Price of Stability in Survivable Network Design

Elliot Anshelevich and Bugra Caskurlu

Computer Science Department, RPI, 110 8th Street, Troy, NY 12180 {eanshel,caskub}@cs.rpi.edu

Abstract. We study the survivable version of the game theoretic network formation model known as the Connection Game, originally introduced in [5]. In this model, players attempt to connect to a common source node in a network by purchasing edges, and sharing their costs with other players. We introduce the *survivable* version of this game, where each player desires 2 edge-disjoint connections between her pair of nodes instead of just a single connecting path, and analyze the quality of exact and approximate Nash equilibria. This version is significantly different from the original Connection Game and have more complications than the existing literature on arbitrary cost-sharing games since we consider the formation of networks that involve many cycles.

For the special case where each node represents a player, we show that Nash equilibria are guaranteed to exist and price of stability is 1, i.e., there always exists a stable solution that is as good as the centralized optimum. For the general version of the Survivable Connection Game, we show that there always exists a 2-approximate Nash equilibrium that is as good as the centralized optimum. To obtain the result, we use an approximation algorithm technique that compares the strategy of each player with only a carefully selected subset of her strategy space as well as proving new results about the laminar structure of survivable networks, which may be of independent interest in classical settings. Furthermore, if a player is only allowed to deviate by changing the payments on one of her connection paths at a time, instead of both of them at once, we prove that the price of stability is 1. We also discuss the time complexity issues.

1 Introduction

Network design is a fundamental problem for which it is important to understand the effects of strategic behavior. To accomplish this, algorithmic game theory has become a major tool for studying networks such as the Internet, which are developed, built, and maintained by a large number of independent agents, all of whom act in their own interests.

In such networks, the global performance of the system may not be as good as in the case where a central authority can simply dictate a solution; rather, we need to understand the quality of solutions that are consistent with self-interested behavior. Much research in the theoretical computer science community has focused on this performance gap and specifically on the notions of the *price of anarchy* and the *price of stability* — the ratios between the costs of the worst and best Nash equilibrium[1], respectively, and that of the globally optimal solution. Both of these notions are important since as the "stable" points in a game, the Nash equilibria are often the only viable outcomes of agent interactions. We will only consider pure (i.e., deterministic) Nash equilibria, as mixed strategies do not make as much sense in our context.

Connection Game In this paper, we consider the price of stability of several important extensions of the *Connection Game*, which was first defined in [5], and later studied in a variety of papers including [3, 9, 11, 15, 18, 19]. This game represents a general framework where a network is being built by many different agents/players who have different connectivity requirements, but

[1] Recall that a (pure-strategy) Nash equilibrium is a solution where no single player can switch her strategy and become better off, given that the other players keep their strategies fixed.

can combine their money to pay for some part of the network. The Connection Game models not only communication networks, but also many kinds of transportation networks that are built and maintained by competing interests. Specifically, each player in this game has some connectivity requirements in a graph $G = (V, E)$, i.e, she desires to connect a particular pair of nodes in this graph. With this as their goal, players can offer payments indicating how much they will contribute towards the purchase of each edge in G. If the players' payments for a particular edge e sum to at least the cost of e, then the edge is considered *bought*, which means that e is added to our network and can now be used to satisfy the connectivity requirements of any player.

Survivable Network Design One of the most important extensions of the Steiner Forest problem is Survivable Network Design (sometimes called the Generalized Steiner Forest problem), and for good reasons. In this problem, we must not simply connect all the desired pairs of terminals, but instead connect them using r edge-disjoint paths. This is generally needed so that in the case of a few edge failures, all the desired terminals still remain connected. Many nice results have been shown for finding the cheapest survivable network, including Jain's 2-approximation algorithm [22].

In this paper, we consider the *Survivable Connection Game,* where each agent/player wishes to connect to her destination using $r = 2$ edge-disjoint paths. The optimal (i.e., cheapest) centralized solution for this game is the optimal solution to Survivable Network Design, which we denote by OPT. Our goals are to understand the quality of exact and approximate Nash equilibria by comparing them to OPT, and thereby understand the efficiency gap that results because of the agents' self-interest. By studying the price of stability, we also seek to reduce this gap, as the best Nash equilibrium can be thought of as the best outcome possible if we were able to suggest a solution to all the players simultaneously.

Our Results We only consider the case where all players are attempting to connect to a single common source. For the single source version of the Connection Game, [5] proved that the price of stability is 1, and that in particular, a pure Nash equilibrium always exists. This is no longer true if the players have arbitrary connection requirements, as a pure Nash equilibrium is no longer guaranteed to exist. Figure 1(A) shows a game where this is the case. In Figure 1(A), the node everyone wishes to connect to is labeled s, and the edge costs are as shown. There is a player for each node v_i that wishes to connect v_i to s using a single path, and there is a player that wishes to connect u to s using 2 disjoint paths. It is easy to see that if the bottom path is entirely bought in a Nash equilibrium, then the v_i-players do not contribute to any edges, since they have 2 disjoint paths connecting them to s, when they only desire a single one, and they would be able to remove a payment to any edge without disconnecting themselves. The u-player would never pay more than 2 in total, however, so the bottom path cannot be fully bought in any equilibrium solution. This means that in any equilibrium solution, the u-player is using the top two paths (and is paying for them entirely, since no one else has a reason to help given that they only desire a single path), and is not paying anything for the bottom path (since the u-player is not using it). All but a single edge of the bottom path must be bought in any solution, however (to ensure the connectivity for the v_i-players), so the u-player has an incentive to switch from paying 1 for one of the top paths to buying the last edge of the bottom path for $1 - \varepsilon$. Similar examples show that a Nash equilibrium is not guaranteed to exist even for 2-player games, and that there may not exist an α-approximate[2] Nash equilibrium for any $\alpha < 2$.

The example in Figure 1(A) shows that adding stronger connectivity requirements to the Connection Game significantly changes it, since the prices of anarchy and stability become infinite. Instead of considering arbitrary connection requirements, therefore, we restrict our attention to the case where all terminals desire to connect using 2 disjoint paths. In this case, we prove results that

[2] An α-approximate Nash equilibrium is a solution where no player can save more than a factor of α by deviating.

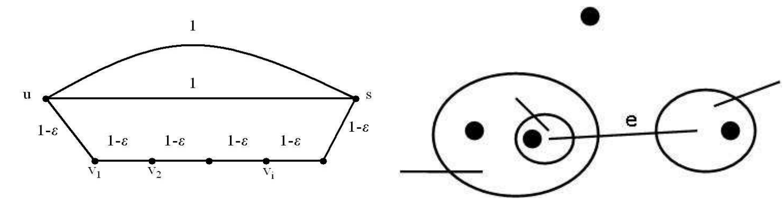

Fig. 1. (A) An instance with no pure Nash equilibrium. (B) Shows various witness sets of an edge e.

are similar to the properties of the original Connection Game. Specifically, our main results are as follows:

– In the special case where all nodes are terminal nodes (i.e., there exists a player that desires to connect this node to the source), there always exists a Nash equilibrium that is as good as OPT.
– For the general Survivable Connection Game, there exists a 2-approximate Nash equilibrium that is as good as OPT.
– For the Survivable Connection Game, there is a polynomial time algorithm which finds a cheap $(2 + \epsilon)$-approximate Nash equilibrium.

Our approach for forming equilibrium payments is similar to [5], in that we show that either every edge can be fully bought using our equilibrium payment scheme, or that OPT is not actually an optimal solution. The fact that OPT is no longer a tree, however, significantly complicates matters, forcing the payment scheme to be a bit more clever and requiring different proof techniques. In order to prove the results:

– We consider a version of the Survivable Connection Game where each player is only allowed to deviate by changing the payments on a single path and show that for that version, there always exists a stable solution that is as good as the centralized optimal solution, i.e, the price of stability is 1. We obtain our grand result by proving that any stable solution of this version corresponds to a 2-approximate Nash equilibrium of the general Survivable Connection Game.
– We prove strong results about the laminar structure of survivable networks, which are outlined in Section 2.1. These results are of independent interest, and give useful techniques for dealing with survivable networks in both game theoretic and more traditional contexts.

Related Work Over the last few years, there have been several new papers about the Connection Game, e.g., [3, 9, 11, 15, 18, 19]. Recently, Hoefer [17] proved some interesting results for a generalization of the game in [5], and showed an interesting relationship between the Connection Game and Facility Location. While the survivable network design games that we consider can be expressed as part of the framework in [17], these results do not imply ours, and our results cannot be obtained using their techniques.

The research on non-cooperative network design and formation games is too much to survey here, see [21, 24, 28, 30, 31] and the references therein. Fabrikant et al. [12] (see also [1]) studied the price of anarchy of a very different network design game, and [6] considered the price of stability of a network design game with local interactions, intended to model the contracts made by Autonomous Systems in the Internet.

A major part of the research on network games has focused on congestion games [4, 8, 10, 27, 30]. Probably the most relevant such model to our research is presented in [4] (and further addressed in [8, 9, 15]). In [4], extra restrictions of "fair sharing" are added to the Connection Game, making it a congestion game and thereby guaranteeing some nice properties, like the existence of Nash equilibria and a bounded price of stability. While the Connection Game is not a congestion game, and is not guaranteed to have a Nash equilibrium, it actually behaves much better than [4] when all the agents are trying to connect to a single common node. Specifically, the price of stability in that case is 1, while the best known bound for the model in [4] is $\frac{\log n}{\log \log n}$ [2]. Moreover, all such models (including cost-sharing models described below) restrict the interactions of the agents to improve the quality of the outcomes, by forcing them to share the costs of edges in a particular way. This does not address the contexts when we are not allowed to place such restrictions on the agents, as would be the case when the agents are building the network together without some overseeing authority. However, as [5] has shown for the Connection Game and we show for the Survivable version of it, it is still possible to nudge the agents into an extremely good outcome without restricting their behavior in any way.

The questions we consider bear some similarity to cost-sharing mechanisms for network design, such as [14, 16, 23, 25]. Unlike non-cooperative games that we consider, cost-sharing games assume that there is a central authority that designs and maintains the network, and decides appropriate cost-shares for each agent, depending on the graph and all other agents, via a complex algorithm. The agents' only role is to report their utility for being included in the network. In contrast, in our game the agents contribute to individual edges directly, rather than contributing money to a central authority. Therefore, each player is free to choose her own paths, instead of having the central authority specify paths for each player. In our game there is no central authority designing the Steiner tree or cost shares. Rather, we study Nash equilibria of our game. In essence, cost-sharing games are appropriate when the network is being designed by a central authority, and the Connection Game is appropriate when the players can choose their own paths and edge contributions.

2 The Model

We now formally define the Survivable Connection Game for N players. Let an undirected graph $G = (V, E)$ be given, with each edge e having a nonnegative cost $c(e)$, and let $s \in V$ be a special *root* (or source) node. Each player i has a single terminal node (also called *player node*) that she must connect to s using 2 edge-disjoint paths. The terminals of different players do not have to be distinct.

A strategy of a player is a payment function p_i, where $p_i(e)$ is how much player i is offering to contribute to the cost of edge e. Observe that players can share the cost of the edges. An edge e is considered *bought* if $\sum_i p_i(e) \geq c(e)$. Let G_p denote the subgraph of bought edges corresponding to the strategy vector $p = (p_1, \ldots, p_N)$. Since every player wants to have 2 edge-disjoint connections between her and s, G_p lies in the feasible region of player i if the minimum cut of G_p between i's terminal node and s is at least 2. While required to connect her terminals using at least 2 connections, each player also tries to minimize her total payments, $\sum_{e \in E} p_i(e)$.

2.1 Properties of the Socially Optimal Network

The rest of this paper is mainly devoted to proving that there exist equilibrium solutions that cost as much as OPT. To prove our results we will provide an algorithm that either forms a stable payment for OPT, or provides us with a feasible solution cheaper than OPT. For our argument to work, we make use of structural properties of OPT outlined below.

OPT, by definition, is the cheapest possible network that satisfies the connection requirements of all the players. Therefore, for every edge e, there is a set of players whose connection requirement will be dissatisfied if e is deleted from OPT, since otherwise a feasible network cheaper than OPT can be obtained by simply deleting e. Note that in a stable solution, only this set of players can make payments on e, since all other players will deviate by setting their payment on e to 0 if they had a strictly positive contribution to e. The players in this set are therefore said to *witness* e, since without them, e would not be needed.

Let v be a player witnessing e, i.e., v will have only 1 path to s if e is deleted. Observe that the size of the min-cut between v and s in OPT is 2 and it becomes 1 when e is deleted. Therefore, there is a cut in OPT between v and s of size 2 with e as one of the cut-edges. We call such a set of nodes a *witness set* of an edge. The 2 cut edges are called the boundary edges of the witness set since one side of them is in the set and the other side is outside the set. Note that since every edge in OPT has necessarily a witnessing player, it has a witness set as well, which can be constructed by the cut argument above. Figure 1(B) shows various witness sets of an edge e. The black circles represent the player nodes.

Definition 1. *A* witness set *of an edge e is a set of nodes including at least one player node and excluding s, with exactly 2 boundary edges, one of which is e.*

Observe that any player in a witness set of e witnesses e, and any player witnessing e has to be involved in some witness set of e. Intuitively, a player inside a witness set must use both of the edges leaving it, since the witness set is a cut of size 2 and she needs 2 disjoint paths. A player witnessing $e = (i, j)$ may be using e either in the direction $i \to j$ or in the direction $j \to i$. If it is using e in the direction $i \to j$ then it is inside a witness set containing i and it is inside a witness set containing j otherwise. Among the sets witnessing e in the direction $i \to j$, the smallest one in terms of the number of nodes included is called a *smallest witness set* of e in the direction $i \to j$ and we denote it as $W_i(i, j)$. $W_j(i, j)$ is also defined similarly. Smallest witness sets of the edges of OPT have very nice structural properties that we exploit in the rest of our paper. Specifically, we rely on the following theorem.

Theorem 1. *Let W be the set of all smallest witness sets of OPT, i.e., $W = \{W_i(e)| \text{ for some } i, e\}$. Then, there exists an equivalent graph (i.e., with the same price of stability) where W is laminar.*

Proof. Although we only prove this theorem below for the case when each player wants to connect to s using $r = 2$ disjoint paths, the proof can be generalized to arbitrary r. To see that the set of smallest witness sets is not necessarily laminar, consider a cycle s, t_1, v, t_2 with t_1 and t_2 being terminals and v being a non-terminal. In this case, the smallest witness set of (t_1, v) from v's side is $\{v, t_2\}$, and the smallest witness set of (v, t_2) from v's side is $\{v, t_1\}$. These two sets have a non-trivial intersection, and so are not laminar. However, if we contract the edges (t_1, v) and (v, t_2) by removing node v, then the witness set system of the new graph becomes laminar. Below we will prove that this is essentially the only way that smallest witness sets may have a non-trivial intersection, and contracting all degree 2 non-player nodes makes the set of smallest witness sets become laminar.

Let S and T be two smallest witness sets of the edges of OPT. Since laminarity does not hold in general, all $S - T, T - S$ and $T \cap S$ can be nonempty, however as we will demonstrate below, if any two smallest witness sets intersect, then their intersection will be in one of two forms which enables us to generalize our results.

Let p_1, p_2, p_3, p_4, p_5 and p_6 denote the number of edges between these 3 sets and the exterior as depicted in Figure 2.

Observe that;

$$p_1 + p_2 + p_5 + p_6 = 2 \tag{1}$$

and

$$p_3 + p_4 + p_5 + p_6 = 2 \tag{2}$$

since S and T are witness sets and therefore will have exactly 2 boundary edges.

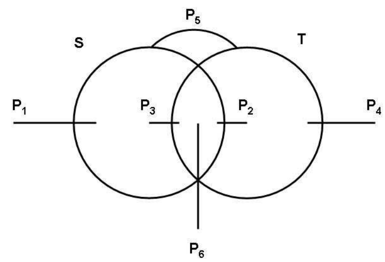

Fig. 2. General topology of two intersecting sets

Also;

$$p_1 + p_4 + p_6 \geq 2 \tag{3}$$

since there has to be at least 2 edges to leave $S \cup T$ for feasibility of the players in S and T. Observe that equations 1 and 3 together imply

$$p_4 \geq p_2 + p_5 \tag{4}$$

where equations 2 and 3 together imply

$$p_1 \geq p_3 + p_5. \tag{5}$$

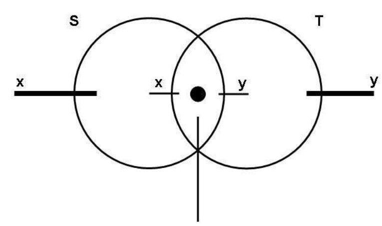

Fig. 3. Two intersecting witness sets that have a terminal in the intersection.

Assume there exists a terminal in $S \cap T$. Then; $p_2 + p_3 + p_6 \geq 2$ due to feasibility of this terminal. Observe that this equation together with equations 1 and 2 would imply $p_3 \geq p_1 + p_5$ and $p_2 \geq p_4 + p_5$ which would imply $p_1 = p_3, p_2 = p_4$ and $p_5 = 0$ when combined with equations 4 and 5. Therefore, the topology of these witness sets would be as described by Figure 3. In Figure 3 it is shown that $p_1 = p_3$ and they are both equal to x, and similarly $p_2 = p_4$ and they are both equal to some number y. Observe that S cannot be the smallest witness set of any of the edges leaving $S \cap T$ since $S \cap T$ is witnessing the same edges from the same side and has fewer nodes. For exactly the same reason, T cannot be the smallest witness set of any of the edges leaving $S \cap T$. Therefore, S is witnessing one of the x edges between $S - T$ and exterior while T is witnessing one of the y edges between $T - S$ and the exterior as depicted in Figure 3 (where the edges that S and T could be witnessing are shown in bold). Note that since $x + y + p_6 = 2$ (a smallest witness set will have exactly 2 boundary edges) and both x and y are greater than or equal to 1, $x = y = 1$.

Consider now the case where both $S - T$ and $T - S$ have at least one terminal. From feasibility of these terminals, we obtain $p_1 + p_3 + p_5 \geq 2$ and $p_2 + p_4 + p_5 \geq 2$ which in turn leads to $p_1 \geq p_4 + p_6$ and $p_4 \geq p_1 + p_6$ when combined with equations 2 and 1 respectively. When the last inequalities are combined together we obtain $p_6 = 0$ and $p_1 = p_4$ which implies $p_2 = p_3$ when we combine equations 1 and 2. Observe that S cannot be the smallest witness set of any of the $p_1 + p_5$ edges between

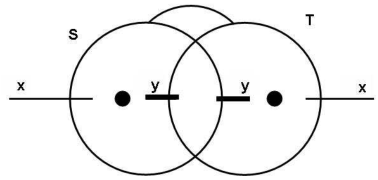

Fig. 4. Two intersecting witness sets that have a terminal in each difference.

$S - T$ and the exterior or $T - S$ because if S witnesses these edges, so does $S - T$ and it is smaller. Therefore, S witnesses one of the edges between $S \cap T$ and $T - S$. For exactly the same reason, T can only be the smallest witness set of one of the edges between $S - T$ and $S \cap T$. In Figure 4, the only edges that may be witnessed by S and T are drawn in bold. The key observation here is that there cannot be a terminal in $S \cap T$. To see this, assume the contrary. Then in Figure 4, $y \geq 1$ due to feasibility of this terminal. Since we know that $x \geq y$ from equation 4 or 5, and S has exactly 2 boundary edges, both x and y would be 1. However, observe that in this case S and T could not be the smallest witness sets of any of the edges since all possible edges would be witnessed by $S \cap T$ then as well. Therefore, $S \cap T$ does not have any terminal nodes. Observe that y has to be at least 1 since otherwise $S \cap T$ would not have any boundary edges and therefore would be empty. Since we know that $x \geq y$ from equation 4 or 5, and S has exactly 2 boundary edges, both x and y would be 1.

In the first case, when $S \cap T$ had at least one terminal that is demonstrated by Figure 3, we haven't mentioned anything about the nodes in $S - T$ and $T - S$. Due to the observations in Figure 4, we can now say that both of them cannot have a terminal since this would imply no terminals in $S \cap T$. Therefore, at least one of $S - T$ or $T - S$ would not have a terminal. In the complementary case, when $S \cap T$ does not have any terminal node, the situation is exactly as depicted in Figure 4, i.e., both $S - T$ and $T - S$ must have terminal nodes, since S and T are witness sets and they have to include terminal nodes by definition.

Since the cases above cover all possible situations, there are only two different situations possible when two smallest witness sets intersect but one is not contained in the other. In both of these cases, there is a "special" set which does not include any terminals and the number of edges entering and exiting the set are 1.

We now show how to construct an equivalent graph where the smallest witness sets are laminar. Since the special set does not have a terminal and there are exactly one entering and exiting edges, the special set is just a path of degree 2 non-terminals. We can contract all the degree-2 non-player nodes without changing the topology of the optimal solution, since we only care about costs of edges and paths, not the number of edges in a path. By "contract" we mean that if a node v has two edges (v, w) and (v, u), then we remove the node v and form a new edge (u, w) with cost $c(v, w) + c(v, u)$. After this contraction all of the smallest witness sets become laminar, and the desired structural properties hold, since all the nodes in the "special set" will disappear. Therefore, after this transformation, the special set will be empty, which implies that at least one of $S - T$, $S \cap T$ or $T - S$ is empty.

One has to be careful here, since we are contracting the nodes of the optimal solution, but these nodes might still have edges attached to them that are in G, but not in OPT. All of our results only depend on the structure of OPT, except for when players consider their deviations. When forming the payment for an edge $e = (i, j)$ that actually consists of several contracted edges, we must "un-contract" the edge, and let the tree $T_i(i, j)$ start paying for the edges on one side, while the tree $T_i(i, j)$ pays for the edges on the other side of the path P represented by e. If these payments "meet in the middle" to cover the cost of all edges in P, then we have paid for e, and if they do not, then we can apply the argument in the proof of Theorem 5. Therefore, the payment scheme we generate actually has a value $p_i(e)$ for every original edge of OPT, not the contracted ones. Because of this, we can accurately calculate the costs of the deviations when forming the payment scheme.

■

Because of Theorem 1, for the rest of this paper we will assume that W is a *laminar* set system. In other words, for any two smallest witness sets W_1 and W_2 (they may be the smallest witness sets of different edges), either one of them is a subset of the other or they are disjoint. Because of this, we can now speak of $W_i(i, j)$ as the unique smallest witness set in the direction of $i \to j$, and therefore e may have 2 smallest witness sets, one in each direction. In the rest of this paper, we show how to construct a stable solution where only the players in the smallest witness sets of an edge e contribute to the payment of e. In fact, this laminar property holds not only for the optimal network, but also for any minimal feasible network G', i.e., where $G' - e$ is not feasible for any $e \in E(G')$. Therefore, if we do not possess the socially optimal network but some minimal feasible network, our techniques can still be used to obtain approximate equilibria with provable cost guarantees.

3 When All Nodes Are Terminals

For the Survivable Connection Game, we do not know whether there exists an exact Nash equilibrium for all possible instances of the problem. However, for the special case where each node of G is a player-node, a Nash equilibrium is guaranteed to exist. Specifically, there is a Nash equilibrium whose cost is as much as OPT, and therefore price of stability is 1. In this section, we will prove this result by forming stable payments on the edges of the socially optimal network.

To form the payments, we will give an algorithm that decides how the cost of each edge of the socially optimal network is shared among the players. For each edge e of the socially optimal network, our algorithm only asks the adjacent terminals to contribute to the cost of e. Recall that since we are trying to form a Nash equilibrium, each terminal can only contribute to the cost of the edges it witnesses. Though a terminal can have arbitrary number of incident edges, it witnesses at most 2 of them. To see this, assume there is a player that witnesses more than 2 of its incident edges. Then at least one of the connection paths of this terminal is using at least 2 of its incident edges by the pigeonhole principle, which implies this connection path contains a cycle. Since that terminal is still 2-connected after removal of the cycle, it does not witness the incident edges included in the cycle. That simple observation allows us to see a nice substructure in the socially optimal network, which we call *chains*.

A *chain* is a path with maximal length in the socially optimal network, where each edge of the path has 2 smallest witness sets. Observe that each intermediate node of the chain is witnessing both of its incident edges in the chain, since every edge has 2 smallest witness sets, one containing each of its incident nodes. Since a terminal can witness at most 2 of its incident edges, no intermediate node of the chain witnesses any incident edge except the ones in the chain. The boundary nodes of the chain are witnessing the edge of the chain they are adjacent to. Observe that boundary nodes of the chain may or may not witness any other incident edge but if they do, this incident edge they witness will have only 1 smallest witness set, since otherwise this edge would have been part of the chain as well.

Observe that every edge e with 2 smallest witness sets is included in some chain. In the simplest case, where both of the adjacent nodes of e does not witness any other incident edges or witness one other edge with 1 smallest witness set, we will have a chain that includes only one edge, namely e. Therefore, the socially optimal network is composed of chains and edges with 1 smallest witness set. To form the stable solution, we first form the payment on the edges with only 1 smallest witness set, and then form the payments on the edges of the chains.

Since we are trying to form a stable solution, we should never ask the players to make a payment that will create an incentive of unilateral deviation. To ensure this, whenever we ask a player i to contribute to the cost of an edge e, the algorithm should compute the cost of the cheapest deviation χ_i for player i on the edges of $G-e$. Observe that all edges of OPT such that i is not contributing any payment to them can be used by i freely. Therefore, when computing χ_i, the algorithm should not use the actual cost of the edges in $G-e$, but instead for each edge f it should use the cost i would face if she is to use f. We call this the *modified cost of f for i,* and denote it by $c'(f)$. Specifically, for f not in OPT, $c'(f)=c(f)$, f's actual cost, since this is how much i would have to pay to purchase f. If i is already paying some amount $p_i(f)$ for f, then $c'(f)=p_i(f)$, since i has to continue paying this amount to use f. And finally, if i is paying nothing for f that is in OPT (or has not been asked to pay anything for it yet by our algorithm), then $c'(f)=0$. Therefore, from i's perspective, all the edges of OPT that are not adjacent to it, or that it is not witnessing, are always free, since our payment scheme will never ask i to pay for them.

Payment Algorithm The algorithm first loops through all edges of OPT that have only 1 smallest witness set. Let $e=(i,j)$ be one of those edges and without loss of generality assume i is the witnessing adjacent player. Then the algorithm asks i to pay for the whole cost of e. As mentioned above, it also computes the cost of the cheapest deviation χ_i. If $\chi_i \geq \sum_{j\neq e} p_i(j)+c_e$ then it sets $p_i(e)=c(e)$ and proceed with the next edge. If $\chi_i < \sum_{j\neq e} p_j(e)+c_e$ then the algorithm breaks (we prove below that this can never happen). If the algorithm succeeds in paying for all the edges with 1 smallest witness set, i.e., it does not break, then we consider the payment of the chains. We loop through all the chains C of OPT. Let $e_1=(n_1,n_2), e_2=(n_2,n_3),\ldots,e_k=(n_k,n_{k+1})$ be the edges of a chain C. To form the payment on the edges of C, the algorithm loops through all the edges of C starting from the leftmost edge e_1 till the rightmost edge e_k. So the payment for e_i is decided after the payments for $e_1,e_2,\ldots,e_{i-1}$ are already decided. To form the payment for e_i, the algorithm asks n_i to make the maximum payment that will not create an incentive of unilateral deviation. The algorithm then asks n_{i+1} to pay for the rest of the cost of e_i while not creating an incentive for unilateral deviation. If the adjacent nodes succeed in paying for the edge, the algorithm continues with the next edge of the chain. Otherwise, the algorithm breaks. If the algorithm succeeds in paying for the chain, it proceed to the next chain.

Observe that the payment algorithm never asks a player to make a payment that will create an incentive of unilateral deviation. Therefore, if the payment algorithm does not break at any of the intermediate stages, then it finds a Nash equilibrium, whose cost is as much as OPT. To prove our result all we need to do is to prove that the algorithm never breaks at an intermediate stage. We will prove this by constructing a feasible network cheaper than OPT whenever the algorithm breaks, which will contradict the optimality of OPT. When constructing the cheaper network, we let a subset of players deviate, i.e., take their cheapest deviation χ_i. When a player i deviates, it forms 2 disjoint paths from itself to s, which is indeed a cycle containing both i and s, which we call the *connection cycle of i*. To show feasibility of the new network, we have to show that not only the deviating player but also all other players have 2 edge-disjoint paths to s. To do this, we often use the following easy lemma.

Lemma 1. *Let C_i be the connection cycle of a player i. If a player t has 2 edge-disjoint paths to C_i, then t has 2 edge-disjoint paths to s as well.*

Proof. Observe that if the other players have 2 edge-disjoint paths to the cycle containing i and s, which we call the *connection cycle of i*, then they are also included in a cycle containing both themselves and s. To see this, let t be the player that has 2 node-disjoint paths, namely P_1 and P_2, to the connection cycle of i. Let u and v be the first nodes of the connection cycle of i that are hit by P_1 and P_2 respectively. Since all u, v, and s are lying on the same cycle, there is a path from u to v on the cycle that includes s and let this path be called P_3. Since u and v are the first nodes that P_1 and P_2 encounter in the connection cycle of i, P_3 is necessarily disjoint from P_1 and P_2. Therefore $P_1 \cup P_2 \cup P_3$ is a cycle containing both t and s and constitutes 2 edge-disjoint paths from t to s. ■

Now we are ready to prove our exact Nash equilibrium result.

Theorem 2. *The payment algorithm does not break in any intermediate stages, and therefore the payments form a Nash equilibrium. Since there is a Nash equilibrium whose cost is as much as OPT, price of stability is* 1.

Proof. For the purpose of contradiction, assume the payment algorithm breaks when it decides the payment on an edge $e = (i, j)$. First consider the case where $e = (i, j)$ has only one smallest witness set and without loss of generality let i be the witnessing adjacent node. Since e has only one smallest witness set, the payment algorithm asks i to pay for the whole cost of e. Since the algorithm broke while deciding the payment for e, player i has a cheap deviation χ_i, i.e., a strategy that makes her 2-connected and the cost is as much as she paid so far. That is, the modified cost of the strategy χ_i is less than the payments that i has agreed to so far plus c_e. We form a new network by letting player i deviate, i.e., player i applies the strategy χ_i instead of the strategy suggested by the algorithm and all other players stay with their existing strategies. This network is clearly cheaper than OPT, since the edges added (i.e., the edges of χ_i that were not already in OPT) are cheaper than the edges removed (i.e., the edges of OPT that i was paying for, but that are not in χ_i). We will prove that this new network OPT' is still a feasible network, thus obtaining a contradiction with the optimality of OPT.

We need to show that all players are still feasible in this new network. Since a player can witness at most 2 of its incident edges, player i may be witnessing and therefore may have paid for one more edge, which we will call $f = (i, k)$. Observe that at that stage of the algorithm, we haven't started deciding the payment of the chains yet. Therefore, player i may have contributed to the cost of f only if f has only one smallest witness set, which is from the side of i. Since in our payment scheme, players only pay for adjacent edges, e and f are the only two edges that i may have contributed to. Therefore, e and f are the only two edges that are in OPT, but are not in OPT'.

Suppose to the contrary that some player node t is infeasible in OPT', that is, that there do not exist 2 edge-disjoint paths from t to s. The removal of only one of e or f cannot make this happen, since this would imply that t was witnessing this edge. Since both e and f only have one witness set, t must have been witnessing the edge from the side of i, and so using this edge in the direction $i \to j$ or $i \to k$ to connect to s. Then, t is still feasible in OPT' by Lemma 1, since i and s are on the connection cycle of i, and t has 2 edge-disjoint paths to i or one path to i and an edge-disjoint path to s.

Because of the above argument, we can assume that the removal of only one of e and f does not make t infeasible. Since the removal of both e and f leaves t without 2 edge-disjoint paths to s, this implies that there exists some cut $(S, V - S)$ in OPT, with $t \in S$, $s \in V - S$, and at most three edges on the boundary, two of them being e and f. If $i \in S$, then by Lemma 1, t is still feasible in OPT', using the same argument as above. If $i \notin S$, then we will argue that i cannot be witnessing both e and f. The fact that i is witnessing both of them means that any 2 edge-disjoint paths from i to s must use both e and f. Therefore, both connection paths of i in OPT must enter S, on edges e and f. However, there are only 3 edges leaving S, and $s \notin S$, so there is no way for both these paths to leave S in an edge-disjoint manner. Therefore, i could not have been feasible in OPT, which gives us a contradiction.

We have now shown that the payment scheme succeeds in paying for all edges with only one smallest witness set. Now let us show that the algorithm will not break while deciding the payment of the chains as well. For the purpose of contradiction, assume the algorithm broke while deciding the payment of a chain C. Let $e_1 = (n_1, n_2), e_2 = (n_2, n_3), \ldots, e_k = (n_k, n_{k+1})$ be the edges of C and without loss of generality assume n_i and n_{i+1} could not pay for e_i. First, to help our understanding of chains, we prove the following lemma.

Lemma 2. *If a connection path reaches node n_j through edge e_{j-1}, then it must leave through edge e_j, and vice versa.*

Proof. Since j is witnessing both e_{j-1} and e_j, then the only way to form 2 edge-disjoint paths from j to s is to use both edges e_{j-1} and e_j. Therefore, if a connection path of terminal t reaches i through e_{j-1}, and does not leave through e_j, then j could use the connection cycle of t to form 2 edge-disjoint paths to s without using e_j, giving us a contradiction. ■

The above lemma states that if a connection path enters a chain, then it must leave the chain at its other side, not in the middle. Consider a node n_j with $j \leq i$. Unless n_j agrees to pay for the entire edge e_j, it must have a deviation preventing it from contributing any more to e_j. Call this deviation χ_j. In addition, n_{i+1} has a deviation χ_{i+1} which is preventing it from contributing more to e_i. Consider the player n_j with $j \leq i$ that is closest to i such that one of the following holds: (1) χ_ℓ for some $j < \ell \leq i+1$ passes through n_j, (2) χ_j passes through e_{j-1}, or (3) n_j does not contribute anything to e_{j-1}. We now form a solution OPT' by first letting the players $n_{j+1}, \ldots, n_i, n_{i+1}$ deviate (and also player n_j if we are in the case (2) or (3) above), and then deleting edges $e_j, e_{j+1}, \ldots, e_i$. We will show that this network is still feasible, and is cheaper than OPT, this giving us a contradiction.

To see that OPT' is cheaper than OPT, consider that we removed the contributions of players $n_{j+1}, \ldots, n_i, n_{i+1}$ (and possibly n_j), and instead added their deviations. The contributions of these players were paying for the edges $e_j, e_{j+1}, \ldots, e_{i-1}$, but were unable to pay for e_i. In the case that we let n_j deviate, it was either not contributing to e_{j-1}, or is using it in its deviation. In either case, its contribution to e_{j-1} remains the same after it deviates to χ_j. Therefore, we removed something that costs more than the contributions of the deviating players, and added their deviations. Since each deviation χ_ℓ was preventing the player from paying any more on the edge e_ℓ, the cost of it must equal the cost of n_ℓ's contributions, and so OPT' is strictly cheaper than OPT.

There is also the possibility that no node n_j satisfying one of the above conditions exists. In this case, we let all the nodes $n_1, \ldots, n_{i+1}$ deviate to form OPT', and delete the edges $e_1, \ldots, e_i$, as well as the edge paid for by n_1 that only has 1 smallest witness set, if such an edge exists. Once again, since we removed all the contributions for players $n_1, \ldots, n_{i+1}$, and added the deviations that prevented these players from paying more, then OPT' is cheaper than OPT.

We must now show that OPT' is a feasible solution. For convenience, we will first form a series of solutions OPT^{i+1}, $OPT^i, OPT^{i-1}, \ldots$, with the final one of these being OPT', and show that each of these is feasible.

We form OPT^{i+1} as follows. Let player n_{i+1} take its deviation χ_{i+1}, without deleting any edges of OPT. This is clearly a feasible solution (since we only added edges to OPT). Let C_{i+1} be the cycle that the new connection paths of n_{i+1} form with s and n_{i+1}. *Mark* the nodes and edges of the chain that are contained in C_{i+1}.

To form OPT^ℓ, we proceed as follows. First, we delete e_ℓ. Then, we look if node n_ℓ has already been marked (notice that condition (1) will correspond exactly to its being already marked). In this case, we do not let n_ℓ deviate, and set the final solution OPT' to be OPT_ℓ. Otherwise, we let n_ℓ deviate, by adding the edges of χ_ℓ to our network, although we do not add back the edges of the chain that were deleted in the previous steps. As we will show, even without the edges $e_\ell, \ldots, e_i$, n_ℓ is still 2-connected to s in this new network. We define C_ℓ analogously to above, and mark all the nodes on the chain that are contained in C_ℓ. We now look if node n_ℓ satisfies any of the three conditions mentioned above. If it does, we set OPT' to be OPT^ℓ. Otherwise, we proceed to the next node.

We will now prove inductively that each network OPT^ℓ is feasible. To do this, we must show that every node is 2-connected to s. First, notice that none of the connection paths of nodes $n_{\ell+1}, \ldots, n_{i+1}$ that are to the right of n_ℓ in the chain use the edge e_ℓ, since otherwise we would have stopped with $OPT^{\ell+1}$. Therefore, they are all still 2-connected to s. To see that n_ℓ is 2-connected to s, first consider the case where it is already marked. In this case, it is contained in a connection cycle C_k for some $k > \ell$, which is still present, since no edges contained in such cycles are ever deleted. Therefore, it is 2-connected to s. If n_ℓ is not marked, consider its deviation χ_ℓ

that we added. If χ_ℓ does not contain any edges $e_\ell, \ldots, e_i$ that have been deleted, then certainly it connects n_ℓ to s using 2 edge-disjoint paths. Otherwise, these paths must reach some of the nodes $n_{\ell+1}, \ldots, n_{i+1}$ that we already proved are 2-connected to s. By Lemma 1, this implies that n_ℓ is 2-connected to s as well. Finally, consider any other node t. We know that t was 2-connected to s in $OPT^{\ell+1}$. If its connection paths did not use e_ℓ, then it is still feasible, since e_ℓ was the only edge that was deleted. If one of its paths used e_ℓ, then by Lemma 1 and the fact that both n_ℓ and $n_{\ell+1}$ are still 2-connected to s, we know that t is also still 2-connected to s in OPT^ℓ.

The algorithm may also have stopped because it reached n_1 at the beginning of the chain. Besides e_1, n_1 may also be paying for another edge f that only has a single smallest witness set, which we also deleted. The result is still a feasible solution, by arguments similar to the arguments above for the payment of edges with only a single smallest witness set. Therefore, by induction, OPT' is a feasible solution.

Since there is a Nash equilibrium that buys the socially optimal network, price of stability is 1. ■

4 Good Stable Solutions When Not All Nodes Are Terminals

Approximation Algorithm Technique To prove our main result, we define a restricted version of the Survivable Connection Game such that this version of the game is identical to the original game, except that for each strategy p_i of a player i, the set of deviations she can take are restricted. In this restricted version of the game, a player is only allowed to deviate by changing the payments on one of her paths, instead of both of them at once, i.e., for any strategy profile $p = (p_1, \ldots, p_n)$, p'_i is a deviation for player i from her strategy p_i if for each edge e along *one* path from i to s in G_p, $p_i(e) = p'_i(e)$. In this restricted version of the game, each player should also determine her connection paths as well as the payment it makes on the edges as part of its strategy. In order to avoid ambiguity, in the rest of the paper we will use the term *stable solutions* for the equilibria of the restricted version of the game and the results we obtain for the stability of the restricted version of the game will also imply results about the Nash equilibria of the original Survivable Connection game due to Theorem 3.

Theorem 3. *A stable solution p is a 2-approximate Nash equilibrium of the original Survivable Connection Game.*

Proof. By a 2-approximate Nash equilibrium, we mean that no player can save more than a factor of 2 in her cost by changing all of her payments at once. Suppose to the contrary that player i can reduce her cost by at least a factor of 2 by switching her payments from $p_i(e)$ to $p'_i(e)$ on all edges e, i.e. $\sum_{e\in E} p'_i(e) < \frac{1}{2}\sum_{e\in E} p_i(e)$. Since in a stable solution player i makes a strictly positive payment only for the edges she uses for her connection paths, there exists a connection path P of player i such that $\sum_{e\in E} p'_i(e) < \sum_{e\in P} p_i(e)$. Then p is not a stable solution since player i can reduce her cost by replacing the payments on the edges of P by $p'_i(e)$. ■

The restricted version of the Survivable Connection Game is also of independent interest since it models the scenarios where a player wishes to keep one of her paths the same as before the deviation, the case where a complex deviation involving re-routing of both of the player paths is too much for a player, or the case where each path of a single player is managed by a different entity, which is possible when a player represents a large company.

In Figure 5(A), we have a game with one player that wants to connect from t to s through 2 edge-disjoint paths. Each thick edge has a cost of 3, each dashed edge has a cost of 1 and the total cost of the thin edges is ϵ. Any feasible solution has to include all 4 of the thin edges. Let p be a strategy of the player where she buys the 2 thick edges and all 4 of the thin edges where she uses the upper path and the lower path in Figure 5(A) as her connection paths. Please note that, though

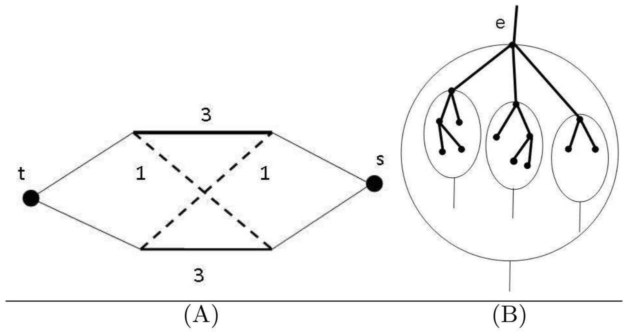

Fig. 5. (A) An example illustrating our stable solution concept. (B) Result of the Tree Generation algorithm. The ovals represent smallest witness sets.

the connection paths are uniquely determined on this game for this set of bought edges, this is not true in general, therefore they are to be specified as part of the strategy. Let p' be a strategy where the player buys the dashed and the thin edges as well as the top thick edge and for each connection path she uses a dashed edge and its 2 incident thin edges. Observe that in this strategy player buys the top thick edge although she does not use it. If the player switches her strategy from p to p', she reroutes both of her connection paths. However, p' is considered a valid deviation since she keeps the payments on one of her connection paths in p the same.

Recall that because of Theorem 3, we can restrict our attention to *stable* solutions as defined above, as results about such solutions would also imply results about approximate Nash equilibria of the general Survivable Connection Game. In the following discussion we will use the terms *price of anarchy* and *price of stability* for the ratio of the worst and best *stable* networks to the socially optimal network instead of the ratio of the Nash equilibria to the socially optimal network. The objective function that we use to measure the quality of a solution is social welfare, which for our game is the same as the total cost of the network. As we show below the price of anarchy cannot be more than $2N$ and the bound is tight. Because of this, we focus on the price of stability in the rest of the paper.

4.1 Price of Anarchy

Observe that the price of anarchy for the *Survivable Connection Game* cannot be more than $2N$. If it was more than $2N$ there would be a player whose cost is more than 2 times the socially optimal network. Since in a stable solution players would not contribute to the payment of any edge that are not on their paths, the payment of this player along one of its paths will be more than the whole socially optimal network. By replacing the payments on this path with the whole socially optimal network, this player would reduce its payments and stay feasible. Therefore, the price of anarchy cannot be more than $2N$. As Figure 6 demonstrates, this bound is actually tight and it is $2N$. If each player is able to change the payments on both paths at the same time, then the same argument applys with the bound of N, instead of $2N$.

In Figure 6, the cost of a dashed edge is 1, the cost of a thick edge is $2N$ and the total cost of the other solid edges are ϵ. Consider the strategy vector where every player contributes 2 to each thick edge, a total of $\frac{\epsilon}{N}$ to the solid edges and nothing to the dashed edges. Observe that this strategy vector describes a stable solution. Consider a possible deviation of a player. Since it has to keep its payments on one of its paths, it has to keep the payment made to one of the thick edges. It cannot change its payments to solid edges since any feasible solution should have them. All it can do is to release its payments on one of the thick edges. However, to satisfy the feasibility it has to buy both of the dashed edges which has a total cost of 2. The players does not have an incentive for unilateral

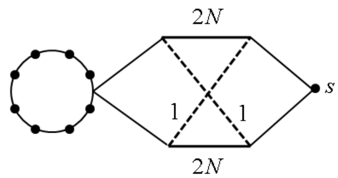

Fig. 6. An instance where Price of Anarchy is $2N$. The terminal nodes are the nodes on the circle.

deviation since all they can do is to replace a payment of 2 to one of the thick edges with a payment of 2 to the dashed edges. Therefore there is a stable solution with total cost $4N + \epsilon$. Observe that the socially optimal network would only buy the solid and the dashed edges with total cost of $2+\epsilon$. Therefore, the price of anarchy can actually be $2N$ for some instances of this problem. Notice also that Figure 6 gives an example where a stable solution is a 2-approximate Nash equilibrium, and no less, showing that the bound in Theorem 3 is tight.

4.2 Forming a Stable Solution on the Edges of OPT

In this section, we present an algorithm to find a stable strategy vector that buys OPT, which implies that the price of stability for the *Survivable Connection Game* is 1. Since a strategy of a player is composed of specifying 2 edge-disjoint paths to s and the amount of payment made on the edges, then we must specify both of these for every player.

Although our proof techniques do not require that the connection paths of the players be *node-disjoint* as well as edge-disjoint, having this property greatly simplifies the proofs. Though it may not be possible to find node-disjoint connection paths for all players in a feasible network, the following theorem states that there always exists an *equivalent graph* where each player has node-disjoint connection paths in OPT. Equivalence among the graphs means that the socially optimal network of the new graph costs as much as OPT, and that for each stable network in the new graph, there corresponds a unique stable network in the original graph with the same cost.

Theorem 4. *There exists an equivalent graph G' with a routing on the centrally optimal solution that is node-disjoint.*

Proof. To prove the result, we will give an algorithm that explicitly transforms G into G' and explicitly constructs the node-disjoint paths on it. We will first select a player node p on G arbitrarily and will try to find 2 node-disjoint connection paths for it. If we can find such 2 connection paths, we will *mark* the edges on them and proceed with the next player. A mark on an edge means that that edge is being used by the connection path of a player. If we can't find 2 node-disjoint paths from p to s, we will select just 2 edge-disjoint connection paths. These connection paths cross at some nodes of G. For each node n where these connection paths are crossing, we will *split* n into two, namely n_1 and n_2.

When we split a node n, we will remove n from G and add 2 nodes instead. Two incident edges of n that are used by one of the connection paths are assigned as incident edges of n_1 and similarly 2 incident edges used by the other connection path of p are assigned to n_2. Observe that n may have had more than 4 incident edges. We assign all other incident edges of n to n_1 or n_2 arbitrarily. We will add 2 extra edges between n_1 and n_2 and set the cost of these edges to 0. These edges are necessary for the equivalence of the new graph, as will be explained shortly. Observe that n may have been a player node. If it was, we will assign one of n_1 or n_2 arbitrarily as the node of this player. Once we do the splitting we will *mark* the edges on the connection paths and proceed with the next player.

Before forming the connection paths of the other players, we would like to show that the socially optimal network is still feasible in this new graph. Since the network we started with was feasible, each player t had 2 edge-disjoint paths before we split n. If these paths were not crossing n, t clearly still has 2 edge-disjoint paths since we haven't made any changes on the nodes and the edges on its paths. Assume t had 2 connection paths such that one or both of the paths were crossing n. Observe that in this new graph, both n_1 and n_2 (and s, by definition) are on the connection cycle of p. Therefore, each connection path of t crosses the connection cycle of p and t still has 2 edge-disjoint paths to s by Lemma 1.

Now, we explain how we form the node-disjoint paths for the remaining players. We loop through all players and form the connection paths for them one at a time. When we form the connection paths for a player t, we first grow 2 paths from t (which we know exist since t is feasible). We stop when these connection paths cross a node adjacent to the marked edges. Let these nodes be v_1 and v_2 respectively. From there on, t connects to s through 2 node-disjoint paths by using only the marked edges. Let us explain how t does that.

Assume we had assigned a ranking to all the players. The first player whose connection paths are formed had rank 1, the next player whose connection paths are formed had rank 2, and so on. Let r_1 and r_2 be the ranking of the lowest ranked players whose connection paths pass through v_1 and v_2 respectively.

Let us first consider the case where $r_1 = r_2$. Then t has connected to a connection cycle of the player with rank r_1 at 2 nodes (v_1 and v_2) and can connect to s through this connection cycle by Lemma 1. Since the player ranked r_1 already has 2 node-disjoint paths (by construction in the previous steps), then so does t, if both of its paths are node-disjoint till it connects to v_1 and v_2. Therefore, the connection paths of t may pass through a common node only until it touches the connection cycle of r_1 or right at the node it connects to the cycle, i.e., if v_1 and v_2 are actually the same node. To make the connection paths of t node-disjoint, we split the nodes that are common to both connection paths of t. Observe that splitting a node does not violate feasibility of any players as we have proven above. Furthermore, splitting of any node except v_1 will not interfere with the connection paths we have already formed since these nodes are not adjacent to the already marked edges. However, splitting of v_1 may interfere with the already formed connection paths. To illustrate, a connection path of r_1 was passing through v_1, i.e., two incident edges of v_1 were on a connection path of r_1. After splitting v_1, each one of these two edges are assigned to the newly added nodes. If they are assigned to different nodes, then the r_1-path is interrupted. To ensure that r_1 still has 2 node-disjoint paths, we should find a path between these 2 nodes that were added to the graph to replace v_1. Observe that we have grown 2 paths from t and v_1 was the first node adjacent to the already marked edges that are touched by these paths. Therefore, r_1 can use this path through t between newly added nodes, and this portion of the path is still node-disjoint from the other path of player r_1, since it does not use any nodes adjacent to marked edges. After we have split the nodes and obtained node-disjoint paths for t, we mark the edges it uses as well.

Now, we need to address the case where $r_1 \neq r_2$. Without loss of generality, let $r_1 < r_2$. Since the connection paths of r_2 are formed after the connection paths of r_1, the connection paths of r_2 both connect to the connection cycle of r_1 by construction. The connection path of t that touches to the connection cycle of r_2 can follow the cycle in any direction until it encounters the connection cycle of r_1. It should choose the direction that will lead to a node in the cycle of r_1 other than v_1. Since t now has two paths connected to the connection cycle of r_1, it can connect to s through this connection cycle by Lemma 1. Once again, the connection paths of t may pass through a common node only until it touches v_1 and v_2 and we will apply a splitting on this common node in exactly the same way as explained in the above paragraph.

We have formed a new graph G' and showed that each player has 2 node-disjoint paths on it. We also need to show that G' is equivalent to G. That is, we need to show that the centrally optimum solution costs the same in G and G', and in fact that the optimum solution in G' consists of the exact same set of edges, on which we formed the node-disjoint routes above. Observe that G' has exactly the same set of edges as G except 2 free edges added at each node splitting among the newly introduced nodes. We have already shown how to form node-disjoint paths on G' for all players on the edges that were part of the socially optimal network on G. Therefore, the network we have formed on G' has exactly the same cost as the socially optimal network on OPT. We also need to show that the network we formed is indeed the socially optimal network on G' as well. For the purpose of contradiction, assume it is not, i.e., there exists a cheaper network on G'. On this new network, the connection paths of the players may not be node-disjoint but they are necessarily edge-disjoint. Observe that exactly the same set of edges except the free edges form a feasible solution on G as well, which would contradict the fact that the network we started with was optimal in G.

To finish our proof of equivalence, we also need to show that for any stable solution on G' the corresponding feasible solution in G is stable as well. Starting with a stable solution on G' and the corresponding solution in G, all we need to show that for all players and their all possible deviations in G, there corresponds an equal cost deviation in G'. If the deviation of a player t does not pass through a node that we have split, then the statement is trivially true. If the deviation of the player passes through a node v that we have split into v_1 and v_2, the corresponding deviation in G' has exactly the same cost due to the zero-cost edges added while transforming the graph.

Therefore, since the optimal solutions cost the same, and the stable solutions in G' have corresponding stable solutions in G, we know that the price of stability in G is at most that of G'. ■

As explained in the proof of Theorem 4, the equivalent graph and the node-disjoint connection paths can be efficiently determined. Since a stable solution in the new graph corresponds to a stable solution in the original graph, it is enough to form a stable solution in G', and so we assume that there is a routing on OPT that is node-disjoint. Fix such a routing. We will now show that it is possible to pay for the edges of OPT so that this routing forms a stable solution.

Our payment scheme is formed by Algorithm 1. While deciding the payment on an edge $e = (u, v)$, the algorithm needs to form the cheapest deviation χ_i on $G - e$, for all players i in $W_u(u, v)$ and $W_v(u, v)$. For each player i in $W_u(u, v)$ or $W_v(u, v)$, we call the connection path of i that does not use e the *enduring path* of player i and denote it as E_i. To form the cheapest deviation χ_i in this algorithm, we need to be able to find the cheapest way for a player to form 2 edge-disjoint paths to s, while keeping the payments on E_i the same. As shown in Algorithm 1, this can be done by using modified costs $c'_i(f)$ for each edge f, that represent how much it costs for player i to use edge f in χ_i. Specifically, for f not in OPT, $c'_i(f) = c(f)$, the actual cost of f. For the edges f of OPT that i has not paid anything for, or for the edges in E_i, we have that $c'_i(f) = 0$, since from i's perspective, it can use these edges for free (it cannot change the payments on E_i, so from a deviational point of view, those edges are free for i to use in χ_i). For all the other edges f that i is paying $p_i(f)$ for, $c'_i(f) = p_i(f)$, since that is how much it costs for i to use f in its deviation χ_i.

We first claim that if this algorithm terminates, then the resulting payment forms a stable solution. Consider the algorithm at some stage where we are determining player i's payment to e. The cost function c'_i reflects the costs player i faces if she deviates in the final solution (not counting the cost of E_i, which stays the same). χ_i is the cost of deviating while preserving the payments on the enduring path, and so is the smallest amount player i would have to pay if she wanted to deviate from the strategy we are forming. We never allow i to contribute so much to e that her total payments exceed the cost of her cheapest deviation. Therefore, it is never in player i's interest to deviate. Since this is true for all players, this algorithm forms a stable solution if it terminates.

```
Input: The socially optimal network OPT
Output: The payment scheme for OPT
Initialize p_i(e) = 0 for all players i and edges e;
Loop until the payments for all edges are determined;
    Pick an edge e = (u,v) whose payment scheme has not been decided yet;
    Pay for all the edges in e's smallest witness sets recursively;
    Loop through all terminals i of W_u(u,v) and W_v(u,v) until e is paid for;
        Define p_i = Σ_{f∈(E\E_i)} p_i(f);
        Define p(e) = Σ_j p_j(e);
        Define c'_i(f) to be the modified cost of f for i;
        Define χ_i as the cost of the cheapest deviation by player i in G − e under c'_i;
        Set p_i(e) = min{χ_i − p_i, c(e) − p(e)}.
```

Algorithm 1: Algorithm That Generates the Payment Scheme

To prove the algorithm succeeds in paying for OPT, we need to show that for any edge e, the terminals inside its smallest witness sets will be willing to pay for e. To show this, we will actually prove a stronger statement. Specifically, for every edge $e = (i, j)$ we will generate two trees T_i and T_j in $W_i(e)$ and $W_j(e)$ rooted at i and j respectively, such that the leaves of T_i and T_j are player nodes/terminals, and all other nodes are non-player nodes. We will show that just the leaves of these trees are willing to pay for all of e, and the other terminals in the smallest witness sets are not needed. In fact, we can just as easily make our algorithm only ask the players that are leaves of these trees to contribute to the payment of e.

Tree Generation Since W is laminar, we construct the trees $T_j(i, j)$ recursively, starting with the smallest sets in W, and continuing to the sets containing those. To construct $T_j(i, j)$, we start the search in $W_j(i, j)$ from j. If j is a player then the tree is just a single node. If it is a non-player node we add all its incident edges in $W_j(i, j)$, along with their corresponding trees in their smallest witness sets from the other side. That is, for every edge (j, k) inside $W_j(i, j)$, we add the edge (j, k) and the subtree $T_k(j, k)$. These trees must have already been generated, since those witness sets are contained inside $W_j(i, j)$. We presented the tree generation in terms of the smallest witness sets but indeed it is equivalent to making a breadth-first search in $W_j(i, j)$ starting from j, except we stop when a branch arrives at a player node.

When *Tree Generation Algorithm* reaches a non-player node it adds all incident edges in $W_j(i, j)$ as well as the corresponding trees of these edges in their witness sets in the other direction, as shown in Figure 5(B). However, we haven't proven these smallest witness sets indeed exist since an edge does not necessarily have a smallest witness set on each side.

Lemma 3. *Any edge f of $T_i(e)$ generated by the Tree Generation Algorithm has a smallest witness set from the side of the lower level nodes of the tree.*

Proof. If i is a player node then *Tree Generation Algorithm* does not expand to any other nodes and returns i as the tree $T_i(i, j)$. For the purpose of contradiction, suppose that i is a non-player node, and one of the incident edges $f = (i, k)$ in $W_i(i, j)$ does not have a witness set from the side of k, i.e., $W_k(i, k)$ does not exist. Since every edge must have a witness set, this implies $W_i(i, k)$ exists. Due to laminarity of smallest witness sets, we know that $W_i(i, k) \subseteq W_i(i, j)$. This implies that (i, j) is a boundary edge of $W_i(i, k)$, since i is contained inside it, and j is outside. Since the only boundary edges of $W_i(i, k)$ are (i, j) and (i, k), then every player node inside $W_i(i, k)$ must witness the edge (i, j). In fact, this means that $W_i(i, k)$ is a witness set of (i, j). $W_i(i, k)$ is strictly smaller than $W_i(i, j)$ (since it does not contain the node k), and so this contradicts $W_i(i, j)$ being the smallest witness set of (i, j). ■

Now that we know *Tree Generation Algorithm* is well-defined, we should make sure that it actually generates trees.

Lemma 4. *The structure $T_j(e)$ generated by the Tree Generation Algorithm for any edge $e = (i, j)$ of OPT is a tree such that all leaf-nodes are player nodes and all non-leaf nodes are non-player nodes.*

Proof. We will prove this result by induction on the size of $T_j(e)$ in terms of the nodes it contains. For the base case, when j is a player node, $T_j(e)$ is a single node and therefore it is clearly a tree. Assume all the structures generated by the *Tree Generation Algorithm* that have at most n nodes are trees, with all leaves being player nodes and all non-leaves being non-player nodes. Let $T_j(e)$ be a structure with $n + 1$ nodes. Then clearly j is not a player node. Since j is a non-player node, *Tree Generation Algorithm* expands to all its incident edges in $W_j(i, j)$.

Let $(j, n_1), \ldots, (j, n_l)$ be the incident edges to j in $W_j(i, j)$ and $W_{n_1}(j, n_1), \ldots, W_{n_l}(j, n_l)$ be their respective smallest witness sets. This is illustrated in Figure 7, where the ovals represent the smallest witness sets $W_{n_1}(j, n_1), \ldots, W_{n_l}(j, n_l)$. If one of these smallest witness sets is a subset of another smallest witness set, i.e., $W_{n_u}(j, n_u) \subset W_{n_v}(j, n_v)$ for some $u, v \in 1, \ldots, l$ then the two boundary edges of $W_{n_v}(j, n_v)$ must be (j, n_u) and (j, n_v), since n_v and n_u are inside this set, and j is outside. This implies that both of the connection paths of all player nodes in $W_{n_v}(j, n_v)$ would have j as an intermediate node, which would violate node-disjointness of the routing paths. Since none of these smallest witness sets is a subset of another they are all disjoint sets due to the laminar property of smallest witness sets.

Since j is a non-player node, *Tree Generation Algorithm* expands to all its incident edges in $W_j(i, j)$, and the tree structure will be composed of j and the structures of the incident edges which are trees due to our inductive assumption. Since all these structures live in disjoint sets due to laminarity of smallest witness sets, then $T_j(e)$ is a tree as well. Since all the leaves of $T_j(e)$ are leaves of the trees in the witness sets $W_{n_1}(j, n_1), \ldots, W_{n_l}(j, n_l)$, then we know that all leaves of $T_j(e)$ are terminals and all non-leaves are non-terminals. ■

Though we have stated above that *Tree Generation Algorithm* is indeed equivalent to making breadth-first search in $W_j(i, j)$ starting from j except we stop when a branch arrives at a player node, we have explained the expansion of trees in terms of the smallest witness sets for good reasons. We now know each player node t at the leaf of a tree $T_j(e)$ is in the smallest witness set of all the edges of the path of the tree between her and j. This implies that every one of these edges *must* be used by t to connect to s, and since the connection paths of s are node-disjoint, this implies that one of the connection paths of t must simply proceed up the tree $T_j(e)$. Therefore, we know that the other connection path of t does not use any edge of this path. Lemma 5, which is one of the key lemmas for our proof, shows an even stronger property and states that the connection paths of all players leaving $W_j(i, j)$ through the other boundary edge (i.e., not the edge (i, j)) don't use any edge of $T_j(e)$ at all.

Lemma 5. *Let $W_j(i, j)$ be a smallest witness set of some arbitrary edge $e = (i, j)$. Let p be a player inside $W_j(i, j)$. Then the other connection path of p (that leaves $W_j(i, j)$ through the other boundary edge) does not use any of the edges of $T_j(e)$.*

Proof. For the purpose of contradiction, assume there exists a smallest witness set $W_j(e)$ of some edge $e = (i, j)$ that includes a player node p such that the other connection path of t uses some edges of $T_j(i, j)$. Let $W_j(e)$ be the *smallest* of such smallest witness sets in terms of the number of nodes included.

If j is a player node then this lemma trivially holds since the tree returned by the *Tree Generation Algorithm* is a single node and it does not have any edge. Therefore, consider the case where j is a non-player node. Let $(j, n_1), \ldots, (j, n_l)$ be the highest level edges of the tree $T_j(e)$ and

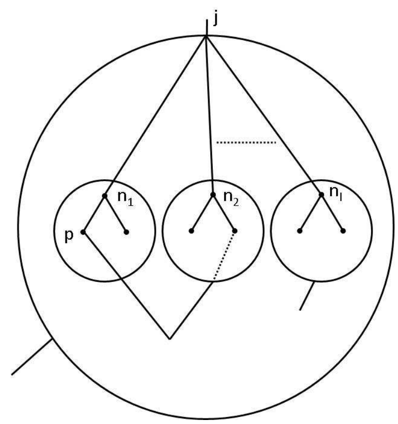

Fig. 7. Structure of Other Paths

$W_{n_1}(j,n_1),\ldots, W_{n_l}(j,n_l)$ be their respective smallest witness sets. This is illustrated in Figure 7, where the ovals represent the smallest witness sets $W_{n_1}(j,n_1),\ldots,W_{n_l}(j,n_l)$. If one of these smallest witness sets is a subset of another smallest witness set, i.e., $W_{n_u}(j,n_u) \subset W_{n_v}(j,n_v)$ for some $u, v \in 1,\ldots,l$ then both of the connection paths of all player nodes in $W_{n_u}(j,n_u)$ would have j as an intermediate node which would violate node-disjointness of the routing paths. Since none of these smallest witness sets is a subset of another they are all disjoint sets due to the laminar property of smallest witness sets.

Let us first consider the case where p is in one of these smallest witness sets. Without loss of generality, let $p \in W_{n_1}(j,n_1)$, as shown in Figure 7. Observe that the other connection path of p cannot use any edge of T_j in $W_{n_1}(j,n_1)$ since $W_j(i,j)$ is assumed to be the smallest of such smallest witness sets. Then it uses some edges of T_j in one of the other smallest witness sets, w.l.o.g. $W_{n_2}(j,n_2)$. Since $W_{n_2}(j,n_2)$ has exactly two boundary edges, the other connection path of p must use them both to enter and exit $W_{n_2}(j,n_2)$. Since j is incident to one boundary edge of both $W_{n_1}(j,n_1)$ and $W_{n_2}(j,n_2)$, both connection paths of p will have to contain j as an intermediate node, which contradicts node-disjointness of our connection paths.

Finally consider the case where $p \in W_j(i,j)$ but it is outside $W_{n_1}(j,n_1),\ldots,W_{n_l}(j,n_l)$. Observe that both of the connection paths of t cannot route through the smallest witness sets $W_{n_1}(j,n_1),\ldots,W_{n_l}(j,n_l)$, since otherwise both of the connection paths will include j as an intermediate node. Since only one of p's connection paths is routing through one of those smallest witness sets $W_{n_1}(j,n_1),\ldots,W_{n_l}(j,n_l)$, the other connection path of p cannot use any edges of $T_j(e)$, since all the edges of $T_j(e)$ are either inside these smallest witness sets, or are incident to j. ■

Now that we generated the trees $T_i(e)$ inside each smallest witness set that are disjoint from the other connection paths of the player nodes, we are ready to state our main theorem.

Theorem 5. *Algorithm 1 fully pays for OPT, and so the price of stability is* 1*. Moreover, the leaves of* $T_i(e)$ *and* $T_j(e)$ *are willing to pay for an edge* $e=(i,j)$ *without help from any other players.*

Proof. For the purpose of contradiction, suppose that for some edge e, after all players have contributed to e, $p(e) < c(e)$. For each player k of $T_i(i,j)$, consider the longest subpath of A_k until it leaves $T_i(i,j)$. Call the highest ancestor of t_k on this subpath k's deviation point, denoted d_k. Let D^i be a minimum set of deviation points such that every player in $T_i(i,j)$ has an ancestor in D^i. The minimum set of deviation points D^j for the tree $T_j(i,j)$ is defined similarly.

First we consider the simpler case where e has a witness set from one side only; i.e. $T_i(i,j)$ exists but $T_j(i,j)$ does not exist. Let $D^i = \{d_1, d_2, \ldots, d_n\}$ be the set of highest deviation points and

$t_1, t_2, \ldots, t_n$ be the players such that their alternate paths pass through $d_1, d_2, \ldots, d_n$ respectively. Make a new network called OPT' by replacing the portion of OPT above D^i by the alternate cycles of $t_1, t_2, \ldots, t_n$. Notice that the payment the players were making for e was not sufficient to buy it. In the construction of the new network, some of the players are deviating and others are sticking to their existing strategies, therefore, nobody is increasing its payment. Since they are able to buy this new network, OPT' is cheaper than OPT. Therefore, to form a contradiction we only need to show that OPT' is still feasible, i.e., all players are 2-connected to s.

First, consider the players $t_1, \ldots, t_n$. Since their alternate connection cycles connect them to s without using any more edges of the tree by Lemma 6 (which is defined below), then they are still 2-connected to s. The players not witnessing e also still satisfy their connection requirements, since the only edges taken away were edges of T_i, and if a player is witnessing an edge in this tree, then by the laminar structure of witness sets, it is also witnessing e since it must be contained inside e's witness set. Therefore, we only have to make sure that the players that are witnessing e from the side of i but are not $t_1, \ldots, t_n$ also satisfy their connection requirements.

Consider a player node p that is a leaf of T_i, but not one of $t_1, \ldots, t_n$. We cannot immediately assume it is still 2-connected, since the alternate connection cycle including the highest deviation point above it in the tree might not be edge-disjoint from p's other connection path. Let P be the unique path from p to the highest deviation point d_k above it in T_i. P is necessarily edge-disjoint from the other connection path of p by Lemma 5, and it is still in OPT' since it is below the highest deviation points. Since P and the other connection path of p are disjoint and they connect to 2 nodes of the alternate connection cycle of a deviating player, i.e., P connects to d_k and the other connection path connects to s, p has 2 edge-disjoint paths to s by Lemma 1.

We have shown that the connection requirements of players in T_i are still satisfied, but we still need to address players that witness e but are not in T_i. The key observation here is that previously the paths of these terminals passing through e were first connected to T_i through a player. This property is due to the tree generation algorithm, since *Tree Generation Algorithm* expands to all the neighbors when it reaches to a non-player node. Consider a player p that witnesses e but does not belong to T_i, and let t be the first node of T_i on p's connection path that uses e. Let P be the union of the segment of p's connection path from p to t and the unique path in T_i between t and the highest deviation point above t, which we know must be disjoint from the other connection path of p. The other connection path of p is still in OPT' by Lemma 5, so we can use Lemma 1 to show feasibility of p in OPT', since P connects to a highest deviation point and the other connection path connects to s, both of which are in the alternate connection cycle of a deviating player.

In the more general case where $e = (i, j)$ has two smallest witness sets, i.e. both $T_i(i, j)$ and $T_j(i, j)$ exist, there are two sets of minimal deviation points $D^i = \{d_1^i, d_2^i, \ldots, d_n^i\}$ and $D^j = \{d_1^j, d_2^j, \ldots, d_m^j\}$ of $T_i(i, j)$ and $T_j(i, j)$ respectively. Let $t_1^i, t_2^i, \ldots, t_n^i$ and $t_1^j, t_2^j, \ldots, t_m^j$ be the corresponding players of $W_i(i, j)$ and $W_j(i, j)$. Observe from the case where an edge has only one smallest witness set, when a player has taken its alternate connection cycle, it has only released its payments to the edges that are above its deviation point. Assume the alternate connection cycles of $t_1^i, t_2^i, \ldots, t_n^i$ do not use any edge of $T_j(i, j)$ above the set of points D^j and also assume alternate connection cycles of $t_1^j, t_2^j, \ldots, t_m^j$ do not use any edge of $T_i(i, j)$ above the set of points D^i. In this case, we can obtain a cheaper feasible network with exactly the same argument above by letting those players to deviate. Observe that the players witnessing e in the direction from i to j satisfy their connection requirements by Lemma 1 by connecting to 2 points of the alternate connection cycle of one of $t_1^i, t_2^i, \ldots, t_n^i$ through 2 edge-disjoint paths. Similarly, the players witnessing e in the direction from j to i satisfy their connection requirements by connecting to 2 points of the alternate connection cycle of one of $t_1^j, t_2^j, \ldots, t_n^j$ through 2 edge-disjoint paths.

Consider now the more complicated case where the alternate connection cycle of at least one of the players of $T_i(i, j)$ intersects $T_j(i, j)$ at a point y above D^j, without loss of generality let this player be t_1^i. Now, to construct OPT' let the players $t_1^i, t_2^i, \ldots, t_n^i$ take their alternate connection cycles similar to the previous cases, but have $t_1^j, t_2^j, \ldots, t_m^j$ stay with their existing strategies. The players witnessing e from the side of i fulfill their connectivity requirements by exactly the same argument above. For the players witnessing e from the side of j, feasibility can be shown with Lemma 1 again. All we need to show is to identify 2 edge-disjoint paths to the alternate connection cycle of t_1^i. Each player has a path to a player-node t in T_j and there is a path from t to y by using the edges of the tree. The union of these paths connects to y, which is a node of the alternate connection cycle of t_1^i, and by Lemma 5 it is necessarily edge-disjoint from the other connection path, which connects to s. ∎

Lemma 6. *Let player t_k be a leaf-node of $T_i(i,j)$. Then A_k, the alternate connection cycle of t_k, does not use any edge of $T_i(i,j)$ except in the subtree below d_k.*

Proof. By definition A_k, the alternate connection cycle of t_k, is a cycle including t_k, s and d_k and it uses the unique path in T_i between t_k and d_k. A_k is the connection cycle t_k will have if it takes the deviation of cost χ_k found by Algorithm 1 while we were deciding how the cost of e is to be shared among the players. Recall that the modified cost of χ_k is equal to whatever player t_k has paid so far in the previous iterations of the algorithm. Among various best deviations (all of which have the same cost), χ_k is the one whose corresponding alternate connection cycle includes as many ancestors of t_k as possible before including edges outside T_i. A_k is composed of 2 edge-disjoint paths between t_k and s, namely P_1 and P_2. Let P_1 be the path of A_k between t_k and s that uses the edges of T_i between t_k and d_k, and P_2 be the other connection path of A_k. We want to show that neither P_1 nor P_2 uses any edge of T_i that is not under the highest deviation point d_k.

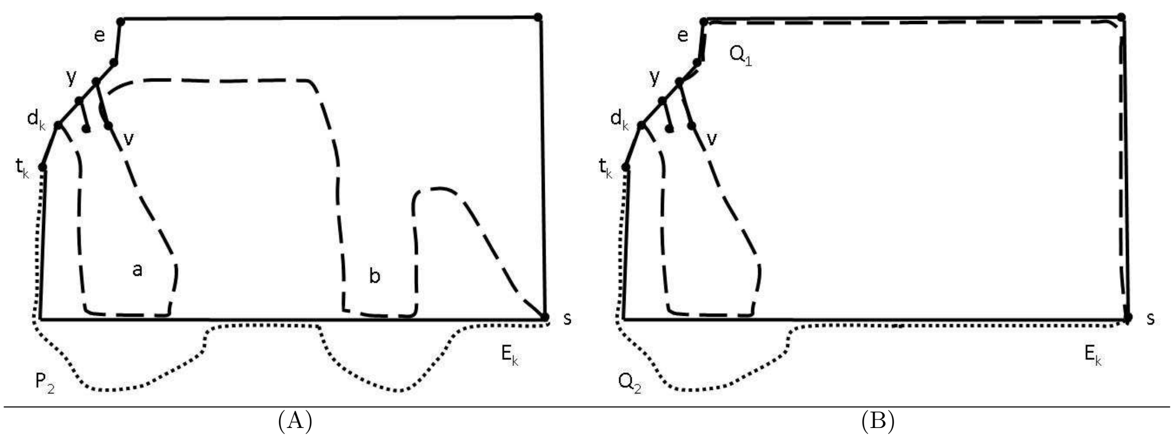

Fig. 8. (A) This figure shows the general alternate path structure of the deviation χ_k. Notice that, despite the way they appear in the figure, in general the paths P_1 and P_2 may not be using the edges of E_k in order. (B) Shows the construction of the deviating paths Q_1 and Q_2.

For the purpose of contradiction, assume A_k, i.e., P_1 or P_2 or both, uses some edges of T_i that are not under d_k. Let v be the closest node to d_k in T_i such that v is not under the subtree rooted at d_k, and the alternate connection cycle passes through v. First let us consider the case where P_1 is the alternate connection path passing through v. Let a be the subpath of P_1 between t_k and v and b be the subpath of P_1 between v and s as shown in Figure 8(A). Note that P_2 is a path from t_k to s that is edge-disjoint from P_1 since P_1 and P_2 forms a deviation.

We will prove such a deviation χ_k does not exist by constructing another valid deviation χ_k' whose modified cost is as much as the modified cost of χ_k, and which includes more ancestors of t_k in T_i before including edges outside T_i. Let the paths of the alternate connection cycle of χ_k', namely P'_1 and P'_2, be as follows. Define P'_1 as using the unique path in T_i between t_k and v and the edges of b between v and s. We will form P'_2 by using only the edges of P_2 and the enduring path E_k of t_k. Let us now describe how we form P'_2.

Recall that P_2 is a path that starts at t_k and ends at s that is edge-disjoint from P_1. If P_1 were not using any edge of the enduring path then P_2 would follow the enduring path between t_k and s, since under the modified costs E_k would be the cheapest path between t_k and s that is edge-disjoint from P_1 (recall that under the modified costs c'_k, the edges of E_k are free). If P_1, either a or b or both, is using some edges of the enduring path, then P_2 would be composed of several (maybe 0) subpaths of the enduring path which are connected to each other by subpaths outside the enduring path as shown in Figure 8(A). Let s_1 and s_2 be two adjacent subpaths of the enduring path that are used by P_2. Then some of the edges of the enduring path between s_1 and s_2 have to be used by P_1, since otherwise one could obtain a deviation cheaper than χ_k in terms of modified costs by just replacing the edges of P_2 between s_1 and s_2 with the edges of the enduring path between them. Therefore, between any two adjacent subpaths of the enduring path used by P_2, there is a subpath of the enduring path which is used by P_1. Similarly, between any two adjacent subpaths of the enduring path used by P_1, there is a subpath of the enduring path which is used by P_2 since otherwise the modified cost of χ_k would be decreased by replacing the portion of P_1 between these two subpaths with the edges of the enduring path between them. Therefore, P_1 and P_2 are using the subpaths of the enduring path in *alternating* order. To illustrate, let $s_1, s_2, \ldots, s_n$ be the subpaths of the enduring path that are used by P_1 and P_2. Then odd indexed subpaths are used by P_2 while the even indexed subpaths are used by P_1, i.e., either by a or b. For convenience, we can notice that P_2 starts at t_k and ends at s, and so say that the first and last segment s_1 and s_n are always segments of P_2, even though these segments might only consist of a single node. We form P'_2 by using only the edges of P_2 and E_k as follows. P'_2 is obtained by joining the adjacent subpaths of P_2 on the enduring path that are separated by a subpath used by a, by the edges of the enduring path. This makes some edges of P_2 redundant, and so we set P'_2 to be the cheapest path (using modified costs c'_k) from t_k to s using only edges in $(E_k - b) \cup P_2$. If several such cheapest paths exist, we choose the one that uses fewest edges outside of E_k.

In order to complete the proof, we need to show χ'_k is a valid deviation, i.e., P'_1 and P'_2 are edge-disjoint paths from t_k to s, and the modified cost of χ'_k is no more than the modified cost of χ_k. P'_2 is using only the edges of P_2 and the edges of the enduring path that are not used by P'_1. Since P_1 and P_2 are edge-disjoint, then P'_2 is disjoint from b. And since v is the closest node of T_i above d_k that is being touched by A_k, then P_2 is disjoint from the path in the tree T_i from d_k to v, and so is P'_2. Therefore, P'_1 and P'_2 are edge-disjoint.

Let us now show that the modified cost of χ'_k is no more than the modified cost of χ_k. P'_1 was obtained from P_1 by replacing a with the unique path between t_k and v. P'_2 was obtained from P_2 by joining the adjacent subpaths of P_2 on the enduring path that were separated by a subpath used by a, by the edges of the enduring path. Therefore, to show that χ'_k is not more expensive than χ_k, all we need to show is that the modified cost of the unique path between t_k and v in T_i is no more than the sum of the modified costs of a and the edges of P_2 that are not used by P'_2. We prove this as a separate technical lemma.

Lemma 7. *The modified cost of the unique path between t_k and v in T_i is no more than the sum of the modified costs of a and the edges of P_2 that are not used by P'_2.*

Proof. Let y be the lowest common ancestor of t_k and v in T_i. Since t_k dos not witness the edges between y and v, Algorithm 1 has never asked t_k to contribute to the cost of these edges and therefore the modified cost of these edges is 0. All we need to show is that the payment t_k made for the edges of T_i between d_k and y is no more than the sum of the modified costs of a and the edges of P_2 that are not used by P'_2.

Consider the time when Algorithm 1 was paying for the edges of T_i between d_k and y. To bound the payment t_k made on these edges, consider the following deviation whose alternate connection paths are Q_1 and Q_2. We are going to form Q_1 and Q_2 by using only the edges of a, the edges of P_2 that are not used by P_2', and the free edges. The cost of this deviation constrains the payment of t_k on the edges of T_i between d_k and y, since when forming these payments, t_k will pay no more than the best deviation available to her. Therefore, if we show that such a deviation exists, then the lemma holds.

We form Q_1 as follows. As shown in Figure 8(B), Q_1 uses the edges of a to reach from t_k to v and then follows the edges of T_i between v and y. To reach from y to s, Q_1 uses the edges of the connection path of t_k in OPT that leaves $W_i(i,j)$ through i, which we will refer as the *first connection path* of t_k. Note that the edges of T_i between v and y are not witnessed by t_k and therefore have a modified cost of 0. The modified cost of the edges above y of the first connection path of t_k used by Q_1 are 0 as well, since either t_k is not in one of the smallest witness sets of these edges, or the algorithm has not started paying for them yet.

We now describe how to form Q_2. Let $s_1, s_2, \ldots, s_n$ be the subpaths of P_2 on the enduring path such that s_1 includes t_k while s_n includes s. Recall that between every 2 adjacent subpaths of P_2 on the enduring path, there is a subpath used by a or b. We claim that there exist a path between s_1 and s_n (and therefore between t_k and s) that is edge-disjoint from a, using just the edges of the enduring path and the edges of $P_2 - P_2'$. We will set an arbitrary such path to be Q_2. We now prove the existence of such a path by induction.

As our inductive hypothesis, we will assume that there is a path between s_1 and s_ℓ that is edge-disjoint from a, using just the edges of E_k and $P_2 - P_2'$. If the subpath of P_1 between s_ℓ and $s_{\ell+1}$ is used by b, then we can obtain a path between s_ℓ and $s_{\ell+1}$ that is disjoint from a, by taking the path between these two subpaths on E_k. This gives us the desired path from s_1 to $s_{\ell+1}$. If instead the subpath of P_1 between s_ℓ and $s_{\ell+1}$ is used by a, then we will use the path that P_2 was using to connect them. The path that P_2 was using to connect s_ℓ and $s_{\ell+1}$ is not used by P_2', since P_2' does not have to be edge-disjoint from a and can use the edges of the enduring path between s_ℓ and $s_{\ell+1}$ freely. In either case, we form the desired path from s_1 to $s_{\ell+1}$.

Both Q_1 and Q_2 are paths between t_k and s. To show that they form a valid deviation, all we need to show is that they are edge-disjoint. Q_2 is edge-disjoint from a by construction. Q_2 does not use any of the edges of the tree between y and v since P_2 does not (by our choice of v), and E_k does not (by Lemma 5). Therefore, this Lemma holds if Q_2 is edge-disjoint from the portion of the first connection path of t_k between y and s. In fact, Q_2 *might not* be edge-disjoint from the portion of the first connection path of t_k between y and s, but then we can form another deviation, composed of Q_1' and Q_2', that is no more expensive.

Consider the first time Q_1 and Q_2 touch the portion of the first connection path of t_k between y and s. Let the nodes they touch be v_1 and v_2 respectively. Now consider only the subpaths of Q_1 and Q_2 between t_k and v_1 (and t_k and v_2). Both of these subpaths have passed through some nodes of the enduring path since they both start at t_k. Suppose the last node along the enduring path touched by these subpaths, v_3, is touched by Q_1. Then let Q_1' consist of the edges of Q_1 from t_k to v_3, followed by the edges of the enduring path between v_3 and s; and let Q_2' consist of the edges of Q_2 from t_k to v_2 followed by the edges of the first connection path of t_k between v_2 and s. Q_1' and Q_2' are disjoint paths, and they cost at most what Q_1 and Q_2 did, since all the new edges they are using are free. Suppose instead that v_3 is touched by Q_2. Then let Q_1' consist of the edges of Q_1 from t_k to v_1, followed by the edges of the first connection path of t_k between v_1 and s; and let Q_2' consist of the edges of Q_2 from t_k to v_3, followed by the edges of the enduring path of t_k between v_3 and s. Since Q_1' and Q_2' are edge-disjoint and they are using only the edges of Q_1, Q_2, and the free edges, then we have found a cheaper valid deviation. ■

The proof for the case where P_2 is the alternate connection path passing through v is analogous to the above discussion. We define a to be the subpath of P_2 from t_k to v, and b to be the rest of P_2. P_1' becomes the path in T_i from t_k to v, followed by b, while P_2' is made from the edges of P_1 and E_k in the same way as described above. ■

We have shown that there exists a 2-approximate Nash equilibrium that is as good as OPT. Since computing OPT is computationally infeasible, we present the following result.

Theorem 6. *Suppose we have a Survivable Connection Game and an α-approximate socially optimal graph G_α. Then for any $\epsilon > 0$, there is a polynomial time algorithm which returns a $2(1+\epsilon)$-approximate Nash equilibrium on a feasible graph G', where $c(G') \leq c(G_\alpha)$.*

Proof. The proof of Theorem 5 suggests an algorithm which forms a cheaper network whenever a stable solution (a 2-approximate Nash equilibrium) cannot be found. The proof followed by contradiction since the network at hand was optimal. Both of the algorithms and the proof of the theorem are based on two properties of OPT: every edge has a witness set, and there is an equivalent graph where W, the set of smallest witness sets, is laminar. Observe that both of these properties hold for any minimal feasible network, i.e., any feasible network such that removal of any edge will violate feasibility, since every edge has a witness set by definition of minimality and the proof of Theorem 1 is general enough to be valid for all minimal feasible networks, though it was stated for only socially optimal networks. Therefore; given an α-approximate socially optimal graph G_α, Theorem 5 suggests an algorithm which forms a cheaper network whenever a stable solution cannot be found.

The proof is based on following this suggested algorithm to obtain a cheaper network whenever a stable solution cannot be found. However, the improvements we consider should be substantial enough to ensure the time-bound, while they should be small enough to ensure the approximation ratio.

To find a $2(1+\epsilon)$-approximate Nash equilibrium, i.e., a solution where no player can reduce its cost by more than a factor of $2(1+\epsilon)$ by taking any deviation, we start by defining $\gamma = \frac{c(G_\alpha)\epsilon}{\alpha(1+\epsilon)m}$, where m is the total number of edges of the graph. We now use Algorithm 1 to pay for all but γ of each edge in G_α. Since G_α is not optimal, it is possible that even with the γ reduction in price there will be some edge e that the players are unwilling to pay for. If this happens, the algorithm suggested by the proof of Theorem 5, indicates how we can rearrange G_α to reduce its cost. If we modify G_α in this manner, it is easy to show that we have reduced the cost by at least γ.

Each call to Algorithm 1 can be made to run in polynomial time. Since each call which fails to form the payments reduces the cost by γ, we can have at most $\frac{\alpha(1+\epsilon)m}{\epsilon}$ calls. Therefore, in time polynomial in m and ϵ^{-1}, we obtained a network G' with $c(G') \leq c(G_\alpha)$ such that G' is a stable solution (2-approximate Nash equilibrium) if the cost of its edges were decreased by γ. (There exists constant factor approximation algorithms for this problem, including Jain's 2-approximation algorithm [22], i.e., alpha is a small constant).

For all players and for each edge e in G', we now increase $p_i(e)$ in proportion to p_i so that e is now fully paid for. Now clearly G' is fully paid for. Observe that the payment player i makes is increased to $\frac{c(G')p_i}{c(G')-m'\gamma}$, where m' denotes the number of edges in G'. To see that this is an $2(1+\epsilon)$-approximate Nash equilibrium, note that player i would not gain more than a factor of 2 by deviating before her payment was increased. Therefore, the cost of the best deviation of player i is at least $\frac{p_i}{2}$. Therefore, player i can gain at most a factor of

$$\frac{2c(G')}{c(G') - m'\gamma} \leq \frac{2c(G')}{c(G') - \frac{m'c(G_\alpha)\epsilon}{\alpha(1+\epsilon)m}} \leq \frac{2c(G')}{c(G') - \frac{c(G')\epsilon}{(1+\epsilon)}} = 2(1+\epsilon).$$

■

References

1. S. Albers, S. Eilts, E. Even-Dar, Y. Mansour, and L. Roditty. On Nash Equilibria for a Network Creation Game. *SODA*, 2006.

2. A. Agarwal, M. Charikar. Unpublished manuscript.
3. E. Anshelevich, B. Caskurlu. Exact and Approximate Equilibria for Optimal Group Network Formation. In *Proc. 17th Annual European Symposium on Algorithms*, 2009.
4. E. Anshelevich, A. Dasgupta, J. Kleinberg, É. Tardos, T. Wexler, T. Roughgarden. The Price of Stability for Network Design with Fair Cost Allocation. In *Proc. 45th Annual IEEE Symposium on Foundations of Computer Science*, 2004.
5. E. Anshelevich, A. Dasgupta, É. Tardos, T. Wexler. Near-Optimal Network Design with Selfish Agents. In *Proc. 35th ACM Symposium on Theory of Computing*, 2003.
6. E. Anshelevich, B. Shepherd, G. Wilfong. Strategic Network Formation through Peering and Service Agreements. FOCS 2006.
7. V. Bala and S. Goyal. A Non-Cooperative Model of Network Formation. In *Econometrica* 68(2000), 1181–1229.
8. C. Chekuri, J. Chuzhoy, L. Lewin-Eytan, J. Naor, and A. Orda. Non-cooperative multicast and facility location games. Proceedings of the 7th ACM Conference on Electronic Commerce (EC), Ann Arbor, Michigan (2006), pp. 72-81.
9. H. Chen, T. Roughgarden. Network Design with Weighted Players. In *SPAA* 2006.
10. G. Christodoulou and E. Koutsoupias. On the price of anarchy and stability of correlated equilibria of linear congestion games. ESA, 2005.
11. A. Epstein, M. Feldman, and Y. Mansour. Strong Equilibrium in Cost-Sharing Connection Games. *EC* 2007.
12. A. Fabrikant, A. Luthra, E. Maneva, S. Papadimitriou, and S. Shenker. On a Network Creation Game. In *PODC*, 2003.
13. A. Fabrikant, C. Papadimitriou, K. Talwar. The complexity of pure Nash equilibria. In *STOC*, 2004.
14. J. Feigenbaum, C. Papadimitriou, S. Shenker. Sharing the Cost of Multicast Transmissions. *Journal of Computer and System Sciences* 63 (2001), 21–41.
15. A. Fiat, H. Kaplan, M. Levy, S. Olonetsky, R. Shabo. On the Price of Stability for Designing Undirected Networks with Fair Cost Allocations. *Proceedings of ICALP* 2006, pp. 608–618.
16. Shai Herzog, Scott Shenker, Deborah Estrin. Sharing the "Cost" of Multicast Trees: An Axiomatic Analysis. *IEEE/ACM Transactions on Networking*, Dec. 1997.
17. M. Hoefer. Non-cooperative Facility Location and Covering Games. In *ISAAC* 2006.
18. M. Hoefer. Non-cooperative Tree Creation. In *MFCS* 2006.
19. M. Hoefer, P. Krysta. Geometric Network Design with Selfish Agents. *COCOON* 2005.
20. R. Holzman, N. Law-Yone. Strong Equilibrium in congestion games. *Games and Economic Behavior,* 21, 1997.
21. 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.
22. K. Jain. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem. *Combinatorica,* 21, 2001.
23. K. Jain and V. Vazirani. Applications of Approximation Algorithms to Cooperative Games. In *STOC*, 2001.
24. R. Johari and J. Tsitsiklis. Efficiency loss in a network resource allocation game. *Mathematics of Operations Research,* 29 (3): 407–435.
25. J. Könemann, S. Leonardi, and G. Schäfer. A group-strategyproof mechanism for Steiner forests. *SODA* 2005.
26. Y.A. Korillis, A.A. Lazaar, and A. Orda. Achieving Network Optima Using Stackleberg Routing Strategies. *IEEE/ACM Transactions on Networking,* 5(1), 1997.
27. D. Monderer, L. Shapley. Potential Games. *Games and Economic Behavior* 14(1996), 124–143.
28. N. Nisan, T. Roughgarden, É. Tardos, and V. V. Vazirani (eds.), *Algorithmic Game Theory,* Cambridge University Press.
29. R. W. Rosenthal. The network equilibrium problem in integers. *Networks*, 3:53-59, 1973.
30. T. Roughgarden. *Selfish Routing and the Price of Anarchy,* MIT Press.
31. É. Tardos. Network Games. In *Proceedings of the 36th Annual ACM Symposium on the Theory of Computing*, 2004.