Strategic Network Formation through Peering and Service Agreements

Elliot Anshelevich [*] Bruce Shepherd [†] Gordon Wilfong [‡]

April 2006

Abstract

We introduce a game theoretic model of network formation in an effort to understand the complex system of business relationships between various Internet entities (e.g., Autonomous Systems, enterprise networks, residential customers). This system is at the heart of Internet connectivity. In our model we are given a network topology of nodes and links where the nodes (modeling the various Internet entities) act as the players of the game, and links represent potential contracts. Nodes wish to satisfy their demands, which earn potential revenues, but nodes may have to pay (or be paid by) their neighbors for links incident to them. By incorporating some of the qualities of Internet business relationships, we hope that our model will have predictive value. Specifically, we assume that contracts are either customer-provider or peering contracts. As often occurs in practice, we also include a mechanism that penalizes nodes if they drop traffic emanating from one of their customers.

We first show that every Nash equilibrium can be represented by a flow of utility with certain constraints. Apart from helping to visualize the general structure of stable solutions and providing useful proof techniques, this yields a polynomial time algorithm for finding Nash equilibria that successfully route a specific set of demands. We then prove that the prices of anarchy and stability can both be unbounded in this context. This leads us to consider how much we must perturb the system to obtain good stable solutions. We thus focus on the quality of Nash equilibria achievable through *centralized incentives*: solutions created by an "altruistic entity" (e.g., the government) able to increase individual payouts for successfully routing a particular demand. We show that if every payout is increased by a factor of 2, then there is a Nash equilibrium as good as the original centrally defined social optimum. Under stronger conditions, we also exhibit factors better than 2. We show how to find the best Nash equilibrium in polynomial time when the underlying network is a multicast tree, as well as in some other special cases. Finally, for a natural objective function other than social welfare, we prove that the price of stability is at most 2.

[*]Department of Computer Science, Princeton University, Princeton NJ. Email: eanshele@cs.princeton.edu. This research supported in part by ITR grant 0311333, Bell Labs, and the NSF Postdoctoral Fellowship in Mathematical Sciences.

[†]Bell Labs, Murray Hill, NJ.

[‡]Bell Labs, Murray Hill, NJ.

1 Introduction

The formation of the Internet marked a step away from centrally planned and controlled networks to networks that employ distributed traffic routing control. Nevertheless, the early Internet adopted a common standard and agreed upon metrics to make routing decisions. With the advent of competition in the 1990's, this ceased to be the case, and today the Internet is composed of tens of thousands of sub-networks called *Autonomous Systems* (AS), each under a single administrative authority with its own distinct goals in controlling the traffic entering and leaving its network. A number of other emerging communication networks also have the characteristic that a collection of domains, with varying self-interests, participate in such a multilateral sharing of network resources.

Network management becomes substantially more complex as a result of the inherent interdependence involved in multi-domain (multilateral) networks. For instance, there is limited ability to predict how changes in the network, or in business relationships between domains, affects current routings [19]. Even obtaining accurate estimates of current traffic conditions is a nontrivial challenge [6, 14]. Such immediate operational tasks in turn depend on the existence of a stable system of business relationships between AS's. Questions of whether certain AS's will peer and what type of *service level agreements* (SLAs) will be forged between AS's are critical for understanding the structural properties of the networks formed. This paper's focus is on these longer-term strategic factors affecting the formation of a stable network-of-networks.

It is natural to employ game theory to analyze the self-interested behaviour of domains, and several models have recently been proposed. These have fallen into two broad categories: models that address routing issues (e.g., [13, 36]) and those that study network creation (e.g., [1, 12, 30]). Our objectives are more aligned with the latter class. In particular, we introduce a network formation model called the *Local Contract Formation Game* (LCFG) in an effort to understand the complex system of business relationships between Internet domains such as AS's, residential customers, and enterprise networks. This system is at the heart of Internet connectivity and by incorporating some of the qualities of these relationships, we hope to capture the essence of these interactions so that our model will have some predictive value. We model three key elements of real-world business interactions between domains. The first element is that, unlike many models, we assume that monetary transfers and business relationships are strictly local. This models current practice where links arise as part of a bilateral agreement between the two endpoints. Such arrangements depend implicitly on the global structure of the networks and traffic demands, but are based only on a *local* bid-ask type contract between two neighboring domains [26, 27]. Second, we allow the links in the network to be one of two types: customer-provider or peer-peer [10, 11, 15, 20]. Third, we include a mechanism that penalizes domains if they drop traffic emanating from one of their customers. This models the fact that SLA penalties have become commonplace in contracts, especially those offered by core network providers [25]. Despite these features, our model remains simple enough to analyze.

In our model, we assume that there are no link capacity constraints. In other words, links have been sized accordingly to carry all possible traffic; a reasonable assumption given our focus on long-term effects, as opposed to brief outages due to traffic bursts. Also, we assume that traffic demands have a specified path they must follow. While this *fixed route* assumption ignores dynamic routing changes in a network, it seems a necessary first step to understanding whether a given configuration of business relationships is stable. Moreover, to understand routing behavior, we must first understand how/whether the links used by those routes would themselves form a stable configuration. The model choice is also driven and justified by our focus on stable interdomain business relationships, which are longer-term than routing decisions. As we will see, the fixed route model already contains considerable complexity. Future work may be to extend this to allow route generation and contract formation as a repeated game, where routes depend on the previous contracts, and contracts depend on the routes during the previous step.

We study stable business relationships by exploring the existence and structure of Nash equilibria for the game LCFG. Nash equilibria are the dominant solution concept in game theory, and correspond to locally

optimal solutions. While solution concepts are possible in this context, if local optimality is not satisfied, then the system is inherently unstable since some node (player) could act to increase their payoff. Our goals are to understand the creation of such stable networks, and how to induce good stable solutions when none would otherwise form. Interestingly, we will see there are several attributes of Nash equilibria in LCFG that match common practice in the Internet. One such is that there always exist equilibria where no money is exchanged between peers. Another is that even without any assumption on the structure of the underlying contract graph, one may assume that no node forwards traffic from one of its providers to another. This mirrors the practice of filtering out route announcements in the Border Gateway Protocol.

Local Contract Formation Game In LCFG we are given a network topology of nodes and links where the nodes (modeling domains) act as the players of the game, and the links represent potential connections or contracts that could be made. We are also given a set of demands each of which is a path in the graph. For example, in Figure 1(a), there is a demand d_1 along the path between u_1 and u_2. A demand d_i is *active* (connected) if appropriate contracts are formed on all links in the path of d_i (i.e., the links are "activated"). If d_i is active, the utility of d_i's endpoints increases (say by some value λ_i). For every active demand going through a node v, however, v suffers 1 unit of disutility, representing the fact that it costs money to transit traffic through one's servers and subdomains. We call this cost the *transit cost*. Figure 1(b) shows the change in the nodes' utility because demands d_1 and d_3 are active. The goal of each player v in this game, then, is to satisfy as many of the demands ending at v as possible, while having as few demands as possible going through v.

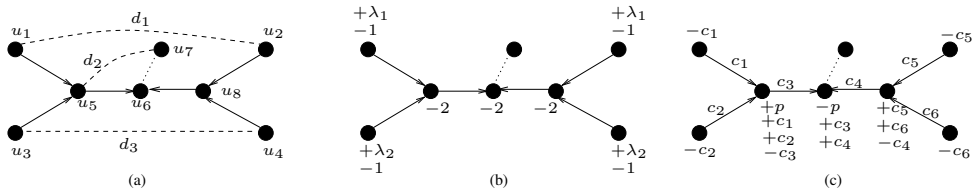

Figure 1: Utilities and transfers. Directed edges represent customer-provider contracts, with the provider at the head. The dashed edge (u_6, u_7) represents an inactive link/contract.

In such a framework without utility transfer, demands would never be activated (i.e., connected), since players have no interest in incurring transiting costs for other players. When bilateral contracts are introduced, however, two players may activate a link between them and thus commit to carrying traffic offered across the link. In fact, two types of contracts are possible: a *customer-provider contract* where one player (the customer) pays the other player (the provider) to activate the link and transit the customer's traffic; and a *peering* contract where no payment is exchanged because it is in both players' interests to activate the link. Figure 1(c) shows the payments c_i that occur because of the formation of these contracts, as well as the impact of these payments on the players' utilities.

In addition, suppose a provider accepts payment from a customer (i.e., a contract is formed between them). The customer agrees to this payment since it then expects the provider to form the necessary contracts with its other neighbors to activate demands from the customer (or the customer's customers etc). If the provider fails to make such a payment for a link, then the provider must pay a "penalty" to its customer. Figure 1(c) shows u_6 paying a penalty p to u_5, since u_6 is the provider of u_5 and u_6 failed to form a contract with u_7 that would have activated demand d_2.

Thus a player's total utility consists of the utility for having demands activated for which it is an endpoint, payments to or from neighbors for contracts formed, transit costs for active demands that go through it and finally, penalties paid by or to it. A node's strategy space then consists of choosing which contracts to form and how much to ask/offer for the formation of such contracts. A node might change its strategy if by doing so it strictly increased its utility. For example, in Figure 1, if node u_6 changed its strategy and paid some

amount c to node u_7 to activate the link between them, then u_6's utility would increase by p since it would no longer pay a penalty, but it would decrease by $c+1$ due to the payment of c plus it would incur a transit cost of 1 for the now active demand d_2. Thus if $p-c-1>0$, then u_6 would benefit by changing its strategy.

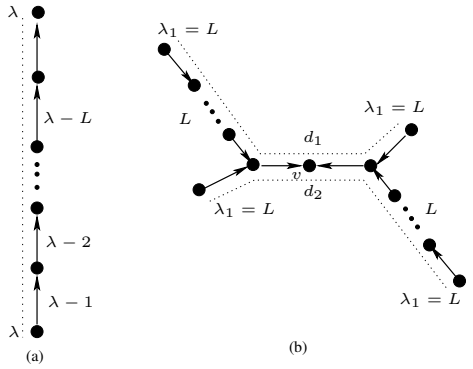

Figure 2: (a) The payments for customer-provider contracts are shown. (b) Cooperation results in a good NE.

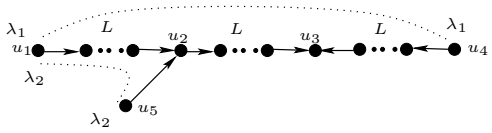

Figure 3: NE with distant monetary effects.

Nash Equilibria (NE) Formal definitions of LCFG and its NEs are given in Section 2, but here we informally illustrate some interesting forms of NEs. Notice that in any NE each player must have non-negative total utility for otherwise it could improve its utility to 0 by requesting a large ("infinite") amount for each of its incident edges effectively deactivating or "cutting" these edges. Also, a trivial NE where no demands are activated can always be achieved by having all nodes cut all of their incident edges. In this way, all nodes end up with a total utility of 0.

In Figure 2(a), we have a single demand, of utility λ, through $L+2$ nodes. One may argue that this is a NE if $L \leq \lambda - 1$. Informally, each internal node would keep (at least) 1 from its incoming payment to pay for its transit cost, before passing on any remaining payment to the provider above it. If each node keeps exactly 1, then the payments along edges are shown in the figure. Assuming the ask prices of the providers do not exceed this amount, then the edges will be formed and the demand will be active as long as $L \leq \lambda - 1$. Note that the total utility for each internal node (contract payment received minus contract payment paid minus 1 for transit) would be 0. If a node cuts its outgoing edge, then it will have to pay a large penalty (in this case the penalty would actually be at least $\lambda - 1$) which is at least as large as the payment it is receiving. Thus this configuration would in fact also be stable.

In a sense, for this example there is sufficient utility from the bottom node of the demand to activate all of the edges. This suggests a likely candidate for a stable solution: one where each demand pays for its own transit. That is, a flow of payments is propagated from the endpoints along the demand path in order to ensure that the internal nodes are willing to incur the transit costs. There are NEs, however, as in Figure 2(b), where demands pay for each other's transit and improve the overall utility as a result. In this example, demand d_2 has plenty of money to pay for its transit on the left of node v, but requires help to pay for transit on the right, with the opposite being true for demand d_1. Both demand paths are of length $L+4$.

More surprisingly, there are NEs where payments from a demand end up at nodes distant from that demand's connection path, as in Figure 3. Here λ_1, λ_2 are approximately L, and it can be shown that a NE

exists with all the edges active. In such a NE, all the payments starting at u_1 towards u_2 would be used to pay for the transit of the two demands through those nodes from u_1 up to u_2. There is a payment from u_5 to u_2 of $\lambda_2 - 1$ which is roughly L. Thus there is about L utility available at u_5 to pay for the transit of d_1 along the path from u_2 to u_3. It is beneficial for u_2 to send the money from d_2 towards u_3 to pay for d_1's transit (i.e., pay for the provider edge from u_2 to the next node) since otherwise it would have to pay a large penalty to its customer node along the path from u_1. It would not have to pay this penalty if it also cut the edge from that customer but then it would have to pay a penalty to u_5 since that would deactivate d_2. These examples show that paying for the transit of others can result in a NE that maximizes social welfare.[1]

Our Results Figure 2(a) with $L > \lambda$ shows a simple example, where the only NE is *trivial*, with no edges installed, and hence the total social welfare is 0. In general, the prices of anarchy and stability[2] in LCFG can both be unbounded. Moreover, as we show in the Appendix, the best (maximum social welfare) NE is not poly-time approximable. This motivates the questions of when good Nash equilibria exist, and how to *induce* a good NE in an instance of LCFG given a small budget, without altering the game mechanism. Some of our results are as follows.

- In Section 3, we provide a structural characterization of Nash equilibria that is also the basis for our subsequent technical results. We prove that every Nash equilibrium can be represented by a flow of utility with certain constraints. Apart from helping to visualize the general structure of stable solutions, this yields a poly-time algorithm for determining the existence of Nash equilibria that successfully route a specific set of demands.
- In Section 4 we focus on measuring the degree to which an instance must be perturbed in order to obtain a Nash equilibrium as good as the centrally defined social optimum. Equivalently, one may view this as seeking high quality NEs achieved through *centralized incentives*; that is, solutions created by an altruistic entity able to increase individual payouts for particular demands. This entity may be the government, or any organization interested in the quality of the overall network. Our main result of this section shows that if every demand value $\lambda_s(d)$ is increased by a factor of 2, then there is a NE as good as the original centrally defined social optimum. We also prove various generalizations of this result, that give us factors less than 2, and in some cases, close to 1.
- Instead of the social welfare objective, in Section 5 we consider a non-mixed sign objective function: the total transit cost incurred plus the value of the demands that are not connected. With respect to this (still natural) objective function, we show that the price of stability of LCFG is at most 2, as well as how to efficiently find a good Nash equilibrium starting with an approximation to the centralized optimum.
- Finally, in Section 6 we show that if the routing network is a multicast tree (all demands have a node in common), then we can find the best Nash equilibrium in polynomial time. Unlike our other results, this problem becomes NP-hard if demands are not unit-size. In the case of non-unit demands, we also show in Section 6 how to efficiently find an interesting non-trivial Nash equilibrium in a multicast arborescence, where all the edges are directed towards the sink.

Related Work The notion that an AS benefits from its demands getting to their destinations, and loses utility from transiting traffic, was explored in several papers (e.g. [13, 30, 34]). Typically, however, the concentration has been on short-term routing and pricing schemes, e.g., [22, 24, 29, 32, 35]. While short-term pricing and admission models are of significant interest, they rely on an underlying set of business (longer-term economic) relationships. Moreover, agreements between entities (ISPs, enterprise or residential customers) tend to be based on more rigid contracts such as fixed bandwidth, or peak bandwidth contracts [26, 27]. This is largely due to the complexity (and expense) of monitoring IP traffic at a packet or flow level.

[1] Recall that the social welfare of a solution is the sum of the players' utilities.

[2] See [31] for a definition of price of anarchy (known there as the coordination ratio), and [2] for a definition of price of stability.

Motivated by this, several game theoretical models have addressed the strongly related notion of network formation. Some of these do not look at contract formation, but instead assume that edges have intrinsic costs [1, 2, 3, 4, 12, 23, 33]. On the other hand, contract formation models of networks have been heavily addressed as well, mostly in the economics literature (e.g., [16], for a survey see [28]). This body of work mostly addresses questions distinct from those studied here. In particular, none of these consider customer-provider and peering contracts, or measure the impact of provider penalties. In [5] a very general model of network formation is considered. Our model is a (very) special case of theirs, and indeed their characterization of "stable networks" is analogous to our characterization of a stable set of strategies in LCPG. The model of [30] is also quite relevant; their flavor of results is quite different from ours, however, as they focus on solutions where all demands are satisfied, and on so-called pairwise-stable equilibria. In addition, [7, 29] address intra-domain concerns of when it is wise to form peering contracts, instead of concerns about the resulting overall network structure.

2 Model and Basic Results

We now formally define an instance of the Local Contract Formation Game (LCFG). We are given a mixed graph $G = (V, E)$, as described in the Introduction. Undirected edges represent possible peering contracts. A directed edge (arc) $e = (u, v)$ is referred to as a *provider edge* of u and a *customer edge* of v. Graph G could in fact be a multi-graph, that is, there may be several types of contracts possible between two given nodes. Notice that the direction of an edge is not meant to indicate traffic flow, but to represent the hierarchical nature of a typical customer/provider network (cf. [15]).

We are also given a set of *demands* D on G, each specified with an unordered pair of nodes st. A demand d is a request for traffic to be exchanged between s and t; to limit notation, we assume each demand is unit size, although in most cases extending our results to demands of variable sizes is easy.

Traffic for a demand d is carried on some fixed, pre-specified path $P(d)$ with endpoints s, t. A path from s to t in G is said to be an *upward path* if it is a simple directed path. We call a simple path P between s and t a *valid path* if P can be written as the concatenation of three subpaths $P = P_1P_2P_3$ where P_1 is a (possibly empty) upward path from s to some node u, P_2 is an empty path or a peering edge between u and some node v and P_3 is a (possibly empty) upward path from t to v. This definition of a valid path is the same as the definition of Type-1 and Type-2 paths in [15], valid signaling paths in [20] and valley-free paths in [10]. For the remainder of this paper, we assume that all our paths $P(d)$ are valid. See the end of this section for why this assumption is the correct one to make.

Notice that in the above definition $P(d)$ specifies exactly which contracts must be active for demand d to be satisfied. In fact, all our results still hold if we relax this condition, and say that demand d is satisfied when there exists any valid path from s to t on the nodes of $P(d)$ (in other words, the route is fixed, but not the contracts along this route).

A *configuration* of our game is determined by a set S of *active edges* (i.e. successfully formed contracts). A demand d is *active* (i.e., successfully routed) in this configuration if all edges of $P(d)$ are active. Let $D^{end}(v)$ be the active demands having endpoint v, and $D(v)$ be the set of active demands for which $v \in P(d)$. Similarly, for each edge e, define $D^{end}(e)$ to be the active demands having e as an initial or final edge, and $D(e)$ to be the set of active d with $e \in P(d)$. In general, for any $S \subseteq V \cup E$, $D(S)$ is the set of active demands that include nodes/edges of S. We denote by $\lambda_s(d)$ the *value* of the demand d to s if d is active and s is an endpoint of d. For any node v, the set of active edges determines a total value to v of: $\sum_{d \in D^{end}(v)} \lambda_v(d)$. In the following, we always assume that $\lambda_v(d) \geq 1$.

The basic goal of nodes (players) is to influence the formation of contracts so as to balance their desire for the values of connected demands with the cost incurred by transiting the traffic of other nodes. We now discuss the basic strategies that players have at their disposal to achieve their goals.

Strategies: A *strategy* for a node v consists of *bids* for each of its incident edges e. We denote by $\text{offer}_v(e)$ the amount v is offering (or demanding if $\text{offer}_v(e) < 0$) to form the business relationship represented by e. Similar to [5], a contract on edge e with endpoints u and v is *formed* if $\text{offer}_v(e) + \text{offer}_u(e) \geq 0$, the payment offered by one endpoint of e is greater than the payment demanded by the other endpoint. In this case e is *active* for these strategies. This is slightly different from [5], since if one of our bids is negative, then the other endpoint transfers an amount equal to its absolute value. If both endpoint bids are nonnegative, then the edge is formed with no payments between u and v. We let $c_u(e)$ denote the ultimate payment made to u for e. A *solution* $\mathcal{S}$ for this game consists of a profile of strategies, i.e., for each node a list of bids for its incident edges. Clearly $\mathcal{S}$ induces a network configuration S given by the set of active edges.

Transit Costs: For any configuration of active edges, say induced by a solution $\mathcal{S}$, define the *traffic transited* by a node v (or similarly by an edge e) to be $D(v)$ (or $D(e)$). For $x \in V \cup E$, let $t(x)$ be the total amount of traffic that x is transiting, i.e. $t(x) = |D(x)|$. We define the *cost of transiting* for a node x to be $t(x)$. That is, there is a normalized cost of 1 for each active demand transited. In general, for any $S \subseteq V \cup E$, let $t(S) = \sum_{s \in S} t(s)$.

Penalties: Finally, a provider must pay penalties to its customers if it fails to meet its obligations. The existence of such penalties is the only difference between peering and customer-provider edges in our model. For node v and demand d with $v \in P(d)$, define $p_v(d) = 0$ except in the following cases. Consider a demand d with endpoints s and t. The demand is *penalty-enabled* for v if there is a nonempty active directed subpath of $P(d)$ from an endpoint of $P(d)$ up to v, the last edge obviously being a customer-provider edge $e = (u, v)$. Assume the endpoint of $P(d)$ in this subpath is s. If v fails to form a contract activating the next edge of $P(d)$ for a penalty-enabled demand d, then v pays a penalty $p_v(d) = \lambda_s(d) - 1$ to u, and symmetrically $p_u(d) = -p_v(d)$. This models the fact that a customer of v wants to send traffic on $P(d)$, but v is unable to activate its incident edges in $P(d)$, thereby failing in its provider duties.

Note that our results hold with other penalty models as well, such as penalizing a provider v for lost demands, even if the "culprit" edges are not incident to v. Intuitively, a node s should be compensated for loss of any λ-value from one of its demands d routed through one of its (paid for) connections to a provider. The benefit to s is $\lambda_s(d) - 1$ (since s receives $\lambda_s(d)$ value if d is active but has a cost of 1 for transit of d). If penalties were any smaller, there would be instability in the system, since providers would prefer to pay a penalty instead of forwarding a customer's traffic. On the other hand, our results, including hardness results, still hold if the penalties were allowed to be higher.

The Utility Function: Given the utility for an active demand, the transiting costs, and the penalties, the utility of node v (for a solution $\mathcal{S}$) is $\textit{utility}(v)$ (or $\textit{utility}_{\mathcal{S}}(v)$), as follows.

$$\textit{utility}(v) = \sum_{d \in D^{end}(v)} \lambda_v(d) - t(v) + \sum_e c_v(e) - \sum_d p_v(d) \tag{1}$$

Equation (1) may seem complicated, but its components are quite intuitive. A node v gains the value of $\lambda_v(d)$ for each active demand that it originates, loses 1 for every demand it transits, gets payment $c_v(e)$ according to the contract it makes with its neighbor on e (either positive if v is paid or negative if it pays) and loses $p_v(d)$ for penalties (either positive or negative depending on whether it pays or receives the penalty).

Nash Equilibria Given a solution $\mathcal{S}$, a *deviation* for a node v is a solution where v changes its strategy while all others remain as in $\mathcal{S}$. A *best deviation* for a node v is a deviation for v that results in the highest payoff to v over all possible deviations for v. In the case that a best deviation for v is to stay with its original strategy, we say that v is *stable*. If all nodes are stable in a solution $\mathcal{S}$, then $\mathcal{S}$ is said to be a *Nash equilibrium* (or NE). We call a NE *nontrivial* if it has at least one active demand (and trivial otherwise). We say that S, a set of edges, *induces* a NE when there is a NE whose set of active edges is S.

Discussion of Filtering and Valid Paths Here we discuss the "no-filtering" and valid path assumptions in our model. An important property in our model is that strategies of a node consist of the amount of money

it is offering/demanding for various connection agreements (i.e., edges). It is not part of a node's strategy to decide which demands it will transit; it must transit all active demands that pass through it. At first glance, this might seem unrealistic, since AS's can do anything they desire with traffic, including not forwarding it or filtering away particular packets. However, consider the case where we drop these restrictions, allow demands to follow paths that are not valid and allow arbitrary filtering of traffic. When would it be in the interest of node v to not transit an active demand $d \in D(v)$? If v is an endpoint of d, then v does not lose anything by transiting d, since we assume $\lambda_v(d) \geq 1$. If $d = st$ does not originate at v, then there must be two edges of $P(d)$ incident to v. If at least one of these edges is a customer edge, then v would have to pay a penalty of $\lambda_s(d) - 1$ (or $\lambda_t(d) - 1$) to its customer for not transiting d. But then the penalty would be at least as much as the cost to transit so there is no gain in filtering such traffic. In fact, the only time when v would gain by filtering a demand d is when the two edges of d incident to v are both non-customer edges. In this case, v can refuse to transit d, save itself the transit cost, and not lose any utility since it has no customers that it would owe penalties to. In fact, this is exactly the type of demand (route) filtering that is done in the Internet today [11, 15]. Because it is *always* in v's interest to filter in such a case, we can simply assume that all demands with such routes have been filtered out, which is equivalent to assuming that all demands follow valid paths and that additional filtering is unnecessary.

2.1 Basic Results and Useful Observations

In our model a trivial Nash equilibrium always exists. A solution where all players v set all $\text{offer}_v(e)$ to a large negative number results in no active edges, and is a Nash equilibrium with all nodes having a utility of 0. Moreover, for every Nash equilibrium there is an equivalent one where the payments demanded on inactive edges are infinite. Without loss of generality, we assume from this point on that this holds for all inactive edges in any stable solution we consider. Thus we can now think of deviations as a node "cutting" edges since forming extra contracts is not an option for a single player in such a solution. Cutting is achieved either by a customer offering less money or a provider requiring more money.

Intuitively (and in practice) money is paid from customers to providers (not the other way around) and peering connections typically involve no money changing hands at all. In our model definition, we did not enforce this to be so. However, this property is inherent in our definition of the players' utilities, as the following proposition illustrates (all proofs not given immediately can be found in the Appendix).

Proposition 2.1 *For every Nash equilibrium A, there exists another Nash equilibrium B with the same active edges, and all strictly positive payments only being paid from customer to provider.*

Thus we now only consider NEs where positive payments are never made to a node's peers or customers. To simplify notation, we let $c(e)$ be the nonnegative payment on e from the customer to the provider.

Price of Anarchy and Stability An *optimal centralized solution*, that we denote by OPT, is a configuration that maximizes the social welfare function $\sum_{v \in V} \textit{utility}(v)$. Notice that all the contract payments and penalties cancel out, so this objective function is just $\sum_{d=st \in D(V)} (\lambda_s(d) + \lambda_t(d) - \|P(d)\|)$ where $\|P(d)\|$ is the number of nodes in $\|P(d)\|$. We say that a solution is a *best Nash equilibrium* if it maximizes the social welfare function over all possible Nash equilibria. If S is the set of active edges in a solution $\mathcal{S}$, then we use the notation $W(S)$ or equivalently $W(\mathcal{S})$ to denote the social welfare of $\mathcal{S}$.

The example in Figure 2(a) with $L = \lambda$ shows that there are instances of LCFG in which a best Nash equilibrium has no active demands and hence 0 social welfare, whereas OPT has non-zero social welfare. This implies that the price of anarchy [31] and price of stability [2] can both be infinite in LCFG. In this example, all the edges are active in OPT, so $W(OPT) = 2L - (L + 2) = L - 2$. However, there is no nontrivial NE in this instance. To see this, note that in a NE, each node must have non-negative utility. In order to cover the cost of transit, each internal node must keep at least 1 from any payment from their

customer. This cannot happen since there are $L+1$ such nodes, and the total payment offered by the bottom node is at most L.

Our focus now is to study when nontrivial Nash equilibria exist, and how good they can be in terms of social welfare. Unfortunately, we can prove the following theorem.

Theorem 2.2 *Finding a nontrivial Nash equilibrium is NP-complete even when G is an arborescence. Moreover, there is no polynomial-factor approximation algorithm for finding the best Nash equilibrium.*

See Appendix B for the proof of this and other hardness results, [and if all 3 contracts are available if one is] such as Theorem 6.4, which states that OPT is inapproximable up to n^{ε} even in star networks. Theorem 2.2 shows the intractability of finding NEs of any value, let alone close to $W(OPT)$. There may also not exist a good quality approximate NE (where players only deviate if they substantially improve their utility) because of the example in Figure 1(a). This drives our focus on how to add incentives to achieve a good NE (say one as good as the original OPT).

3 Nash Equilibrium As A Flow of Payments

In Section 4 we show how to induce good Nash equilibria through limited centralized incentives, and in Section 5 we show a price of stability bound for a different objective function. In this section we build up some techniques, and in particular we give a result that helps visualize the "movement" of money in a Nash equilibrium. It is also an essential tool for our later proofs.

In the following, we assume that S is a fixed set of edges and we look for node strategies (i.e., values $c(e)$) that induce a NE with S as the active set of edges. The basic framework is as follows. We define a flow problem in a graph $G(S)$ such that every feasible flow in $G(S)$ corresponds to a stable set of payments $c(e)$. Such "Nash flows" are obtained by stitching together n circulation problems, one local to each node. For any fixed set of edges S, our results yield a compact LP formulation for the polyhedron of Nash equilibria (viewed as payments on edges). Hence we may also optimize to find "best" Nash equilibria under any linear objective function of payments.

Before formally defining the node circulation problems, we give some intuition on the constraints for a stable set of payments. Specifically, for every node v, think of each edge incident to v as a separate entity that has a budget equal to the utility it is bringing (or taking) from v. It may possibly use its budget to pay for its own transit or to lend to other edges. We show that a critical property of any NE is that edges cannot lend too much. We call this the *bounded lending constraint*, and it essentially represents the fact that the only thing keeping a node from cutting an edge is the penalty it would incur. To understand the bounded lending constraint (defined rigorously below), note that if a demand d is the only demand that goes through node v, and v is paying more than $\lambda_d - 2$ to its provider, then v would rather cut its provider edge, since it would incur a penalty of $\lambda_d - 1$, but its utility would increase because it would not longer have to pay its provider, and would not incur d's transit cost. In this case, for node v to be stable, v's customer edge should not "lend", or transfer, more than $\lambda_d - 2$ utility to v's provider edge. This kind of reasoning puts bounds on how much lending is allowed, and the following Remark states that a node is stable exactly when the payments on its incident edges can be expressed in the above manner.

Remark 3.1 *If for all v, its incident edges can pay their transit costs in v, and no edge exceeds the* bounded lending constraint *(defined below), then the payments form a Nash equilibrium. Conversely, every Nash equilibrium can be expressed in this manner.*

Nash circulations and flows. We start by forming a circulation problem at a given node v. The problem instance for v has a node v_e for every active edge e incident to v, as well as an extra node s that represents

the "outside world" (see Figure 4 where 4 copies of the same node s are displayed). The node s is the origin of all utility obtained by v, as well as the sink of all utility v must spend on transit costs, provider payments, and the utility v keeps as profit.

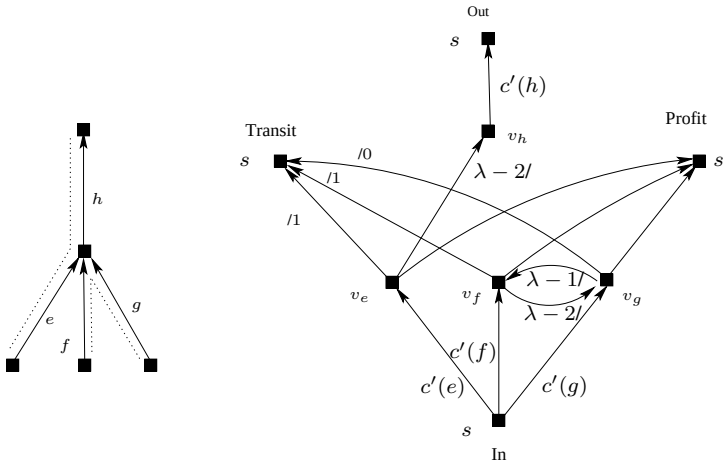

Figure 4: (Left) Node v with customer and provider edges. (Right) The corresponding circulation gadget, with the 4 s-labeled nodes being the same. Edges are labeled as "capacity/lower bound".

For each customer edge e, we introduce an arc (s, v_e) and the flow on this arc, denoted by $c'(e)$, represents the utility that v perceives as coming from e. This includes the payment $c(e)$ given to v, as well as values for any active demand d originating at v, i.e., $d \in D^{end}(e)$. Similarly, if v is paying a penalty $p_v(e)$, then v has extra incentive to cut e. Thus, $c'(e) = c(e) + \sum_{d \in D^{end}(e)}(\lambda_v(d) - 1) - p_v(e)$ is the utility that v sees coming from e. Note that a given flow $c'(e)$ uniquely determines the payment $c(e)$, and vice versa.[3] Similarly for each provider edge, we add an edge (c_f, s) where the flow $c'(f)$ on this edges represents the utility v perceives as leaving on a provider edge f. For a given payment $c(f)$, we have the equality $c'(f) = c(f) - \sum_{d \in D^{end}(f)}(\lambda_v(d) - 1) - p_v(f)$.

Transit: For any active demand $d \in D(v) - D^{end}(v)$, we assign one of the edges incident to v to *own* the demand. A provider/peer edge will never own a demand, so if d uses a customer edge e as well as a provider/peer edge, then we set e to be the owner. Otherwise, it uses two customer edges e, f and we pick an owner arbitrarily. Now for each customer edge e, we include an arc (v_e, s) with a lower bound equal to the number of demands it owns. These arcs represent utility lost due to transit.

Bounded Lending: For any active pair of edges e, f incident to v, let $P(e, f)$ denote the set of penalty-enabled demands where v would have to pay a penalty to the other endpoint of e if f were not active. Then, we form an edge (v_e, v_f), with capacity $\sum_{d \in P(e,f)}(\lambda_u(d) - 1) - O(e, f)$, where u is the endpoint of d below e, and $O(e, f)$ is the number of active demands in $P(e, f)$ that e owns. Thus edges are allowed to transfer money to each other, but only after paying for the transit of the demands they own. We call this the *bounded lending constraint.*

Finally, to represent profit which is kept by node v, for every node v_e we add an extra edge (v_e, s), with lower bound 0 and infinite capacity. Figure 4 illustrates this point with arcs into the "profit copy" of node s. Note that $c'(e)$'s can actually be negative if $p_v(e)$ is large. Thus in fact, we would also include reverse arcs (v_e, s) in the construction and let $c'(e)$ be the flow on (s, v_e) minus that on (v_e, s).

Lemma 3.2 *Node v is stable with payment vector c if and only if there exists a circulation where the flow on corresponding edges is c'.*

[3] If we end up with some $c(e) < 0$, it just means that v is stable, even with $c(e) = 0$.

Proof Sketch. We apply Hoffman's circulation theorem to the circulation problem above, where $c'(e)$ is both an upper and lower bound of (s, v_e). Then the utility lost by v from cutting a set of edges S is exactly the sum of upper bounds of edges in the circulation leaving $s \cup \overline{S}$ minus the sum of lower bounds of edges entering $s \cup \overline{S}$, where $\overline{S}$ is the set of nodes v_e with $e \notin S$. For a full proof, see Appendix A. □

One can hook up the circulation gadgets above to form a graph $G(S)$, where feasible flows uniquely determine Nash equilibrium payments with active edges S, and vice versa. Using the above Lemma, it is easy to prove the following theorem.

Theorem 3.3 *An edge set S induces a Nash equilibrium with payment vector c if and only if $G(S)$ has a feasible flow with values c' on corresponding edges. In addition, every such flow corresponds exactly to payments obeying the constraints of Remark 3.1.*

Corollary 3.4 *For any active edge set S, we can polytime compute an NE payment vector c for S (or determine that none exists) that optimizes any linear objective function.*

4 Creating Good Nash Equilibria

In Section 2.1 we established that, since all Nash equilibria in LCFG may be of very poor quality, we must allow incentive schemes if we hope to form good Nash equilibria. We consider incentives in the form of payments by some central authority to players, under the constraint of some total budget B.

There are several forms of incentives an entity could offer to players, four of which are as follows:

1. For some or all players, give some amount of money to a player only if it follows a particular strategy.
2. For some or all players, for some or all edges incident to a player, give some amount of money to a player only if it does not cut this edge.
3. Increase the penalties for not delivering particular demands.
4. Increase the values of particular demands ($\lambda_s(d)$'s).

We define a *Type-i Nash equilibrium* as a NE in an instance of LCFG where we also employ incentives of form i, $i = 1, 2, 3, 4$, as described above. In this paper, we consider Type-4 Nash equilibria, since they are more general than the others because of the following theorem.

Theorem 4.1 *Let N_i be the collection of sets of edges that induce a Type-i Nash equilibrium, for $i = 1, 2, 3, 4$, and a fixed incentive budget B. Then, $N_1 = N_2 \subseteq N_4$, and $N_3 \subseteq N_4$.*

To prove this theorem, we first need to prove the useful Lemma 4.2. The proofs of both Lemma 4.2 and Theorem 4.1 are in the Appendix.

Lemma 4.2 *Suppose there is a node v with 2 best deviations, cutting S_1 and S_2. Then, cutting $S_1 \cap S_2$ is also a best deviation of v.*

4.1 A Nash Equilibrium as good as OPT

The main result of this section is that if we increase the λ-values for every demand by a factor of 2, then in the resulting game instance there is a Nash equilibrium whose active edges are exactly the active edges of OPT. For the results in this section, we assume $\lambda_s(d) = \lambda_t(d)$ for s, t being endpoints of d and we write $\lambda(d)$ to be this common value. If this were not the case, then instead of increasing $\lambda_s(d)$ by a factor of 2, the results hold if we set $\lambda_s(d)$ to $\lambda_s(d) + \lambda_t(d)$, which is still a factor of 2 increase in total.

Theorem 4.3 *Let E^* be the set of active edges in OPT. If we increase $\lambda(d)$ by a factor of 2 for every d, then E^* induces a Nash equilibrium.*

To prove this, we need to show a new sufficient condition for a given set S of active edges to induce a Nash equilibrium. We use this new condition to prove Theorem 4.3. Consider the following bipartite b-matching problem $MP(S)$ with node sets A and B where A has a node for every (s, d) pair where s is an endpoint of active demand d, and B contains a node for every active non-peering edge in S. For each $e \in B$ we define the *capacity* of e to be $x(e) = |D(e)| - |D^{end}(e)|$. The capacity $x(s, d)$ for each node $(s, d) \in A$ is defined as $\lambda_s(d) - 2$. For every $(s, d) \in A$, there is an edge in $MP(S)$ between (s, d) and all nodes in B representing non-peering edges in $P(d)$ directed away from s (i.e., that are reachable from s via a directed subpath of $P(d)$). The basic idea here is that if in an x-matching an amount y is matched between $s \in A$ and $e \in B$ then y "flows" from s to the head v of e to cover some of v's transit costs.

Lemma 4.4 *If an x-matching exists in $MP(S)$, then S induces a Nash equilibrium.*

Note that the existence of a NE on S does not always imply the existence of an x-matching in $MP(S)$. Only a subset of NEs can be described by such matchings, as they are essentially solutions where each demand pays only for transit on its own demand path (although it may pay for transit of other demands on those nodes). Using Lemma 4.4, we can now prove Theorem 4.3. Since we are dealing with the optimal centralized solution, cutting any set of edges in OPT decreases the social welfare. We can use this fact to form a matching $MP(OPT)$ as above after increasing all $\lambda(d)$ values by a factor of 2. For the full proof, see Appendix A.

Extensions of Theorem 4.3 We now show that there is no need to increase λ by a factor of 2, when we can instead increase it by an additive term. Let P_1 and P_2 be the two maximal directed paths in $P(d)$ for demand d starting at the two endpoints of d. Note that P_i might just consist of an endpoint of d. Let δ_i be such that $\|P(d)\| = \delta_i \|P_i\|$, where $\|P\|$ is the number of nodes in path P. These values δ_1 and δ_2 represent the imbalance of this path. If we let δ_d be the smaller of these, we can now present the following theorem.

Theorem 4.5 *To form a Nash equilibrium on the same edges as OPT, it is enough to increase $\lambda(d)$ to become $A\lambda(d) + (1 - \frac{A}{2})(\|P(d)\|/\delta_d)$, for any $0 \leq A \leq 2$.*

Corollary 4.6 *To form a Nash equilibrium on OPT, it is enough to increase $\lambda(d)$ to $A\lambda(d)+(1-\frac{A}{2})\|P(d)\|$, for any $0 \leq A \leq 2$. In particular, it is always enough to set $\lambda(d)$ to be $\frac{3}{2}\lambda(d) + \frac{\|P(d)\|}{4}$.*

Corollary 4.7 *Suppose $\lambda_1(d)$ is the demand value at the start of $P_1(d)$, and $\lambda_2(d)$ is the same for $P_2(d)$. Then, to form a Nash equilibrium on OPT, it is enough to set $\lambda_1(d)$ to $A[\lambda_1(d) + \lambda_2(d)] + (1 - A)\|P_1(d)\|$, and similarly for $\lambda_2(d)$, for any $A \geq 0$.*

These corollaries show that in many graphs, it is not necessary to increase every single λ by a factor of 2. The amount of money we need actually depends on the length of demand paths, as well as the imbalance between the two directed parts of the paths. Unfortunately, assuming that $\|P_1(d)\| = \|P_2(d)\|$ does not improve these bounds, even in the special case where all the paths $P(d)$ are confluent.

Relative Difference We can also generalize the result of Theorem 4.3 as follows. Define the *relative difference* of a Nash equilibrium N compared to OPT in terms of social welfare as as $rel(N) = (W(OPT) - W(N))/\Lambda$, with $\Lambda = \sum_{d \in D(OPT)} \lambda(d)$. This says that in terms of social welfare, the difference between OPT and N is small relative to the total value of the active demands in OPT.

The following theorem gives us a bound on relative difference. For the case where $\varepsilon = 1$, this is exactly Theorem 4.3, where we can form a Nash equilibrium with no relative difference.

Theorem 4.8 *If for all d we set $\lambda(d)$ to $(1 + \varepsilon)\lambda(d)$, then there is a NE N such that $rel(N) \leq (1 - \varepsilon)$.*

5 A Different Objective Function and Price of Stability

In LCFG, the social welfare can be positive or negative, and so is a mixed-sign objective. We have shown that if we consider OPT to be the solution maximizing social welfare, then the prices of anarchy and stability can both be unbounded. This often occurs with mixed-sign objectives in optimization problems, but such objectives can often be transformed into natural same-sign objective functions that give better approximation ratios (see e.g. [18]). Consider, the objective of $\sum_{d \in D(V)} \|P(d)\| - \sum_{d=st \notin D(V)} (\lambda_s(d) + \lambda_t(d))$. In this objective, we want to minimize the transit cost in the entire solution, and we also look to minimize the total λ-value of demands that are not connected. This is the objective function used in [30]. Notice that the minimum of this objective is also the solution with maximum social welfare, so the change of objective only matters for approximations.

As in the previous section, assume that $\lambda_s(d) = \lambda_t(d)$, and call this value $\lambda(d)$. Then the following holds (as before, there is a corresponding result for the case that $\lambda_s(d) \neq \lambda_t(d)$).

Theorem 5.1 *With respect to the non-mixed sign objective function above, the price of stability is at most 2.*

The proof of the above theorem (which can be found in the Appendix) does not only give us an existence result, but also an approximation algorithm that runs in polynomial time. Given any solution S, we can find a Nash equilibrium that is at most twice as expensive as S. This is done by attempting to find a matching similar to $MP(S)$ from Lemma 4.4. If we find such a matching, then we are done, and otherwise obtain a set of edges that can be taken out from the current solution without greatly increasing the cost.

6 Multicast Settings

We know from Theorem 2.2 that it is difficult to find a good, or even a non-trivial Nash equilibrium. In this section, we consider the special case of multicast on a tree. By *multicast* we mean that all demands have one endpoint in common. In the Appendix, we show that approximating the best NE in a multicast DAG is NP-hard. However, in the case where the underlying graph is a tree, we can find a best Nash equilibrium in polynomial time via a complex dynamic program, as the following theorem states.

Theorem 6.1 *In a multicast tree, we can find the best Nash equilibrium in polynomial time if all demands are unit-size.*

While all the other results in this paper extend to non-unit demands, this one does not. In fact, in the case where the size of the demand is specified as part of the input (i.e. where the input is size $\log D$ instead of D), we show in the Appendix that in a multicast tree, finding a non-trivial Nash equilibrium is NP-hard. If, however, the underlying tree is actually an arborescence (i.e. all edges are directed towards the sink), then we can find a non-trivial Nash equilibrium with nice properties. This is possible even with non-unit demands, as evidenced by the following theorem.

Theorem 6.2 *In a multicast arborescence, we can find a Nash equilibrium S maximizing $\sum_{d \in D(S)} [\lambda(d)] - t(S)$ in polynomial time.*

The proofs of our multicast results can be found in Appendix C. The proof of Theorem 6.2 is especially interesting, as it establishes the structure and some nice properties of Nash equilibria on an arborescence.

Acknowledgments

We would like to thank Martin Pál and Jon Kleinberg for productive and illuminating discussions. We are grateful to Jennifer Rexford and Francis Zane for several helpful insights.

References

[1] E. Anshelevich, A. Dasgupta, É. Tardos, T. Wexler. Near-optimal network design with selfish agents. *STOC*, 2003.

[2] E. Anshelevich, A. Dasgupta, J. Kleinberg, É. Tardos, T. Wexler, T. Roughgarden. The Price of Stability for Network Design with Fair Cost Allocation. *FOCS*, 2004.

[3] V. Bala, S. Goyal. Self-organization in communication networks. McGill University Technical Report, 1996.

[4] V. Bala, S. Goyal. A noncooperative model of network formation. *Econometrica*, 68, pp. 1181–1229, 2000.

[5] Francis Bloch, M.O. Jackson. The Formation of Networks with Transfers among Players. *J. Economic Theory*, to appear.

[6] R. Cáceres, N. Duffield, A. Feldmann, J.D. Friedmann, A. Greenberg, R. Greer, T. Johnson, C.R. Kalmanek, B. Krishnamurthy, D. Lavelle, P.P. Mishra, J. Rexford, K.K. Ramakrishnan, F.D. True, J.E. van der Merwe. Measurement and analysis of IP network usage and behaviour. *IEEE Communications Magazine*, pp. 144–151, May 2000.

[7] Jacomo Corbo, Thomas Petermann. Selfish peering and routing in the Internet. CoRR cs.GT/0410069, 2004.

[8] Jacomo Corbo, David C. Parkes. The price of selfish behavior in bilateral network formation. *PODC*, pp. 99–107, 2005.

[9] Jacomo Corbo, David C. Parkes. The Role of Bilateral Consent in Strategic Network Formation. Harvard Working Paper Series.

[10] T. Erlebach, A. Hall, A Panconesi, D. Vukadinović. Cuts and Disjoint Paths in the Valley-Free Path Model of Internet BGP Routing. *Proceedings of CAAN,* pp. 49–62, 2004.

[11] T. Erlebach, A. Hall, T. Schank. Classifying Customer-Provider Relationships in the Internet. *Proceedings in Informatics*, pp. 52–64, September 2002.

[12] A. Fabirikant, A. Luthra, E. Maneva, C.H. Papadimitriou, S. Shenkar. On a network creation game. *PODC*, 2003.

[13] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenkar. A BGP-based Mechanism for Lowest-Cost Routing. *Distributed Computing 18*, pp. 61-72, 2005. (Special issue of selected papers from Proc. of ACM PODC'02.)

[14] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, F. True. Deriving traffic demands for operational IP networks: methodology and experience. *IEE/ACM Trans. on Networking* vol. 9, No. 3, pp. 265–279, June 2001.

[15] L. Gao, J. Rexford. Stable internet routing without global coordination. *Proceedings of ACM SIGMETRICS '00*, Santa Clara, CA, June 2000.

[16] A. Gomes. Multilateral contracting with externalities. *Econometrica*, 73(4), pp. 1329–1350, 2005.

[17] G. Gopalakrishnan, B. Hajek. Do greedy autonomous systems make for a sensible internet? *Conference on Stochastic Networks*, June 19-24, 2002.

[18] M. Goemans, D. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. In *Approximation Algorithms for NP-Hard Problems,* ch. 4, pp 144-191. PWS Publishing Company, 1997.

[19] A. Greenberg, A. Shaikh. Operations and Management of IP Networks: What Researchers Should Know. *SIGCOMM Tutorial*, 2005.

[20] T. Griffin, G. Wilfong. On the Correctness of IBGP Configuration. *Proceedings of SIGCOMM,* pp. 17–29, 2002.

[21] P. Hall. On Representatives of Subsets. *Journal of London Mathematical Society*, 10, pp. 26–30, 1935.

[22] L. He, J. Walrand. Dynamic provisioning of service level agreements between interconnected networks. *Conference on Stochastic Networks*, June 19-24, 2002.

[23] H. Heller, S. Sarangi. Nash networks with heterogeneous agents. *Virginia Tech. Working Paper Series*. E-2001-1, 2001.

[24] B.E. Hermalin, M.L. Katz. Network interconnection with two-sided user benefits. *Working paper available:* `http://faculty.haas.berkeley.edu/hermalin/Interconnection_v14.pdf`

[25] `http://hosting.bellsouth.net/bellsouthhosting/s/s.dll?spage=cg/news/service_level.htm`

[26] `http://www.verio.com/access/pricing.cfm`

[27] `http://www.xo.com/products/smallgrowing/internet/dia/oc3.html`

[28] M. Jackson, A survey of models of network formation: stability and efficiency. *Group Formation in Economics: Networks, Clubs and Coalitions*, eds. G. Demange and M.Wooders, Cambridge Univ. Press. `http://www.hss.caltech.edu/ jacksonm/netsurv.pdf`

[29] R. Johari, J. Tsitsiklis. Routing and Peering in a Competitive Internet. *Proceedings of the IEEE Conference on Decision and Control*, 2004.

[30] R. Johari, S. Mannor, and J. Tsitsiklis. A contract-based model for directed network formation. Submitted (November 2003).

[31] E. Koutsoupias, C. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science,* pp. 404–413, 1999.

[32] J.J. Laffont, J.S. Marcus, P. Rey, J. Tirole. Internet connection and the off-net pricing principle, *The RAND Journal of Economics*, vol. 34, n. 2, pp. 370–390, 2003. Earlier version in *Conference on the Economics of the Internet and Software Industries* Jan. 18-20, 2001.

[33] J. Larrosa, F. Tohmé. Network formation with heterogeneous agents. Economics Working Paper Archive EconWPA, Microeconomics series, No. 0301002, 2003.

[34] B. Shepherd, G. Wilfong. Multilateral Transport Games. *Proceedings of INOC*, pp. B2-378–B2-385, 2005.

[35] S. Shakkottai, R. Srikant. Economics of network pricing with multiple ISPs. *INFOCOM*, 2005.

[36] H. Wang, H. Xie, Y.R. Yang. On the stability of interdomain inbound traffic engineering. *SIGCOMM Poster Session*, 2005.

Appendix A: Proofs and Open Questions

Discussion and Open Questions

Some immediate open questions are to devise other sufficient conditions besides the existence of a matching in $MP(S)$ for a Nash equilibrium to exist, and then find incentives to induce these conditions. Longer-term open questions follow.

Our model has several limitations, the foremost of which are the assumptions that paths $P(d)$ are fixed and that the links have no capacity. Actual network contracts usually involve a bandwidth limit, which is effectively a capacity. The paths $P(d)$ are usually determined by the BGP protocol, which each node has some control over, since it can determine its own path preferences. A more realistic (but also far more complicated) model would be one where a node's strategy combines contract formation and the path preferences it publishes. An alternate simpler model would let a node choose the paths of its demands as part of its strategy, but then even finding a best deviation becomes NP-Complete. Studying these types of models is future work.

Contract formation is a two-person operation, and for this reason most economics literature addresses the notion of pairwise stability (also known as link stability), where both sides of an edge can deviate at once to form a contract. This has some nice properties, such as the absence of trivial Nash equilibria. Pairwise stability usually involves a single edge activating or de-activating as a deviation. However, a node should be able to make (or break) several contracts at once as part of a deviation. Combining pairwise stability with these considerations might require a cooperative game theory model instead. Looking at the dynamics of our game would also be very interesting.

We study the stable solutions that arise after giving small incentives to some players. Notice that the central authority giving these incentives has very limited control over the network. It cannot change the mechanism used by the players, and cannot alter the game except via a small budget. It would be interesting to look at other such realistic limited powers (such as building a few links, or taking over a few nodes) that an altruistic entity could use to induce good stable solutions.

Finally, while our model is intended to model more than just the interaction of Autonomous Systems, contracts between them are certainly our main application. In reality, the contract graph is hierarchical, and has small depth. All our positive results still hold for this case, of course, as well as some of our negative results. Specifically, if we allow non-unit demands, then most things remain hard even with small depth. Looking at unit demands with bounded depth graphs is another good direction.

Proofs

Proposition 2.1 *For every Nash equilibrium A, there exists another Nash equilibrium B with the same active edges, and all strictly positive payments only being paid from customer to provider.*

Proof. Construct B as follows. Suppose $e = (u, v)$ but v is actually making a payment to its customer u. Change the solution so that now e remains active but the payment for it is 0. We claim this solution B is still a Nash equilibrium. All nodes except u and v have the same deviations available to them in B as they did in A and v's deviations can only result in a utility at least as bad as in A (since in B it no longer gains the payment it was making to u if it cuts e). Suppose to the contrary that u could cut a set of edges S that results in increasing its utility. Since A is a Nash equilibrium, u in A cannot gain in utility by cutting any subset of edges, and so e must be in S. Consider the set of edges $S' = S \setminus \{e\}$, and let $\Delta_A(S')$ be the utility increase of u resulting from cutting S'. Similarly, let $\Delta_B(S)$ be the utility increase of u from cutting S in B. Let $D(\overline{S'}, e)$ be the set of demands in $D(e)$ that do not include edges from S'. Then, notice that $\Delta_B(S) - \Delta_A(S') = -\sum_{d \in D^{end}(u) \cap D(e)} \lambda_u(d) + t(\overline{S'}, e) - \sum_{d \in D(\overline{S'}, e)} (\lambda_u(d) - 1)$. Since the penalties in the last term above are always greater than the transit cost, then we have that $\Delta_B(S) - \Delta_A(S') \leq 0$. We

also know that $\Delta_A(S') \leq 0$ since A is a Nash equilibrium, so $\Delta_B(S) \leq 0$, contradicting the assumption that cutting S is a desirable deviation. Thus B is a Nash equilibrium and we can repeat this process for each edge e where there is a payment other than a payment from a customer to a provider. □

Lemma 3.2 *Node v is stable with payment vector c if and only if there exists a circulation (as illustrated in Figure 4) where the flow on corresponding edges is c'.*

Proof. First we need to specify exactly what it means for node v to be stable. Let $Cust(v)$ be the set of active customer edges of v, and $Prov(v)$ be the set of active provider and peer edges of v. Node v is stable if, given current payments, it does not desire to cut (break connections with by raising its price or lowering its offer) any set of active edges $S_1 \cup S_2$, where $S_1 \in Cust(v)$ and $S_2 \in Prov(v)$. The utility of v before cutting $S_1 \cup S_2$ is

$$\sum_{e \in Cust(v)} c'(e) - \sum_{e \in Prov(v)} c'(e) - t(v). \tag{2}$$

Define $D(A, B)$ as the demands d such that one edge in d's route is in set A, and one edge is in set B. Let $t(A, B) = |D(A, B)|$. Define $p(A, B)$ as the extra penalties v would have to pay to its customers across edges in B because of cutting edges in A. The fact that cutting $S_1 \cup S_2$ does not increase v's utility is equivalent to the following inequality:

$$\sum_{e \in S_1} c'(e) - \sum_{e \in S_2} c'(e) - t(S_1 \cup S_2) + p(S_1 \cup S_2, Cust(v) - S_1) \geq 0. \tag{3}$$

The first two terms are simply the edge-by-edge utility v loses by cutting edges. v also gains $t(S_1 \cup S_2)$ utility because it does not have to pay the transit cost anymore for demands using only S_1 and S_2. The last term represents the extra penalties v would have to pay.

Now we form a circulation gadget as in Figure 4, but with the edges having both upper and lower bounds of $c'(e)$. If a feasible flow existed for the gadget with no bounds on these edges, such that the flow out of s was c', then this flow would still be feasible in the new gadget. Conversely, if no such feasible flow existed before, there would still be none.

This new gadget has a feasible circulation exactly when for each set of nodes S in the graph, the sum of upper bounds of edges leaving S minus the sum of lower bounds of edges entering S is nonnegative. Let $\delta(S)$ denote this quantity. In the above construction, if S does not include the node s, then $\delta(S) \geq 0$ trivially, since there is an edge from every node v_e to s with infinite capacity.

Now consider the cuts of this circulation graph containing s. We can make each of these cuts correspond exactly to a possible strategy by our node as follows. We can think of each cut in the circulation graph as $S = \{s\} \cup (Cust(v) - S_1) \cup (Prov(v) - S_2)$, where $Cust(v) - S_1$ is the set of nodes corresponding to customer edges included on the s-side of the cut, and $Prov(v) - S_2$ is the similar set of provider/peer edges. In other words, the nodes v_e on the s-side of the cut correspond to edges that our node decides not to cut, while it cuts the rest.

We now show that $\delta(\{s\} \cup (Cust(v) - S_1) \cup (Prov(v) - S_2))$ is exactly the amount of money v would lose by cutting $S_1 \cup S_2$. For every $e \in S_1$, δ increases by $c'(e)$ because of the edges (s, v_e), and for every $e \in S_2$, δ decreases by $c'(e)$ because of the edges (v_e, s). For every demand d going through a customer edge $e \in S_1$ and provider/peer edge e', δ decreases by 1 if d is active because of the edges (v_e, s), contributing a total of $t(S_1, Prov(v))$. If $e \notin S_1$ and $e' \in S_2$, δ increases by $\lambda_u(d) - 2$ for active d and by $\lambda_u(d) - 1$ for inactive penalty-enabled d, because of the edges $(v_e, v_{e'})$. These edges contribute a total of $p(S_2, Cust(v) - S_1) - t(Cust(v) - S_1, S_2)$ to δ. So far, we have accounted for δ increasing by

$$\sum_{e \in S_1} [c'(e)] - \sum_{e \in S_2} [c'(e)] - t(S_1, Prov(v)) + p(S_2, Cust(v) - S_1) - t(Cust(v) - S_1, S_2).$$

Now consider a demand d going through 2 customer edges e and e' of v. If d is active and both e and e' are in S_1, δ decreases by 1 because of the edge (v_e, s), for a total of $t(S_1, S_1)$. If $e \notin S_1$ but $e' \in S_1$, then a simple analysis of the circulation graph tells us that because of edges $(v_e, s), (v_{e'}, s), (v_e, v_{e'})$, and $(v_{e'}, v_e)$, we have that δ increases by $\lambda_u(d) - 2$ if d is active, and by $\lambda_u(d) - 1$ if d is inactive but penalty-enabled. This contributes to the increase of δ by

$$-t(S_1, S_1) + p(S_1, Cust(v) - S_1) - t(S_1, Cust(v) - S_1).$$

We have now considered the contributions of all edges to δ, so in total

$$\delta(S) = \sum_{e \in S_1} [c'(e)] - \sum_{e \in S_2} [c'(e)] - t(S_1) - t(S_2, Cust(v) - S_1) + p(S_1 \cup S_2, Cust(v) - S_1)$$

which are exactly the terms in Inequality 3. Therefore, $\delta(S)$ is nonnegative exactly when the corresponding strategy does not improve the node's utility, and so a feasible flow exists exactly when our node is stable. □

Theorem 3.3 *An edge set S induces a Nash equilibrium with payment vector c if and only if $G(S)$ has a feasible flow with values c' on corresponding edges. In addition, every such flow corresponds exactly to payments obeying the constraints of Remark 3.1.*

Proof. One can hook up the circulation gadgets to form a graph $G(S)$, where feasible flows uniquely determine Nash equilibrium payments with active edges S, and vice versa. Instead of a node s for each gadget, we have a single node s, where all the λ values flow from, and all the transit payments flow to. Otherwise, for every edge $e = (v, w)$ in G, we connect the gadgets as expected, joining v_e to w_e. The only tricky part is that $c'_v(e)$ should not equal $c'_w(e)$. We form extra nodes v'_e and w'_e, so that instead of joining s, the provider edge of v's gadget joins v'_e, and the customer edge of w's gadget starts at w'_e. There is an edge (v'_e, w'_e) with flow value of $c(e)$, and edges from and to s from v'_e and w'_e with capacities and lower bounds exactly enough to convert $c'_v(e)$ into $c(e)$ and then into $c'_w(e)$. The rest of the proof is clear from Lemma 3.2. □

Corollary 3.4 *For any active edge set S, we can polytime compute an NE payment vector c for S (or determine that none exists) that optimizes any linear objective function.*

Proof. Form a graph $G(S)$ as above, where S is all the edges in the routes of the desired demands. Now find a feasible flow if one exists, which only requires a single flow computation. The values of this flow provide us with values $c(e)$ in a Nash equilibrium, since we can convert $c'(e)$ to $c(e)$. □

Lemma 4.2 *Suppose there is a node v with 2 best deviations, cutting E_1 and E_2. Then, cutting $E_1 \cap E_2$ is also a best deviation of v.*

Proof. Let S be some solution to an instance of LCFG. As usual we assume that all inactive edges have infinite prices so a deviation is just a set of edges a node cuts. Suppose there is a node v in the current solution which has several best deviations, each increasing its utility by x. Then, we prove that cutting the intersection of these deviations also increases the utility of v by x (so it is also a best deviation).

It is enough to prove that if cutting a set of edges E_1 and E_2 are two best deviations, then cutting $E_1 \cap E_2$ is also a best deviation. Let $f(S)$ be the utility gained by v from cutting S. As shown in the proof of Lemma 3.2, $f(S)$ is exactly the negative of the quantity in Inequality 3. If we let $E_1 = C_1 \cup S_1$, and $E_2 = C_2 \cup S_2$, where C_1, C_2 are sets of customer edges and S_1, S_2 are sets of provider/peer edges, then $f(E_1 \cup E_2)$ is:

$$\sum_{e \in S_1 \cup S_2} c'(e) - \sum_{e \in C_1 \cup C_2} c'(e) + t(E_1 \cup E_2) - p(E_1 \cup E_2, Cust(v) - (C_1 \cup C_2)). \quad (4)$$

We now prove that $f(E_1 \cup E_2) \geq f(E_1) + f(E_2) - f(E_1 \cap E_2)$. Since $\sum_{e \in S_1 \cup S_2} c'(e) = \sum_{e \in S_1} c'(e) + \sum_{e \in S_2} c'(e) - \sum_{e \in S_1 \cap S_2} c'(e)$ and $t(E_1 \cup E_2) = t(E_1) + t(E_2) - t(E_1 \cap E_2) - t(E_1 - E_2, E_2 - E_1)$, then the only hard part is the penalty term. Notice, however, that

$$p(E_1 \cup E_2, Cust(v) - (C_1 \cup C_2)) = \\ p(E_1, Cust(v) - (C_1 \cup C_2)) + p(E_2, Cust(v) - (C_1 \cup C_2)) - p(E_1 \cap E_2, Cust(v) - (C_1 \cup C_2)),$$

Looking at each term, we have that

$$p(E_1, Cust(v) - (C_1 \cup C_2)) = p(E_1, Cust(v) - C_1) - p(E_1, C_2 - C_1),$$

and similarly for E_2, as well as

$$p(E_1 \cap E_2, Cust(v) - (C_1 \cup C_2)) = p(E_1 \cap E_2, Cust(v) - (C_1 \cap C_2)) - p(E_1 \cap E_2, C_1 \cup C_2 - C_1 \cap C_2).$$

Therefore, $f(E_1 \cup E_2) = f(E_1) + f(E_2) - f(E_1 \cap E_2) + \alpha - \beta$, where α is the "leftover" penalty:

$$\alpha = p(E_1, C_2 - C_1) + p(E_2, C_1 - C_2) - p(E_1 \cap E_2, C_1 \cup C_2 - C_1 \cap C_2) = \\ p(E_1, C_2 - C_1) + p(E_2, C_1 - C_2) - p(E_1 \cap E_2, C_1 - C_2) - p(E_1 \cap E_2, C_2 - C_1) = \\ p(E_1 - E_2, C_2 - C_1) + p(E_2 - E_1, C_1 - C_2),$$

and β is the "leftover" transit:

$$\beta = t(E_1 - E_2, E_2 - E_1).$$

For each demand d contributing to the β term, there is a penalty appearing in the α term, since one of the two edges of P_d incident to our node must be a customer edge. Since the penalty is always greater than the transit cost of d (which we assumed to be 1), is must be that $\alpha \geq \beta$. Therefore, $f(E_1 \cup E_2) \geq f(E_1) + f(E_2) - f(E_1 \cap E_2)$ (and f is supermodular). Since E_1 is a best deviation, $E_1 \cup E_2$ cannot be a better one, so $f(E_1 \cup E_2) \leq f(E_1)$, and $f(E_1 \cap E_2) \geq f(E_2)$, as desired. □

Theorem 4.1 *Let N_i be the collection of sets of edges that induce a Type-i Nash equilibrium, for $i = 1, 2, 3, 4$, and a fixed incentive budget B. Then, $N_1 = N_2 \subseteq N_4$, and $N_3 \subseteq N_4$.*

Proof. In the following, we assume that no payments in any Nash are paid from a provider to a customer, or from a peer to a peer, since we already showed that for each such Nash, there exists an equivalent one without this problem.

First we show that $N_1 \subseteq N_2$. Take a Nash equilibrium in N_1, with x the amount of utility being paid to node v to stay in the current strategy (we can assume x is minimal). According to Lemma 4.2, there must be a minimum set S of active edges incident to v, that v could cut and obtain x more utility. Choose any edge e in S, and instead of paying v x amount not to change its strategy, pay it $x - 1$ to not change the strategy and 1 to not cut edge e. If we can show that v still does not wish to deviate, then we know that $N_1 \subseteq N_2$, since we can use this process to convert a Nash equilibrium with power 1 to a Nash equilibrium with power 2. v now desiring to deviate means it can cut a set of edges S' and increase its utility by at least x, to offset the $x - 1$ utility it would lose by deviating. If $e \notin S'$, then S' was a best deviation for v before, and by Lemma 4.2, it must be that $S \subseteq S'$. Since $e \in S$, this is impossible. Therefore, it must be that $e \in S'$. Since we added 1 to the payment on e, this means that by cutting S' before this, v could have gained greater than x utility. This contradicts the fact that x was the largest utility gain that v could make before.

It is clear that $N_2 \subseteq N_4$, since instead of paying a dollar to v to not cut edge e, we could increase the $\lambda(d)$ of some active demand path going through e by 1, and alter the Nash equilibrium to propagate the payment of this 1 unit all the way to v. All nodes are still stable, since v now has one extra incentive not to

cut e because of the penalty increase, and while the nodes on the demand path of d could cut their provider edge and obtain 1 extra utility than before, they would also have to pay one extra penalty.

It is also trivial to see that $N_2 \subseteq N_1$, and $N_3 \subseteq N_4$. Unfortunately, it is not the case that $N_4 \subseteq N_2$. By decreasing $\lambda_s(d)$ by 1, and then paying 1 to s, we can make sure that s is still stable. However, the penalty for cutting this path also decreases, and there are examples where another node will then want to deviate. □

Lemma 4.4 *If an x-matching exists in $MP(S)$, then S induces a Nash equilibrium.*

Proof. By Theorem 3.3, it is enough to construct a Nash flow. We do this by defining what happens to the money $\lambda_s(d)$ in this flow. This money is used to pay for the transit cost of d at s, and also at the adjacent node after the first customer-provider edge of $P(d)$, if one exists. This leaves at least $\lambda_s(d) - 2$ left over. Beyond this, if in the x-matching this money is used to pay for some portion of $x(e)$, then in the flow it is used to pay for the same amount of transit cost at the head of e for demands in $D(e)$. This defines a flow of payments, since we just send flow on edges of $P(d)$ (keeping track of where each unit of flow originated), so that the above payments are possible. Notice that this flow pays for all the transit cost, since $x(e)$ together with the payments close to the endpoints of the demand paths covers all of it.

Additionally, for every penalty paid by v to u across edge e, increase the payment from u to v on e by the size of the penalty. Notice that this flow of money has a very nice structure, since the money $\lambda_s(d)$ is only being used to pay for the transit on its demand path $P(d)$ (although it might be paying for the transit of other demands at those nodes).

This results in a Nash flow. Consider a node v. $c'(e)$ enters v on each customer edge, all the transit is paid for by the customers, and the rest of the money goes out on provider edges. The only things we need to make sure of is that the bounded lending constraint is satisfied. In the flow we described, there is no borrowing happening between customer edges. Since the demand pays for itself on the first edge of the path, at most $(\lambda_s(d) - 2)$ money proceeds from a customer to a provider edge for each active d, as desired. Therefore, all the constraints in Remark 3.1 are met. □

Theorem 4.3 *Let E^* be the set of active edges in OPT. If we increase $\lambda(d)$ by a factor of 2 for every d, then E^* induces a Nash equilibrium.*

Proof. For any $S \subseteq E^*$ let $D(S)$ be the set of demands that are active in OPT and include an edge of S in their paths. Recall that $\|P(d)\|$ denotes the number of nodes in the demand path $P(d)$. For any $S \subset E^*$, the social welfare of the solution in which only the edges in $E^* \setminus S$ are active is

$$W(E^* \setminus S) \quad = \quad W(E^*) - \sum_{d \in D(S)} (2\lambda(d) - \|P(d)\|). \tag{5}$$

Thus the fact that $W(E^*)$ is optimal implies that

$$\sum_{d \in D(S)} \|P(d)\| \leq \sum_{d \in D(S)} 2\lambda(d). \tag{6}$$

We now form $MP(E^*)$ as defined in Section 4 above, except now the capacity of a node (s, d) is $x(s, d) = 2\lambda_s(d) - 2$. This is because our old values for $\lambda(d)$ have been doubled. For any set of non-peer edges $S \subseteq E^*$, we have that

$$\sum_{e \in S} x(e) \leq \sum_{d \in D(S)} (\|P(d)\| - 2) \leq \sum_{d \in D(S)} (2\lambda(d) - 2). \tag{7}$$

The first part of the inequality follows from the fact that a demand d "contributes" 1 to the value of $x(e)$ for at most $\|P(d)\| - 2$ edges e (the exact amount depending on whether $P(d)$ has a peering edge and on the

number of nonempty maximal directed paths it contains). The second part of the inequality follows directly from Inequality 6.

Notice that $MP(E^*)$ contains an edge between each $e \in S$ and one of the two endpoints of $P(d)$ for $d \in D(e)$. This means that Inequality 7 is exactly the condition of Hall's Theorem [21] guaranteeing the existence of an x-matching. Therefore, by Lemma 4.4, there exists a Nash equilibrium with active edges E^*. □

Theorem 4.5 *To form a Nash equilibrium on the same edges as OPT, it is enough to increase $\lambda(d)$ to become $A\lambda(d) + (1 - \frac{A}{2})(\|P(d)\|/\delta_d)$, for any $0 \leq A \leq 2$.*

Proof. Let E^* be the set of active edges in OPT. By Lemma 4.4, it is enough to show that a matching for $MP(E^*)$ can be formed using the updated λ-values. Let $D_1(S)$ be the demands d such that $P_1(d)$ passes through set S, and define $D_2(S)$ similarly. Then, for $x(e)$ defined as in Lemma 4.4, we have that

$$\sum_{e \in S} x(e) \leq \sum_{d \in D_1(S)} (\|P_1(d)\| - 2) + \sum_{d \in D_2(S)} (\|P_2(d)\| - 2). \tag{8}$$

Suppose we now set new $\lambda'(d)$ to be $A\lambda(d)+(1-A/2)(\|P(d)\|/\delta_d)$. To show that the appropriate matching exists, it is thus enough to prove that

$$\sum_{d \in D_1(S)} \|P_1(d)\| + \sum_{d \in D_2(S)} \|P_2(d)\| \leq \sum_{d \in D_1(S)} \lambda'(d) + \sum_{d \in D_2(S)} \lambda'(d). \tag{9}$$

¿From the optimality Inequality 6, since $A \geq 0$, we know that

$$\frac{A}{2} \sum_{d \in D(S)} \|P(d)\| \leq \sum_{d \in D(S)} A\lambda(d).$$

This means the right side of Inequality 9 is at least

$$\frac{A}{2} \sum_{d \in D(S)} \|P(d)\| + \sum_{d \in D_1(S)} (1 - \frac{A}{2}) \frac{\|P(d)\|}{\delta_d} + \sum_{d \in D_2(S)} (1 - \frac{A}{2}) \frac{\|P(d)\|}{\delta_d} \geq$$

$$\frac{A}{2} \sum_{d \in D(S)} \|P(d)\| + \sum_{d \in D_1(S)} (\|P_1(d)\| - \frac{A}{2}\|P_1(d)\|) + \sum_{d \in D_2(S)} (\|P_2(d)\| - \frac{A}{2}\|P_2(d)\|) \geq$$

$$\sum_{d \in D_1(S)} \|P_1(d)\| + \sum_{d \in D_2(S)} \|P_2(d)\|,$$

as desired. □

Theorem 4.8 *If for all d we set $\lambda(d)$ to $(1+\varepsilon)\lambda(d)$, then there is a Nash equilibrium N such that $rel(N) \leq (1 - \varepsilon)$.*

Proof. Consider the set of edges E^* active in OPT. If there is no set $S \subseteq E^*$ such that

$$\sum_{d \in D(S)} \|P(d)\| > \sum_{d \in D(S)} (1 + \varepsilon)\lambda(d),$$

then using the proof of Theorem 4.3 we can form a Nash equilibrium N on exactly the active edges of OPT, and therefore have $W(N) = W(OPT)$. Therefore, assume that there is such a set. Form a maximal set S' with this property by adding to it greedily singleton sets of edges while making sure the above property holds. After we remove this set from E^*, the resulting set S_N has no more subsets of edges with the above

property. This is because if there were such a set, there would be a singleton set with this property as well, so we could have added it to S' without violating the property above. Therefore, the set S_N induces a Nash equilibrium using the proof of Theorem 4.3. Then $W(OPT) - W(S_N)$ is bounded by

$$\begin{aligned} W(OPT) - W(S_N) &\leq \sum_{d \in D(S')} 2\lambda(d) - \sum_{d \in D(S')} \|P(d)\| \\ &\leq \sum_{d \in D(S')} \lambda(d)(1-\varepsilon). \end{aligned}$$

This finishes the proof. □

Theorem 5.1 *With respect to the non-mixed sign objective function above, the price of stability is at most 2.*

Proof. Let $P_1(d)$ and $P_2(d)$ be as in Theorem 4.5. Let $D_1(e)$ be the set of active demands d with $P_1(d)$ including e, and similarly for $D_2(e)$. Denote the endpoints of d by s_1, s_2, with $s_i \in P_i(d)$.

To find a Nash equilibrium with cost at most $2 * OPT$, do as follows. At each step, we have a solution S. Start with S being the optimal solution, and attempt to find a b-matching $MP(S)$ as defined in Lemma 4.4. If we find such a matching $MP(S)$, this means we have a Nash equilibrium induced by S. Otherwise, we obtain a set of edges $S' \subseteq S$ such that

$$\sum_{d \in D_1(S')} x(s_1, d) + \sum_{d \in D_2(S')} x(s_2, d) < \sum_{e \in S'} x(e). \tag{10}$$

This set S' is the set of edges such that the λ values of the appropriate demands are not large enough to pay for the transit in S'. We would like to show that

$$\sum_{d \in D_1(S')} \lambda_d + \sum_{d \in D_2(S')} \lambda_d \leq \sum_{d \in D(S')} \|P(d)\|. \tag{11}$$

Suppose we knew this was true. Then we could form a new solution $S - S'$, and repeat the above process (i.e., attempt to find a matching $MP(S - S')$). This algorithm must either find such a matching eventually, giving us a Nash equilibrium, or remove all the edges, which results in a trivial Nash equilibrium.

Moreover, at every step of the above algorithm our solution S has an objective value of at most twice the optimum. Suppose the value of OPT is $\Lambda + T$, where Λ is the total λ value of demands that are disconnected in OPT, and T is the total transit cost of all active demands in OPT. Every time we remove some set of edges S' in the above algorithm, we decrease the transit cost by some value $t = \sum_{d \in D(S')} \|P(d)\|$, and increase the λ value of inactive demands by at most $2t$, since Inequality 11 implies that $\sum_{d \in D(S')} \lambda_d \leq \sum_{d \in D(S')} \|P(d)\|$. Since we never add edges, the value of our final Nash equilibrium is at most $\Lambda + 2T$, giving us a price of stability of at most 2.

All that is left is to prove Inequality 11. To do this, let $\chi(P(d))$ be the contributions of $P(d)$ to the values $x(e)$ (the amount that $x(e)$'s increase in total because of $P(d)$).

Let $d \in D_1(S')$, but $d \notin D_2(S')$. By the definition of $x(e)$ in $MP(S)$, we know that $\chi(P_1(d)) \leq \|P_1(d)\| - 2$ times. This is because $x(e)$ may increase for every edge of $P_1(d)$ except the first one. Since $\|P_1(d)\|$ is the number of nodes in $P_1(d)$, we have a difference of 2 above. If all demand were of this type, we would now be done, since $x(s, d) = \lambda_d - 2$, and so Inequality 10 would give us the desired result.

Now consider $d \in D_1(S'), D_2(S')$. We would like to say that the demand path $P(d)$ contributes to the $x(e)$ values at most $\|P(d)\| - 4$ times. Unfortunately, this is not true, for example in the case that $\|P(d)\| < 4$. We must therefore alter the definition of $x(e)$, forming a new matching $MP'(S)$.

Define $MP'(S)$ exactly the same as $MP(S)$, with the following changes. Take an active demand path $P(d)$ with endpoints s, t that contains a node v with 2 customer edges (call them f_1, f_2). If neither f_1 nor

f_2 are incident to s or t, then take $x(f_1)$ and reduce it by 1 (we pick one of f_1 and f_2 arbitrarily). If exactly one of f_1 or f_2 is incident to s or t, reduce the x value of the one that is not by 1. Otherwise, the entire path $P(d)$ consists of s, v, and t, and $P(d)$ makes no contributions to the x value of any edges. In this case, set $x(s, d)$ to be $\lambda_d - 1$ instead of $\lambda_d - 2$.

It is easy to check that Lemma 4.4 still holds for this new matching $MP'(S)$. Notice also that in all of the above cases, if $x(s_1, d) + x(s_2, d) = 2\lambda_d - k$ for some $k = 3, 4$, then $\chi(P(d)) \leq \|P(d)\| - k$. Since $\sum_{e \in S'} x(e) \leq \sum_{d \in D(S')} \chi(P(d))$, then by Inequality 10, we know that $\sum_{d \in D_1(S')} \lambda_d + \sum_{d \in D_2(S')} \lambda_d \leq \sum_{d \in D(S')} \|P(d)\|$, as desired. □

Appendix B: Hardness Results

Finding a Non-Trivial Nash is Hard, Even in a Tree

Theorem 2.2 *Finding a non-trivial Nash equilibrium is NP-complete even when G is an arborescence. Moreover, there is no polynomial-factor approximation algorithm for finding a best Nash equilibrium.*

Proof. Consider the problem NON-TRIVIAL NASH described as follows. We are given a network $H = (U, A)$, a collection of demands D and we wish to know if there is a subset $S \subseteq A$ of edges that induce a non-trivial Nash equilibrium.

Then, NON-TRIVIAL NASH is NP-complete even when G is an arborescence (and even if all demands are of size 1).

We show a reduction from STABLE SET. An instance of STABLE SET is where we are given a graph $G = (V, E)$, an integer $k \geq 2$ and we wish to know if it has a stable set $W \subset V$ such that $|W| \geq k$.

Given an instance $J = (G = (V, E), k)$ of STABLE SET, we now describe how to construct an instance $I = (H = (U, A), D)$ of NON-TRIVIAL NASH where H is an arborescence and J has a solution if and only if I does.

The arborescence H is rooted at a node r. For each node $v \in V$ there is a path P_v of length $k + 3$ directed from a node labeled v to r. There is also a path Q of length $k + 1$ directed into r from a node z. These paths are pairwise node disjoint except for the common end point r. There is a node t and edge (t, r). Finally, for each $v \in V$ there is an edge (v', z) (see Figure 5).

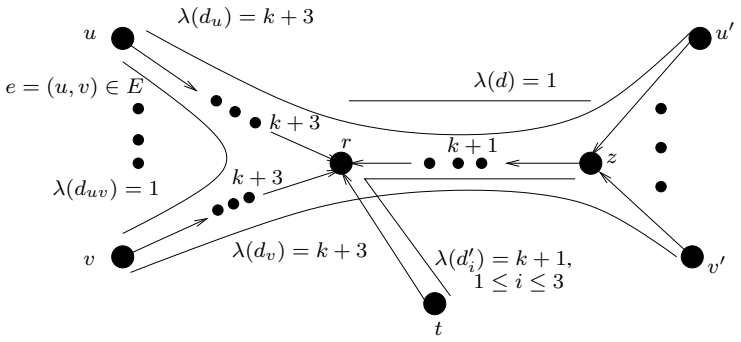

Figure 5: Illustrating construction in the proof of Theorem 2.2.

There is a demand d between z and r and hence $P(d) = Q$. Define $\lambda(d) = 1$ and so d will need help to pay for transit at k nodes between its endpoints. For each node $v \in V$ there is a *node demand* d_v from v to v' and so $P(d_v) = v, \ldots r, \ldots z, v'$. Define $\lambda(d_v) = k + 3$. Thus each node demand d_v has the ability to pay for its transit from v to the node just before r on P_v. It also has exactly enough to pay for its transit from v' to r or as we shall take advantage of, it has enough to pay for its transit from v' to the node just before

r on Q and then have 1 left over to help pay for the transit of d. For each edge $(u, v) \in E$ there is an *edge demand* d_e between u and v and so $P(d_e)$ consists of P_u and P_v. Let $\lambda(d_e) = 1$. Then edge demands never have enough to pay for their transit and there are no demands that can afford to help them. Finally, there are 3 *dummy demands* d'_1, d'_2, d'_3 between t and z each with $\lambda(d'_i) = k + 1$. These demands have sufficient money at z to pay for their transit from z to the node just before r on Q and then from t they each have $k-1$ extra after paying their transit at t and r.

Suppose we have a non-trivial Nash equilibrium N. Then clearly there can be no active edge demands in N. If the dummy demands or some node demand is active, then d is active and hence $m \geq k$ node demands must be active to pay for d's transit along Q. Therefore we know that demand d and at least k node demands are active in N. If the dummy demands are not active, then r only gets payment of $m - k$ for the m node demands that it transits and so its utility would be negative. Therefore the dummy demands must be active. Thus the active demands in N consist of at least k node demands, all dummy demands and d. Since the dummy demands pay for the transit at r for the k active node demands that help d with its transit costs, the social welfare of N is $3(k-1)-k = 2k-3 > 0$. The amount $3(k-1)-k$ comes from $3(k+1)$ for payouts to t minus 6 for transiting the dummy demands at r and t and minus k for the k node demands whose transit the dummy demands pay for at r. Thus if there is a non-trivial Nash equilibrium, then it has strictly positive social welfare and has at least k active node demands and no active edge demands. Therefore if V' is the subset of nodes of G such that d_v is active in N, then clearly V' is a stable set of size at least k.

Now suppose there is a stable set V' such that $|V'| \geq k$. For each $v \in V'$ we activate the edges along each $P(d_v)$. Also, activate edge (t, r). Then d is active, all the dummy demands are active and for each $v \in V'$, the node demand d_v is active. Then it's straightforward to check that we have a Nash equilibrium with social welfare $3(k-1) - k = 2k - 3 > 0$.

In the above construction, any non-trivial Nash equilibrium has strictly positive social welfare. Therefore, the above proof shows that it is NP-Complete to distinguish between an instance where the social welfare of the best NE is 0 or non-zero. To form a α-approximation algorithm, however, we must be able to produce a Nash equilibrium with social welfare at least $\frac{1}{\alpha}W$ for any instance where the best NE has social welfare W. Since for any $W > 0$, $\frac{1}{\alpha}W$ is also positive, the above proof shows this is NP-Complete. □

Theorem 6.3 *Determining if Nash equilibrium exists with social welfare at least k is NP-Complete even in a multicast tree, if the demands are not required to be unit size.*

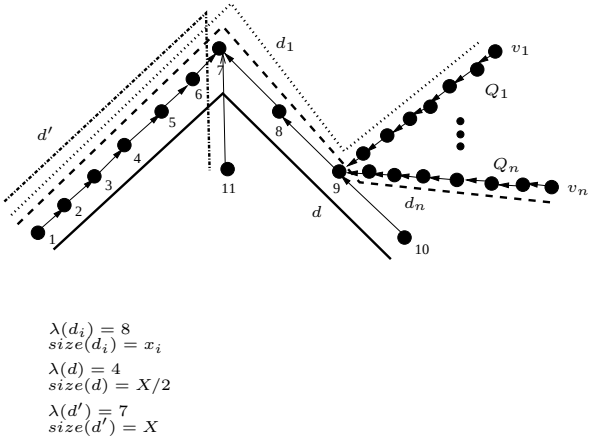

Figure 6: Construction used in 6.3.

Proof. The proof is via a reduction from PARTITION. An instance of PARTITION consists of a collection of positive integers $x_1, x_2, \ldots, x_n$ where $\sum_{i=1}^{n} = X$ and we wish to know if there is a subcollection S such

that $\sum_{x_i \in S} x_i = X/2$. From such an instance we create an instance of LCFG that has a Nash equilibrium with social welfare at least $4X$ if and only if there is a solution to the PARTITION instance. The tree network consists of a path P from a leaf node labeled 1 to the root node labeled 7 (through nodes 2, 3, 4, 5, and 6), a path R from a node labeled 10 to the root node 7 through nodes 9 and 8, and paths Q_i from leaf v_i to node 9 where $\|Q_i\| = 8$, $1 \leq i \leq n$. There is a demand d between 1 and 10 where $\lambda(d) = 4$ and $size(d) = X/2$. There is a demand d' between nodes 1 and 11 where $\lambda(d') = 7$ and $size(d') = X$. Also, for $1 \leq i \leq n$, there is a demand d_i between v_i and 1 where $\lambda(d_i) = 8$ and $size(d_i) = x_i$. (See Figure 6.)

Suppose there is a solution S to the PARTITION instance. Then consider the solution to the game where the active demands are d, d' and d_i for each $x_i \in S$. On the right hand side of the picture, each d_i can just cover its transit costs along Q_i but not enough to cover the transit at node 8 or node 7. However, d can pay for its transit along R at nodes 10, 9 and 8 with exactly $X/2$ to spare to pay for the active d_i's transit at 8. Thus all but the transit at nodes along P are covered. On the left side along P, the active d_i demands can pay for their transit along P from node 1 through node 6 with exactly X left over to pay $X/2$ to each of 5 and 6 for d's transit. Demand d' can pay for all the demands at 7 and so this is a Nash equilibrium. It can be checked that the social welfare of this Nash equilibrium is $4X$.

Suppose there is a Nash Equilibrium N with social welfare at least $4X$. It's easy to check that any solution where d' is not active, has social welfare of at most 0. Thus d' must be active in N. Let S be the set of active d_i demands in N. Notice that each d_i can pay for its transit through at most 8 nodes and so by themselves, no subset of these demands could form a Nash equilibrium. Thus demand d must be active in N. But d only has $X/2$ to spare for paying for transit at node 8 for the the d_i's in S and so $\sum_{d_i \in S} x_i \leq X/2$. However demand d can only pay for its transit on P at nodes 1, 2, 3, and 4 requires help to pay for all its transit at nodes 5 and 6, i.e., it needs a total of X extra to pay for this transit. In total, the demands in S have $8a - 6a = 2a$ extra after paying for their own traffic along P from node 1 to node 6, where $a = \sum_{d_i \in S} x_i$. Thus $a = \sum_{d_i \in S} x_i \geq X/2$ if there is to be enough extra to pay for d's transit on 5 and 6. Hence $\sum_{d_i \in S} = X/2$. The only transit that remains unaccounted for is the X amount of transit at 7 for demands d and the demands of S. But d can pay for them with a payment from node 11 to 7. The social welfare of N is then clearly $4X$ as required. □

Inapproximability of the NASH and Centralized Optimization Problems

Theorem 6.4 *A best (highest social welfare) Nash equilibrium, as well as OPT are inapproximable up to n^{ε} even in star networks.*

Proof. Here we show that finding a best Nash equilibrium and OPT is inapproximable up to n^{ε} even in star networks.

Consider an n-node instance of the STABLE SET problem $G = (V, E)$. We create an instance of the Local Contract Formation Game as follows. We consider a graph H with a subset of nodes $V \cup \{r\} \cup \{v' : v \in V\}$. For each $v \in V$, we add a path P_v between r and v of length $n + 2$ and a path $P_{v'}$ between v' and r of length $n + 4$. The P_v's and $P_{v'}$'s are also chosen to be internally disjoint and the edges are directed towards r. This is the graph for our instance of LCFG (see Figure 6.4). For each v, we also assume the existence of a demand d_v between v and v' and hence on the path $P(d_v)$ consisting of the paths P_v and $P_{v'}$. We set $\lambda(d_v) = n + 4$. Finally, for any $e = (u, v) \in E$, we add a demand d_e between u and v whose path is $P(d_e)$ consisting of paths P_u and P_v. These demands have $\lambda(d_e) = 1$. For any demand d, let $E(d)$ be the edges in $P(d)$.

Then for every v such that the demand d_v is active there is a gain of 1 to the social welfare (since the transit cost of d_v is $2n + 7$ which is 1 less than $2(n + 4) = 2n + 8$). Notice that if any edge demand d_e is active, this results in a loss of $2n + 5 - 2 = 2n + 3 > n$ and hence is greater than the social welfare due to all n d_v's.

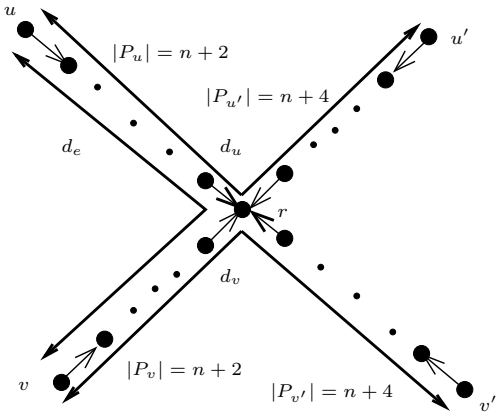

Figure 7: Illustration of construction in the proof of Theorem 6.4.

If S is a stable set in G, then let $E_S = \cup_{v \in S} E(d_v)$. Then the total social welfare of E_S is exactly $|S|$ since no $E(d_e)$ is contained in E_S for any $e \in E$. Thus OPT is at least $\alpha(G)$, the maximum stable set size of G. Note that the amount OPT can also be achieved by the Nash equilibrium where each endpoint of an active demand d pays $\lambda(d)$ to its neighbor and then any other node on an edge in E_S pays 1 less to its provider than it is paid by its customer. Note that there are penalty payments activating only "half" of a d_e demand but that payment is $\lambda(d_e) - 1 = 0$. Thus a best Nash equilibrium has social welfare of at least $\alpha(G)$ as well.

Conversely, suppose that E' is some subset of the edges of H and let $S \subset V$ be those nodes such that E' contains the edges of P_v and $P_{v'}$. If S is not a stable set, then there is some $P(d_e)$ contained in E', and hence the social welfare of E' is negative. Since the social welfare of this subgraph is at most $|S|$, we have that $\alpha(G)$ is at least the social welfare of E'. Combining the two inequalities we have that $\alpha(G)$ is precisely the maximum social welfare of a subgraph of H. □

Theorem 6.5 *OPT and best Nash equilibrium are inapproximable up to n^{ε} even in acyclic single-sink networks where all demands terminate at the sink r, and all demands are unit-size.*

Proof. Again start with an instance $G = (V, E)$ of STABLE SET and create a network H created from arc disjoint paths $P_v : v \in V$ as follows. For each edge $e = uv \in E(G)$, we define special nodes $a_{e,1}, a_{e,2}, a_{e,3}$. For each $i = 1, 2$, there are two parallel arcs $e_{i,u}, e_{i,v}$ with tail $a_{e,i}$ and head $a_{e,i+1}$. Then for each node v in G and for each edge e having v as an endpoint, P_v is defined so that $e_{1,v}e_{2,v}$ is a subpath of P_v. After P_v has passed through each of these edges, it then passes through sufficient auxiliary nodes until it reaches the root node r so that its total length is as determined below. There is a demand d_v such that $P(d_v) = P_v$. This proof only holds if $\lambda_r(d) = 0$ for all d, so we let $\lambda(d_v)$ denote the demand value of the other endpoint. (If we want the demand values to be the same for both endpoints of a path, this proof can be used to show that finding the best Nash equilibrium or OPT is NP-Hard, but we lose the inapproximability result.) We set $\lambda(d_v) = \|P_V\| + 1$ so that, as above, satisfying d_v results in a net utility of 1. In addition, for each e, we add a path Q_e that follows $e_{1,u}$ and then $e_{2,v}$ and then follows P_v, say, on to the root (see Figure 8). We set demand on these paths to have huge negative utility (which can be done by padding the lengths of P_v's as needed).

Note next that if a solution ever contains P_u and P_v, then the negative utility demand on Q_e gets routed. Thus any solution must only use P_v's from a stable set and so $W(OPT) \leq \alpha(G)$. Conversely, if S is a stable set in G, then one can easily check that we can form a Nash equilibrium on $\cup_{v \in S} E(P_v)$ without any negative utility demand, and hence with social welfare $|S|$. Thus $W(OPT) \geq W(NE) \geq \alpha(G)$, where

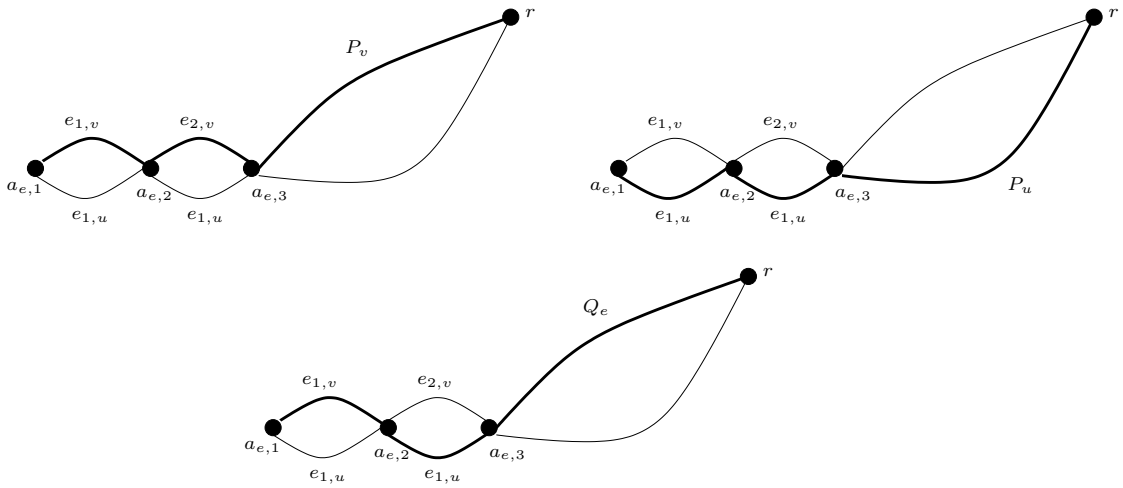

Figure 8: Construction in the proof of Theorem 6.5.

$W(NE)$ is the social welfare of the best Nash equilibrium, and $\alpha(G)$ is the size of the maximal stable set. □

Appendix C: Multicast Settings

Finding a Nash Equilibrium on an Arborescence

Suppose that the graph of potential contracts is an arborescence directed towards the sink node r, *and* all demands terminate at r. We now show how to find in polynomial time a Nash equilibrium N maximizing

$$f(S) = \sum_{d \in D(S)} [\lambda(d)] - t(S),$$

where S is the set of active edges in N. The proof below is easily extensible to demands of arbitrary sizes. The proof implies that we can find if there exists a non-trivial Nash equilibrium on a tree in polynomial time. If the demand values $\lambda_r(d)$ for each demand d to the sink node r are all 0 (the only players who benefit from connections are the customers), then the Nash equilibrium we find actually maximizes social welfare.

We call a Nash equilibrium *sensible*, if its active edges form a connected component containing r. Notice that there are no penalties paid in a sensible Nash equilibrium on a multicast arborescence.

Lemma 6.6 *If N is a Nash equilibrium with active edges S in a multicast arborescence then there is a sensible Nash equilibrium with active edges $S' \subseteq S$ in which all nodes have the same utility as they do in N and $f(S) = f(S')$.*

Proof. Let N be a Nash equilibrium and suppose there is some connected component K of the active edges of N that does not include r. Then the total utility of the nodes in K is 0 since there can be no active demands originating in K and all penalty payments cancel out. There cannot be any nodes in K with positive utility since that would imply that there must be at least one with negative utility and this would contradict that N is a Nash equilibrium. Thus cutting all the edges in K would not change any of the node utilities. Also, the set of active demands remain the same so $f(S) = f(S')$. □

For each node v in a tree T rooted at r let T_v denote the subtree rooted at v. Let $\lambda(T_v) = \sum_{d \in D(T_v)} \lambda(d)$ be the total demand value to nodes in T_v, and $t(T_v)$ be the total transit cost to nodes in T_v. A subtree T_v

is called *self-sufficient* if $\lambda(T_v) - \sum_{d \in D(T_v)} \|P(d)\| \geq 0$. We call an arborescence *super-sufficient* if each of its subtrees is self-sufficient. We in fact show that a Nash equilibrium maximizing f can be obtained by cutting some edges and leaving a super-sufficient tree. This is possible because of the following structural properties of Nash equilibria.

Lemma 6.7 *T induces a sensible Nash equilibrium in a multicast arborescence exactly when for every subtree T_v of T, we have that $\lambda(T_v) \geq t(T_v)$.*

Proof. Consider the structure of Nash equilibria in a multicast arborescence as prescribed by Theorem 3.3. There are no penalty-enabled inactive demands for any node, and therefore no penalty payments. The bounded lending constraint is always satisfied, since there is no borrowing between customer edges, and the bounds on the amount passed from a customer edge to a provider edge are always satisfied. Therefore, a set of active edges T induces a Nash equilibrium exactly when there exists a flow of payments in T where $\lambda_s(d)$ appears at s for every active d, payments follow directed edges, and all transit cost is paid for. Clearly, if $\lambda(T_v) < t(T_v)$ for some T_v, then this is impossible. Conversely, assume that $\lambda(T_v) \geq t(T_v)$ for all v. Then we can build the desired flow of payments recursively from the leaves of T, by simply sending up the provider edges all of the flow that is not used to pay for transit. □

For a tree T and a subtree T_v we use the notation $T - T_v$ to mean the tree resulting from removing T_v and v's provider edge from T.

Lemma 6.8 *If T induces a Nash equilibrium N, and T_v is a maximal subtree which is not self-sufficient, then $T - T_v$ induces a Nash equilibrium with greater f objective value than that of N.*

Proof. Since T_v is not self-sufficient, then $f(T - T_v) > f(T)$. Thus we just need to show that $T - T_v$ induces a Nash equilibrium. Let w be the first (lowest) node in $T - T_v$ such that $\lambda(T_w - T_v) < t(T_w - T_v)$. If w is not an ancestor of v, then $T_w - T_v = T_w$, and so by Lemma 6.7 T could not have been a Nash equilibrium. If w is an ancestor of v, however, we know that T_w is not self-sufficient since both T_v and $T_w - T_v$ are not self-sufficient. This contradicts the maximality of T_v, and so such a node w cannot exist. Therefore, by Lemma 6.7, $T - T_v$ induces a Nash equilibrium. □

Lemma 6.9 *If $T - T_v$ induces a Nash equilibrium N, and T_v is super-sufficient, then T induces a Nash equilibrium with at least as large f objective value as N.*

Proof. Consider any subtree T_w of T. If $T_w \subseteq T_v$, then T_w is self-sufficient, because T_v is super-sufficient. Otherwise, w must be an ancestor of v, so $T_w = T_v \cup (T_w - T_v)$. Since N is a Nash equilibrium, by Lemma 6.7 we know that $\lambda(T_w - T_v) \geq t(T_w - T_v)$. The total transit cost in T of all demands in T_v is at most $\sum_{d \in D(T_v)} \|P(d)\|$, and so after the addition of T_v the total transit cost of T_w increases by at most $\sum_{d \in D(T_v)} \|P(d)\|$. Since T_v is super-sufficient, we now have that $\lambda(T_w) = \lambda(T_w - T_v) + \lambda(T_v) \geq t(T_w - T_v) + \sum_{d \in D(T_v)} \|P(d)\| \geq t(T_w)$. Therefore, by Lemma 6.7, T induces a Nash equilibrium, and $f(T) - f(T - T_v) \geq \lambda(T_v) - \sum_{d \in D(T_v)} \|P(d)\| \geq 0$, since T_v is super-sufficient. □

Theorem 6.2 *In a multicast arborescence, we can find a Nash equilibrium maximizing f in polynomial time. Moreover, there exists such an optimal Nash equilibrium which is super-sufficient.*

Proof. Because of Lemma 6.8, for every Nash equilibrium induced by T, there exists a super-sufficient Nash equilibrium induced by $T' \subseteq T$ such that $f(T') \geq f(T)$. Therefore, we can restrict our search to super-sufficient Nash equilibria.

Our algorithm to find the Nash equilibrium maximizing f proceeds as follows. If the entire arborescence T is super-sufficient then it induces a Nash equilibrium and clearly it is optimal since all demands are

active in it and by super-sufficiency all demands contribute a non-negative term to the objective function. Otherwise, consider a minimal subtree T_v that is not self-sufficient (clearly we can find one if one exists in polynomial time). We claim that by deleting T_v, we do not destroy any sensible optimal super-sufficient Nash equilibrium unless $r = v$ (and in this latter case the trivial Nash equilibrium is optimal). For otherwise, there is a subtree T' that contains v and r and induces an optimal super-sufficient Nash equilibrium. Without loss of generality we may choose T' to be a maximal, sensible such Nash equilibrium. Now T_v cannot be contained in T', since T_v is not self-sufficient. Therefore, there exists an arc $(x, y) \in T_v - T'$; let this be the highest such arc. Since T' is sensible, we have that no arc of T_x is in T'. By choice of T_v, we have that T_x is super-sufficient. But then, by Lemma 6.9, we can add T_x to T' to obtain another sensible, super-sufficient, optimal Nash equilibrium, contradicting maximality. Thus indeed we may work on the smaller instance $T - T_v$ and still be assured that an optimal super-sufficient Nash equilibrium is contained inside. The final tree we are left with is super-sufficient, and hence induces a Nash equilibrium by Lemma 6.7. □

Multicast on a Tree

In this section we address the case where all demands have a node r in common (which we call the sink), and the underlying graph is a tree. This is different from the previous section, since we can now have edges directed away from the sink. The only reason why the results here are not more general is because, unlike Theorem 6.2, they cannot be extended to demands of non-unit size.

First, we consider the case in the previous section, but with unit-size demands, and show that we can find the best Nash equilibrium (with the objective $W(S)$ used in the rest of this paper).

Theorem 6.10 *In a multicast arborescence where all edges are directed towards the sink, we can find the best Nash equilibrium in polynomial time if all demands are unit-size.*

Proof. Root the tree at r. Every node v has 1 edge e in the direction of r, and edges $e_1, e_2, \ldots, e_{l_v}$ away from r. Let T_v^i be the subtree of the given arborescence, containing v, that occurs if we cut the edges $e, e_{i+1}, e_{i+2}, \ldots, e_{l_v}$. By Lemma 6.7, a tree T containing r induces a Nash equilibrium exactly when $\lambda(T') \geq t(T')$ for all subtrees $T' \subseteq T$ containing v that arise from cutting the edge above $v \in T$. Below when we refer to a subtree of a tree T, we mean only this type of subtree. In the following proof we assume that the demand values at both endpoints of a demand d are the same, so we denote the demand value at an endpoint by $\lambda(d)$. This proof easily extends to the case when they are different.

We now form a dynamic program to find the edges S with largest objective $W(S)$, such that the condition in Lemma 6.7 holds. For any tree not containing r, let $g(T) = 2\lambda(T) - \sum_{d \in D(T)} \|P(d)\|$, and for a tree containing r, let $g(T) = \lambda(T) - \sum_{d \in D(T)} \|P(d)\|$. Notice that the objective function value contributed by a tree T to $W(S)$ is actually $g(T)$, since adding T increases the objective by this much (the demand values at r contribute as well). We call this the g-value of a tree. Since for a tree T containing r, we have that $g(T) = W(T)$, our goal now is to find a tree containing r with the largest g-value.

Let D be the number of demands, n the number of nodes, and Δ the maximum degree. Denote a tree T as (v, i, x, y)-optimal if it has the highest g-value out of all subtrees of T_v^i containing v that meet the following constraints:

1. $x = \min\{\lambda(T) - t(T), nD\}$ and $y = |D(T)|$.
2. For every subtree T' of T, we have that $\lambda(T') \geq t(T')$.

It is clear that the optimal Nash equilibrium is an (r, l_r, x, y)-optimal tree for some x, y. There are only $\Delta n^2 D^2$ such subtrees, so if we could find them, we are done.

An important property of (v, i, x, y)-optimal trees is that every subtree of such a tree is also a (v', i', x', y')-optimal tree. To see this, let T be a (v, i, x, y)-optimal tree, and let T_1 be a subtree of T rooted at u, including edges up to j edges below u. Let $x_1 = \min\{\lambda(T_1) - t(T_1), nD\}$ and $y_1 = |D(T_1)|$. Now, let T_1^* be a

(u, j, x_1, y_1)-optimal tree, and replace T_1 with T_1^*. Since the function g is additive over subtrees, this only increases the g-value of T. All that is left to show is that T still satisfies Property 2 above. Let T' be a subtree of the new tree T, rooted at v'. If v' is below u, then $\lambda(T') \geq t(T')$, since T_1^* satisfies Property 2 above. If v' is an ancestor of u, let k be the number of nodes on the path from u to v'. If $x_1 = nD$, then $\lambda(T') \geq nD$, and so $\lambda(T') \geq t(T')$ since the total transit in the network cannot be more than nD. Therefore, we can assume that $x_1 = \lambda(T_1) - t(T_1)$. The contribution of T_1 to $\lambda(T') - t(T')$ before replacement was $\lambda(T_1) - t(T_1) - ky_1$, since the latter was the total transit caused in T' by demands from T_1. This means that to show that after replacement $\lambda(T') \geq t(T')$, it is enough to show that $\lambda(T_1^*) - t(T_1^*) - ky_1 \geq \lambda(T_1) - t(T_1) - ky_1$. This is true by definition of ()-optimal tree, since they both have the same x_1. Therefore, we can replace T_1 with T_1^*, still satisfy the constraints, and only increase the g-value. This proves that any subtree of a ()-optimal tree is also a ()-optimal tree.

We can now form a dynamic program to find all (v, i, x, y)-optimal trees. For leaf nodes v, it is easy to initialize these entries. Now, to find a general (v, i, x, y)-optimal tree, where $e_i = (u, v)$, and there are a demands with endpoint v of total λ value b, we proceed as follows. We look at the unions of $(v, i-1, x_1, y_1)$- and (u, l_u, x_2, y_2)-optimal trees such that $x = \min\{x_1 + x_2 + b - y_1 - y_2 - a, nD\}$, and $y = y_1 + y_2 + a$, and we take the one with the highest g-value. By the above argument, any (v, i, x, y)-optimal tree must have such a form, and there are n^2D^4 possibilities, so by induction, this finds all (v, i, x, y)-optimal trees in polynomial time.

The above dynamic program works because the objective function is additive over subtrees. In fact, this proof works for any such objective. □

Theorem 6.1 *In a multicast tree, we can find the best Nash equilibrium in polynomial time if all demands are unit-size.*

Proof. This proof parallels the proof of Theorem 6.10. First we need to establish a version of Lemma 6.7 for trees. A multicast tree rooted at r is a collection of arborescences directed towards r, ending at an arborescence directed away from r. We need to show stability conditions for the part directed away from r, as Lemma 6.7 still applies for the rest. We can transform every NE with penalties being paid, into an equivalent one with the penalties being immediately "repaid" back up the edge, so they simply cancel, and we can assume there are no penalties being paid in the Nash equilibria we consider. Using the terminology of Remark 3.1, there can also be customer edges borrowing from other customer edges, such as in the case of a node v with customer edges e_1, e_2 and provider edge f, where e_1 is the edge between v and r. Since this is a multicast setting, there are no demands from e_2 to f, so it only makes sense for e_1 to borrow from e_2. Since e_2 cannot use this money to pay for any more transit of its demands, we might as well assume that e_1 borrows all of the money that is left over after e_2 pays for transit at v. All that is left is the necessity that all transit be paid for, and the bounded lending constraint that no more than $\lambda(d) - 2$ for every d is contributed by a customer edge to a provider edge. This exactly characterizes Nash equilibria in this setting.

We call T an (v, i, x, y, z)-optimal subtree of T_v^i if it has the highest g-value out of all subtrees of T_v^i containing v that meet the following constraints:

1. $x = \min\{\lambda(T), nD\}$, and $y = |D(T)|$.
2. If z utility enters v from the root direction, then all the transit can be paid for in T without violating the above constraints (in other words, T can be made stable).

As before, in the case where x or z is nD, we can pay for all the transit, so we only consider the case when they are less. The best Nash equilibrium is a $(r, l_r, x, y, 0)$-optimal tree, so if we can find all such trees, we would be done.

We can show, as in the previous proof, that every subtree of a (v, i, x, y, z)-optimal tree is also a (v', i', x', y', z')-optimal tree. To see this, let T be a (v, i, x, y, z)-optimal tree, and let T_1 be a subtree

of T rooted at u, including edges up to j edges below u. Let $x_1 = \lambda(T_1)$, $y_1 = |D(T_1)|$, and let z_1 be the amount of payments entering T_1 from the direction of r. As discussed above, we can assume that x_1 and z_1 are smaller than nD. Now, let T_1^* be a (u, j, x_1, y_1, z_1)-optimal tree, and replace T_1 with T_1^*. Since the function g is additive over subtrees, this only increases the g-value of T. All that is left to show is that T still satisfies Property 2 above. Make the payments outside T_1 be exactly as before, which we can do without changing stability because the demand benefit appearing at r is the same for both T_1 and T_1^* (because they have the same x_1), and the transit caused by them is the same as well (because the have the same y_1). Then, the money entering T_1^* can be z_1 after the replacement, and so T can be made stable once again.

We can now form a dynamic program to find all (v, i, x, y, z)-optimal trees. For each of the arborescences directed towards r that are hanging off of the tree, we use the dynamic program in Theorem 6.10, but with an extra parameter: the total sum of λ's of the subtree (floored with nD). Now, to find a general (v, i, x, y, z)-optimal tree, where $e_i = (u, v)$, and there are a demands with endpoint v of total λ value b, we proceed as follows. We look at the unions of $(v, i-1, x_1, y_1, z_1)$- and (u, l_u, x_2, y_2, z_2)-optimal trees such that $x = \min\{x_1 + x_2 + b, nD\}$, $y = y_1 + y_2 + a$, and z is calculated appropriately as well, and we take the one with the highest g-value. Specifically, z is the amount needed to get z_1 and z_2 to enter their respective trees, but so that all the new transit is paid for at v and so that the above constraint is satisfied. Notice that z_2 can be negative if the corresponding tree is an arborescence directed towards r. By the above argument, any (v, i, x, y, z)-optimal tree must have such a form, and there are $n^4 D^6$ possibilities, so by induction, this finds all (v, i, x, y, z)-optimal trees in polynomial time. □