in [4]. A natural way to measure the utility of a node in this model is the following. A node u gets utility from receiving flow with 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 can be easily modelled in the utility function by choosing the weights appropriately. We define the utility of a node y in the following equation.

$$U(v) = w_s \sum_{\substack{e \in out(v) \\ x \in V}} f_e^{vx} + w_r \sum_{\substack{e \in in(v) \\ x \in V}} f_e^{xv} - \sum_{\substack{e \in out(v) \\ x,y \in V}} f_e^{xy} \quad (1)$$

The parameters $w_s \geq 2$ and $w_r \geq 1$ can be used to model the utility of successful flow, and also the trade-off between successful flow utility and cost of transmission. The parameter w_s is used for the flow that is sent from a node, and w_r is used for the utility of arriving flow. We assume that $w_s \geq 2$ since we need to have some utility from sending flow to a destination after subtracting the cost of transmission (if $w_s = 1$ then it does not make any sense to transmit any flow). The paramters w_s, w_r depend on the application and other details of the model. Determining values for these parameters can be a difficult task.

We have now defined all the ingredients of the this game theoretic model. There is a player for each node, with the utility function defined in equation (1). Each player (node v) needs to decide on its own strategy, which incudes the following variables:

- α_e: The tolerance values for each outgoing edge $e = (v, x)$. If the target neighbor x on the edge e is not relaying at least α_e then the edge is cut off
- β_e: The drop thresholds for each incoming edge $e = (x, v)$. v will relay at most β_e of flow coming from e.
- f_e^{vx}: The flow assignment that v will route towards all of its assigned targets x.

The strategy profile of a node σ_v contains all of the above parameters: all values α_e for outgoing edges, all β_e for incoming edges and all flow assignments f_e^{vx} for all edges in the graph and all destination nodes x.

Each node will choose to relay traffic in a way that maximizes it own utility as defined by equation (1). In order to maximize its utility it needs to pick a strategy profile σ wisely. A Nash equilibrium in this network game, is a complete strategy profile for all nodes $\sigma = (\sigma_{v_1}, \sigma_{v_2}, \ldots, \sigma_{v_n})$ 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. This is the standard definition of Nash equilibrium, adapted for the network game we have defined here. We will refer to such a strategy profile as the *Nash equilibrium* or the equilibrium solution.

Now recall that the network instance includes many source-destination pairs, and for each pair there is some amount of demand (maximum amount of flow

available to be sent to the destination). In fact every node will be part of some such pair otherwise there is no reason for being part of the network in the first place. It is easy to see that if the source node u of a source-destination pair u, v is connected directly to the target v (there is a directed edge (u, v) in the graph) then it is always beneficial for both nodes for u to send all of its 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. Obviously we are interested in non-trivial equilibria and in what follows. We say that a flow assignment is *connected* if it routes a non-zero amount of flow for each commodity. A connected non-trivial Nash equilibrium solution is what we are interested in. In what follows, whenever we refer to an equilibrium solution we mean a connected non-trivial Nash equilibrium solution, unless we explicitly want to make a reference to trivial solutions or commodities with zero flow routed.

4 Existence of equilibrium solutions

Looking at the definition of the model we see that the equilibrium solution depends heavily on the choice of the α and β parameters. A node would rather relay as little through-traffic as possible: through traffic only incurs cost for it. However if the node starts reducing the amount of flow it is supposed to relay, then incoming edges will eventually be cut off and this will stop the node from receiving flow destined for it and therefore lose utility by that fact. Hence the decision to relay less traffic needs to be balanced with the traffic a node expects to receive from each edge. This is precisely the point that is used to characterize the equilibrium solutions in this network game. Following the analysis of [4] we can extend the main theorem regarding successful flows to our network model that has a slightly generalized utility function. The main difference in the utility function we introduce here, is the use of the scaling parameters w_s and w_r.

So far we have described the flow assignments in terms of edges. We now switch to path flows as this formulation makes the description of the theorem and flow splits more intuitive. Everything can be described in terms of edges and, in fact, we do use the edge based formulation in our experiments.

For every source-destination pair (u_i, v_i) we denote the set of all possible paths connecting u_i to v_i by P_i. We also denote by $\mathcal{P}$ the set of all relevant paths in the network, $\mathcal{P} = \cup_i P_i$.

Recall that each node is essentially trying to optimize the amount of flow it will get to its destination, but it will also need to make sure the assignment it proposes is tolerable by the nodes it needs to use as relays. We can indeed write this joint optimization problem as a linear program, and for $w_s = 2$ we get the following.

$$\textit{maximize} \quad \sum_{P \in \mathcal{P}} f_P \tag{2}$$

$$\textit{subject to}:$$

$$\sum_{\substack{P \ni e \\ through \\ edge}} f_P - w_r \cdot \sum_{\substack{P \ni e \\ final \\ edge}} f_P \leq 0 \qquad \forall e \in E$$

$$\sum_{P \in P_i} f_P \leq d_{u_i v_i} \quad \forall i$$

$$f_P \geq 0 \qquad \forall P \in \mathcal{P}$$

Note that the size of this linear program is exponential in the number of nodes since it is formulated using paths. However we can easily convert it to a linear program written on the network edges that has size polynomial in the size of the network.

Let D be the total demand in the network between neighboring nodes. A trivial equilibrium solution will route a total flow equal to D. The following theorem describes the non-trivial equilibrium solutions [4]:

Theorem 1 *A network game has non-trivial equilibrium solutions with only successful flows if and only if the linear program described in* (2) *has a solution* $f^* P$ *with objective value* $\sum_{P \in \mathcal{P}} f_P^* > D$.

The complete proof of this theorem is analogous to the proof of Theorem 2 in [4].

5 Evaluation

In this section, we present an extensive set of experiments to evaluate the forwarding model presented above. Our main goals are the following:

1. The theoretical results state that equilibria solutions may exist for some networks. How often do these network games actually have such equilibrium solutions? Are these solutions practically significant?

2. These equilibrium solutions route a non-zero fraction of the available flow potentially for every commodity. But is this routed fraction significant or very close to zero?

3. How good are the equilibrium solutions compared to the optimal routing solution?

We answer all these basic questions by analysing random families of graphs with randomly chosen source-destination pairs and demands. For these random graphs we solve the linear program (2) and find the equilibrium flow assignments.

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 uniformly at random. Flow demands are chosen uniformly at random between 1 and a predetermined maximum value. All demands are integers. We choose $w_r = 1$ for our experiments. As discussed above, the parameter should be greater or equal to 1 in general. The boundary case of $w_r = 1$ means that arriving flow is worth as much as the cost to transmit it.

Given a randomly generated instance (graph plus commodities) we solve the linear program (2) and compare the total routed flow at equilibrium with the best possible solution. In the optimal solution 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

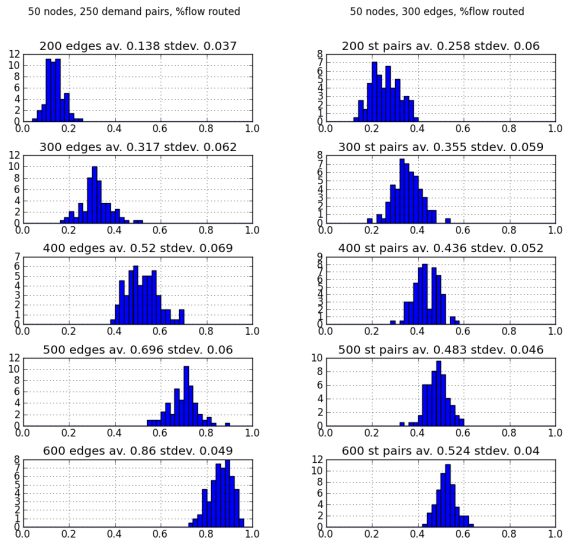

Figure 1: Percent of available flow routed at equilibrium for Erdös-Renýi graphs as a function of edge density, and commodity density

solution, we define the *equilibrium flow ratio* to be (equilibrium flow - trivial flow) divided by (optimal flow - trivial flow). This ratio will always be between 0 and 1. Our experiments explore the equilibrium flow ratio for varying values of edge density, commodity density, and graph type. The experiments are done on a 16-processor Intel Xeon 2.27GHz machine with 24GB of memory. The linear program is solved by the CPLEX solver, using IBM's OPL Studio.

5.1 Amount of successful flow at equilibrium

Figure 1 shows two series of histograms. On the left we include a series of histograms that plot the resulting equilibrium flow ratio distribution for 100 different experiments with increasing edge density. On the right we have a similar series of histograms with increasing commodity density. Each histogram also shows the average and standard deviation of the plotted values. We see that for a relatively sparse net-

work with average degree 4 (top left histogram), the equilibrium solution will route just below 14% of the available flow. For a network with average degree 10 however, the equilibrium solution is expected to route close to 70% of the available flow. For average degree 12 the equilibrium flow is over 85% of the total available (bottom left histogram). We see a similar but slightly more modest increase as the commodity density increases. Figure 1 shows that equilibrium flows carry non-negligible flow in the network and they increase significantly when the network becomes more dense, or when the flow demands in the network become more dense. This is an interesting, typically game theoretic result: the more overloaded the network becomes, the more efficient are the equilibrium flows. In other words, the incentives become stronger for nodes to relay when the network is more dense with flow demands, or with possible flow paths. Going back to the description of the equilibrium constraints of this game, we see that essentially an equilibrium flow solution is looking for cases where a node has an incentive to relay flow in order to keep edges open, that carry flow towards that node. In a sense, these open edges are a deal between nodes across an edge $e = (u, v)$: u will continue to send v flow with destination v (and therefore flow that v wants to receive) provided that v relays enough flow further in the network. Increasing the edge density by adding more edges in the network, increases the probability that there are such deals to be made across edges for each flow demand. If the network is sparse then the existing paths may not bring together nodes whose interests are aligned. The same holds true for increasing density of commodities. The more source-destination pairs we add to the network the more likely it becomes that edges will remain open as the interests of the two nodes on those edges are aligned. Figure 1 also shows that edge density has a stronger impact on the equilibrium flow routed. A dense network is likely to reach a very good level of efficiency, routing almost all available flow.

For powerlaw graphs we see a different picture. We generate powerlaw graphs using the Barabasi-Albert preferential attachment model. Figure 2 shows how the routed equilibrium flow changes when edge density increases and when demand density increases.

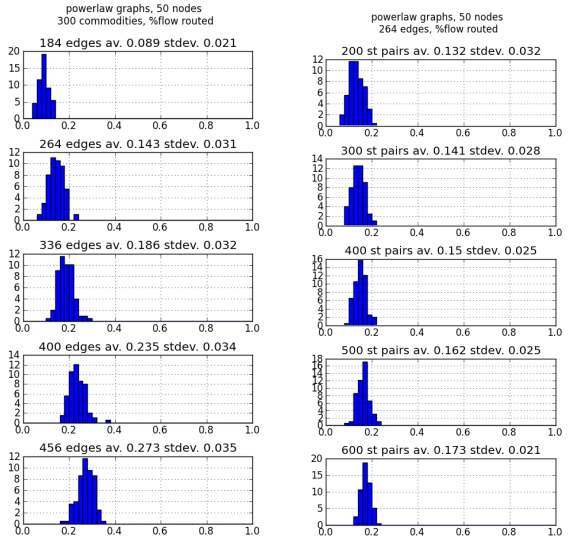

Figure 2: Percent of available flow routed at equilibrium for powerlaw graphs as a function of edge density (left) and demand density (right).

For similar values of edge density and commodity density, we see that the expected equilibrium flow is significantly lower than what we see in Erdös-Renýi graphs. However we still see that increased density has a positive effect (expected amount of flow routed is still increasing) but at a more modest rate. We also observe that increasing edge density has a marginally more positive effect on the flow routed at equilibrium. The equilibrium flows are calculated by solving the linear program (2). The generated linear program has a size that grows fast with the graph size. For a graph with n nodes, m edges, c commodities, the number of constraints is bounded by $O(n \cdot m + m^2 \cdot c)$. This is a generous over-estimate as the expected node degree is smaller than what is used for the calculation of the upper bound (maximum possible degree is $O(m)$, the number of edges in the graph). The expected size of the linear program is $O(n \cdot d + d^2 \cdot c)$ where d is the maximum of the average out-degree and average in-degree. In practice the solve-time can vary greatly with different LP instances. The run-

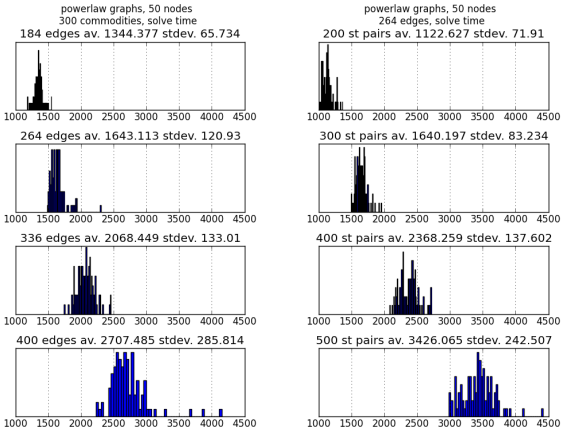

Figure 3: Solve time for finding the flow routed at equilibrium for powerlaw graphs as a function of edge density (left) and demand density (right).

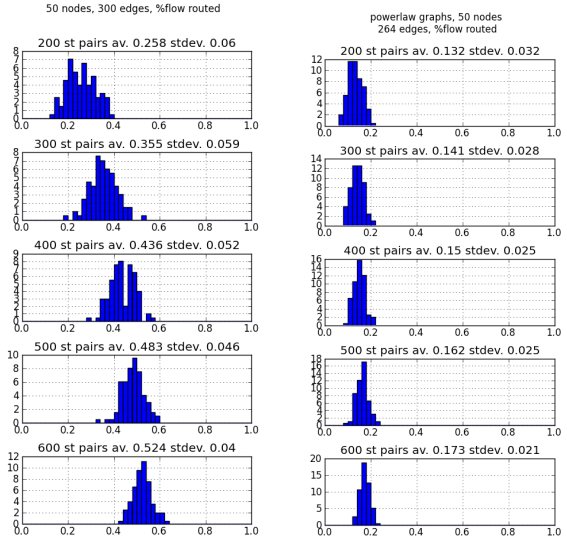

Figure 4: Percent of available flow routed at equilibrium for Erdös-Renýi graphs and for Barabasi-Albert graphs as a function of commodity density

ning time is shown in Figure 3 for powerlaw graphs. The results show that the variance increases as the number of edges (edge density) and the number of commodities increase.

Figure 4 compares the percent of available flow routed at equilibrium as a function of increasing commodity density. The results for Erdös-Renýi (G_{nm}) graphs are on the left side and the results for powerlaw (Barabasi-Albert) graphs are on the right. We see clearly that the equilibrium solution has a stronger dependence on the available commodity flow pairs for G_{nm} graphs rather than for powerlaw graphs. A well-known characteristic of powerlaw graphs is that there is a small number of strongly connected nodes, nodes with degree that greatly exceeds the average degree of the graph. These high-degree nodes are usually referred to as "hubs". In terms of connectivity, powerlaw graphs are more fault-tolerant than Erdös-Renýi graphs, in case of random failures of nodes or edges. Adding a random edge in the network may connect a hub to another node, or connect two average-degree nodes. In case the new edge is added to a hub, the result is minimal in terms of connectivity and path length, since the hub node is already very well connected, with a very high degree. If the new edge connects two average degree nodes, then the new edge is in a part of the network that does not affect connectivity as much, as the hubs are the nodes that make the components of the network well-connected. Hence adding new edges in a powerlaw graph is expected to have a moderate effect in terms of connectivity. The structure of powerlaw graphs makes it more common for connecting paths to pass through hubs: Nodes with average degree are usually in components that are connected together by hubs. This is consistent with the results for sparse graphs that we have shown above. As the commodity density increases, the graph connectivity is not affected in any way. Most of the added commodity pairs will naturally involve average degree nodes, and therefore will have fewer chances of finding willing relaying nodes. As a result, the increasing commodity density does not have a strong impact on the routed traffic at equilibrium.

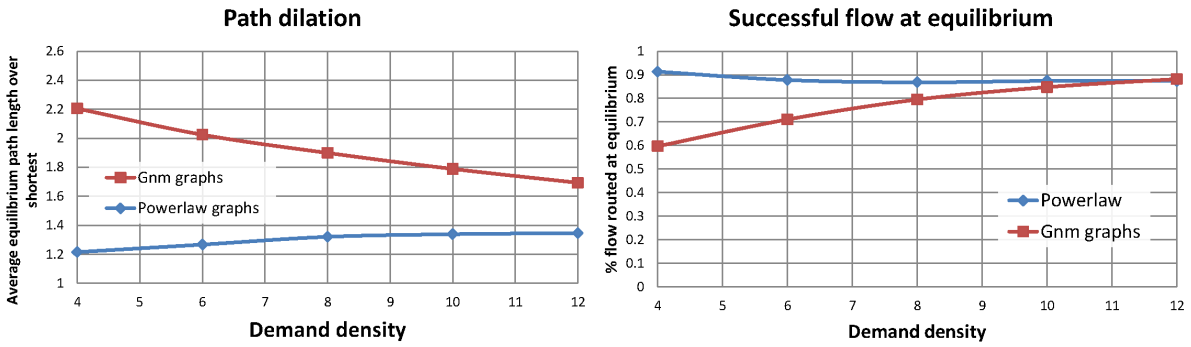

Figure 5: Path dilation and percent of available flow routed at equilibrium for Erdös-Renýi G_{nm} graphs and for Barabasi-Albert powerlaw graphs as a function of demand density

5.2 Path dilation

Given a network instance (network connectivity and origin-destination demand pairs), the optimal routing of the flow would be along the shortest path connecting the origin and the destination of each demand pair. This minimizes the total energy required for the flow to be transmitted and reach its destination.

However, this routing may not be compatible with the incentives of the individual nodes: a selfish node on a shortest path may not have an incentive to relay flow for others. Hence the chosen paths at equilibrium may use longer paths. These paths are less cost-efficient, but invlolve nodes that are willing to keep certain edges open, and therefore relay flow, since they are receiving flow for themselves from those edges. The lack of centralized optimal control is expected to lead to longer routing paths. In this section we present experimental results to quantify how much longer we should expect the equilibrium paths to be, compared to the optimal ones (shortest paths). We run experiments on randomly generated graphs as before, on Erdös-Renýi and Barabasi-Albert (powerlaw) graphs, varying the edge density (average number of edges per node) and the demand density (average number of destinations per node).

As mentioned above, powerlaw graphs have a very different structure to Erdös-Renýi graphs with a small number of very popular, highly connected nodes, and a long tail of sparsly connected nodes. The graphs usually have relatively short paths connecting nodes, but many of those paths are expected to pass through the same, very popular nodes.

Therefore, when we increase the density of the graphs, we do not expect to see a very strong impact to path dilation or amount of flow routed, since most of the new edges or demands will involve the long tail of sparsly connected nodes.

Figure 5 shows the experimental results for increasing number of flow demand pairs. The x-axis is the average number of destination demands for each node of the network. For example, demand density of 8 means that, on average, each node in the network wants to send flow to 8 destinations. The y-axis is the sum of lengths of all paths used from successful flows, divided by the sum of all shortest paths that would be used for all successful flows for an cost-optimal routing solution

The figure also includes the amount of successful flow as a percent of the total flow demand that could actually be routed in the network. As we generate random graphs and random st-pairs, it is possible that there are no directed paths between some of the generated st-pairs and therefore no flow could be routed through those paths anyway. These origin-destination pairs are ignored.

Here there is a clear downward trend of path dilation, as we increase the demand load on the network for random Erdös-Renýi graphs, while powerlaw graphs show very little sensitivity to the demand

density. The amount of flow routed is also far less sensititve for powerlaw graphs. As we mentioned above, we do expect this behaviour, as a result of the structure of powerlaw graphs. When we add new demand pairs in the network, we are more likely adding origin-destination nodes in the much larger, less connected part of the network.

For Erdös-Renýi graphs, increasing demand density results in better paths. This is a result of more deamnd for routing flow in the network, which increases the probability of having nodes on the shortest paths that are willing to relay flow to keep edges open for themselves. This is less probable in powerlaw graphs, as most of the graph is not well connected and adding more demand pairs will mainly involve nodes in the less-connected tail nodes.

Figure 6 shows the experimental results for increasing number of edges per node. The x-axis is the average out-degree for each node of the netowrk. For example, average degree of 10 means that, on average, each node in the network has 10 neighbors to which they can shoose to send flow. The demand density for these experiments is 8, and the network has 50 nodes.

Here the results are quite different. Both powerlaw and Erdös-Renýi graphs behave in a more similar manner, with the path dilation increasing with the average degree. Again, the effect is marginal for powerlaw graphs- we do not see any significant change when we increase the edge density of the graph. For Erdös-Renýi graphs, path dilation now increases with edge density. When we increase the number of edges in the network, the shortest paths are expected to decrease for Erdös-Renýi, and less so for powerlaw graphs, for the same reasons as mentioned above (additional edges connect the larger, less dense part of the network and are more likely to have less effect, as short paths are more likely to involve the very few, very popular nodes).

For the range of parameters used in the experiments, the variance of the path dilation values is relatively low in Erdös-Renýi graphs, and higher for powerlaw graphs.

Figure 7 shows a histogram for path dilation for powerlaw graphs. The values come from 100 random graphs with 50 nodes, edge and demand density 8.

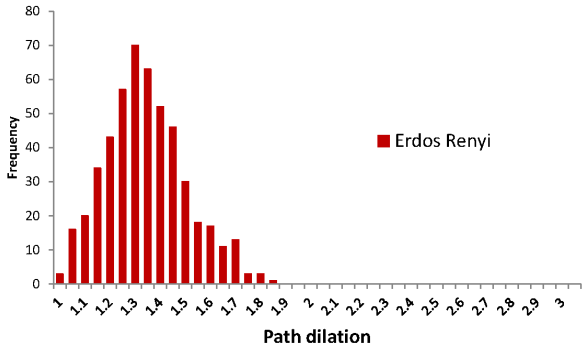

Figure 7: Histogram of the path dilation values for Erdös-Renýi G_{nm} graphs

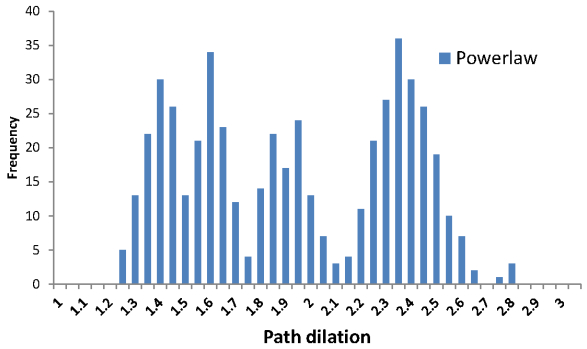

Figure 8: Histogram of the path dilation values for Erdös-Renýi G_{nm} graphs

The values are clustered around the mean, and are bounded below 2. Figure 8 shows the corresponding histogram for Erdos Renyi graphs. Here the values show higher variance and mean, still bounded below 3.

6 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

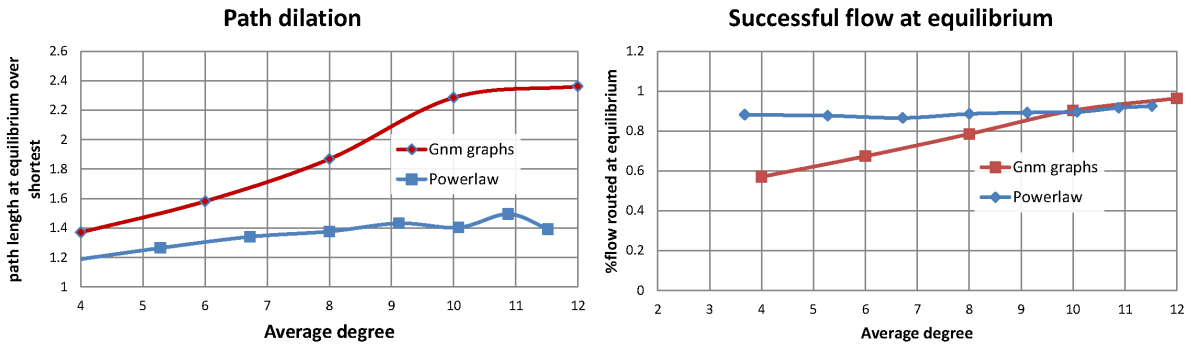

Figure 6: Path dilation and percent of available flow routed at equilibrium for Erdös-Renýi G_{nm} graphs and for Barabasi-Albert powerlaw graphs as a function of edge density, the average number of out-edges for each node

can carry a significant fraction of the available flow, and that the resulting paths are not much longer than the shortest ones.

For a relatively wide range of densities of both edges and demands, arguably the powerlaw networks behave in an efficient and very stable manner at equilibrium. Path dilation is usually less than 1.4 and we expect at least 80% of the available flow to be routed successfully, without the need of any central control, or any payment scheme.

For Erdös-Renýi graphs, we see a much higher sensitivity to both parameters. High edge density makes the network more successul, in the sense that more flow is routed, but this results in longer, more inefficient paths. For demand density, things are very different. Higher demand density makes the network more successful (more flow routed) and more efficient (paths are shorter at equilibrium)

References

[1] Sonja Buchegger and Jean-Yves Le Boudec. Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes : Fairness In Dynamic Ad-hoc NeTworks). In *Components*, volume 48 of *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 226–236. Citeseer, 2002.

[2] M Felegyhazi, J P Hubaux, and L Buttyan. Nash equilibria of packet forwarding strategies in wireless ad hoc networks, 2006.

[3] Qi He, Dapeng Wu, and Pradeep Khosla. SORI: a secure and objective reputation-based incentive scheme for ad-hoc networks, 2004.

[4] George Karakostas and Euripides Markou. Emergency Connectivity in Ad-Hoc Networks with Selfish Nodes. *Discovery*, pages 350–361, 2008.

[5] Ratul Mahajan, Maya Rodrig, David Wetherall, and John Zahorjan. Sustaining Cooperation in Multi-Hop Wireless Networks. *Symposium A Quarterly Journal In Modern Foreign Literatures*, pages:231–244, 2005.

[6] Pietro Michiardi and Refik Molva. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In B Jerman-Blazic and T Klobucar, editors, *Conference on Communications and Multimedia Security*, volume IFIP 228 of *Advanced Communications and Multimedia Security. IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, pages 107–121. Kluwer Academic Pub, 2002.

[7] Fabio Milan, Juan José 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, page 3, 2006.

[8] V Srinivasan, P Nuggehalli, C F Chiasserini, and R R Rao. Cooperation in Wireless Ad Hoc Wireless Networks. In *IEEE Infocom*, page 2009, 2003.

[9] 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.