Near-Optimal Network Design with Selfish Agents

Elliot Anshelevich* Anirban Dasgupta† Éva Tardos ‡ Tom Wexler §

November 2002

Abstract

We introduce a simple network design game that models how independent selfish agents can build or maintain a large network. In our game every agent has a specific connectivity requirement, i.e. each agent has a set of terminals and wants to build a network in which his terminals are connected. Possible edges in the network have costs and each agent's goal is to pay as little as possible. Determining whether or not a Nash equilibrium exists in this game is NP-complete. However, when the goal of each player is to connect a terminal to a common source, we prove that there is a Nash equilibrium as cheap as the optimal network, and give a polynomial time algorithm to find a $(1+\varepsilon)$-approximate Nash equilibrium that does not cost much more. For the general connection game we prove that there is a 3-approximate Nash equilibrium that is as cheap as the optimal network, and give an algorithm to find a $(4.65+\varepsilon)$-approximate Nash equilibrium that does not cost much more.

*Department of Computer Science, Cornell University, Ithaca, NY 14853. Email: eanshel@cs.cornell.edu Research supported in part by an NSF graduate fellowship.

†Department of Computer Science, Cornell University, Ithaca, NY 14853. Email: adg@cs.cornell.edu. Research supported by the Computer Science Department of Cornell University.

‡Department of Computer Science, Cornell University, Ithaca, NY 14853. Email: eva@cs.cornell.edu. Research supported in part by ONR grant N00014-98-1-0589.

§Department of Computer Science, Cornell University, Ithaca, NY 14853. Email: wexler@cs.cornell.edu. Research supported in part by an NSF graduate fellowship.

1 Introduction

Many networks, including the Internet, are developed, built, and maintained by a large number of agents (Autonomous Systems), all of whom act selfishly and have relatively limited goals. This naturally suggests a game-theoretic approach for studying both the behavior of these independent agents and the structure of the networks they generate. The stable outcomes of the interactions of non-cooperative selfish agents correspond to Nash equilibria. Typically, considering the Nash equilibria of games modeling classical networking problems gives rise to a number of new issues. In particular, Nash equilibria in network games can be much more expensive than the best centralized design. Papadimitriou [15] uses the term *price of anarchy* to refer to this increase in cost caused by selfish behavior. The price of anarchy has been studied in a number of games dealing with various networking issues, such as load balancing [5, 6, 13, 18], routing [17, 19, 20], facility location [22], and flow control [2, 7, 21]. In some cases [17, 19] the Nash equilibrium is unique, while in others [13] the best Nash equilibrium coincides with the optimum solution and the authors study the quality of the worst equilibrium. However, in some games the quality of even the best possible equilibria can be far from optimal (e.g. in the prisoner's dilemma). The best Nash equilibrium can be viewed as the best solution that selfish agents can agree upon, i.e. once the solution is agreed upon, the agents do not find it in their interest to deviate. Papadimitriou [15] defines the price of anarchy to study the question of how *bad* an equilibrium can be. We study the complementary question of *how good an equilibrium can be* in the context of a network design game. Schultz and Stier [20] study the ratio of the best equilibrium to the optimum, in the context of a capacitated routing game. We call this ratio the *optimistic* price of anarchy.

In this paper we consider a simple network design game where every agent has a specific connectivity requirement, i.e. each agent has a set of terminals and wants to build a network in which his terminals are connected. Possible edges in the network have costs and each agent's goal is to pay as little as possible. This game can be viewed as a simple model of network creation. Alternatively, by studying the best Nash equilibria, our game provides a framework for understanding those networks that a central authority could persuade selfish agents to purchase and maintain, by specifying to which parts of the network each agent contributes. An interesting feature of our game is that selfish agents will find it in their individual interests to *share* the costs of edges, and so effectively cooperate.

More precisely, we study the following network game for N players, which we call the *connection game*. For each game instance, we are given an undirected graph G with non-negative edge costs. Players form a network by purchasing some subgraph of G. Each player has a set of specified terminal nodes that he would like to see connected in the purchased network. With this as their goal, players 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 by any player. Each player would like to minimize his total payments, but insists on connecting all of his terminals. We allow the cost of any edge to be shared by multiple players. Furthermore, once an edge is purchased, any player can use it to satisfy his connectivity requirement, even if that player contributed nothing to the cost of this edge. Finding the centralized optimum of the connection game, i.e. the network of bought edges which minimizes the sum of the players' contributions, is the classic network design problem of the generalized Steiner tree [1, 10].

Our Results

We are most interested in deterministic Nash equilibria of the connection game, and in the optimistic price of anarchy, as the pessimistic price of anarchy in our game can be quite bad. In a game theoretic context it might seem natural to also consider *mixed* Nash equilibria when agents can randomly choose between different strategies. However, since we are modeling the construction of large-scale networks, randomizing over strategies is not a realistic option for players. We also explore the notion of an *approximate equilibrium*, and study the question of how far from a true equilibrium one has to get to be able to use the optimum solution, i.e. how unhappy would the agents have to be if they were forced to pay for the socially optimal design. We view this as a two parameter optimization problem: we would like to have a solution with cost close to the minimum possible cost, and where users would not have large incentives to deviate. Finally, we examine how difficult it is to find equilibria at all.

• In Section 3 we consider the special case when the goal of each player is to connect a single terminal to a common source. We prove that in this case, there is a Nash equilibrium, the cost of which is equal to the cost of the optimal network. In other words, with a single source and one terminal per player, the optimistic price of anarchy is 1. Furthermore, given an $\varepsilon > 0$ and an α-approximate solution to the optimal network, we show how to construct in polynomial time an $(1+\varepsilon)$-approximate Nash equilibrium (players only benefit by a factor of $(1+\varepsilon)$ in deviating) whose total cost is within a factor of α to the optimal network.

We generalize these results in two ways. First, we can extend the results to the case when the graph is directed and players seek to establish a directed path from their terminal to the common source. Note that problems in directed graphs are often significantly more complicated than their undirected counterparts [4, 9]. Second, players do not have to insist on connecting their terminals at all cost, but rather each player i may have a maximum cost $\max(i)$ that he is willing to pay, and would rather stay unconnected if his cost exceeds $\max(i)$.

• In Section 4 we consider the general case, when players may want to connect more than 2 terminals, and they do not necessarily share a single source node. In this case, there may not exist a deterministic Nash equilibrium. When deterministic Nash equilibria do exist, the costs of different equilibria may differ by as much as a factor of N, the number of players, and even the optimistic price of anarchy may be nearly N. However, in Section 4 we prove that there is always a 3-approximate equilibrium that pays for the optimal network. Furthermore, we show how to construct in polynomial time a $(4.65+\varepsilon)$-approximate Nash equilibrium whose total cost is within a factor of 2 to the optimal network.

• Finally, in the Appendix we show that determining whether or not a Nash equilibrium exists is NP-complete when the number of players is part of the input. We also show that the same problem is poly-time solvable for 2 players that have two terminals each. Since there is only a polynomial number of Nash equilibrium structures, the algorithm simply enumerates these.

Related Work

We view our game as a simple model of how different service providers build and maintain the Internet topology. We use a game theoretic version of network design problems considered in approximation algorithms [10]. Fabrikant et al [8] study a different network creation game. Network games similar to that of [8] have also been studied for modeling the creation and maintenance of social networks [3, 11]. In the network game considered in [3, 8, 11] each agent corresponds to a single node of the network, and agents can only buy edges adjacent to their nodes. This model of network creation seems extremely well suited for modeling the creation of social networks. However, in the context of communication networks like the Internet, agents are not directly associated with individual nodes, and can build or be responsible for more complex networks. There are many situations where agents will find it in their interest to *share* the costs of certain expensive edges. An interesting feature of our model which does not appear in [3, 8, 11] is that we allow agents to share costs in this manner. To keep our model simple, we assume that each agent's goal is to keep his terminals connected, and agents are not sensitive to the length of the connecting path.

Jain and Vazirani [12] study a different cost-sharing game related to Steiner trees. They assume that each player i has a utility u_i for belonging to the Steiner tree. Their goal is to give a truthful mechanism to build a Steiner tree, and decide on cost-shares for each agent (where the cost charged to an agent may not exceed his utility). They design a mechanism where truth-telling is a dominant strategy for the agents, i.e. selfish agents do not find it in their interest to misreport their utility (in hopes of being included in the Steiner tree for smaller costs). Jain and Vazirani give a truthful mechanism to share the cost of the minimum spanning tree, which is a 2-approximation for the Steiner tree problem. This game is quite analogous to our single source network creation game considered in Section 3. We can view the maximum payment $\max(i)$ of agent i as his utility u_i. However, in our game there is no central authority designing the Steiner tree or cost shares. Rather, we study Nash equilibria of our game. Also, in our game, agents must offer payments for each edge of the tree (modeling the cooperation of selfish agents), while in a mechanism design framework, agents pay the mechanism for the service, and do not care what edge they contribute to.

2 Model and Basic Results

The Connection Game We now formally define the connection game for N players. Let an undirected graph $G = (V, E)$ be given, with each edge e having a nonnegative cost $c(e)$. Each player i has a set of terminal nodes that he must connect. 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. Any edge e such that $\sum_i p_i(e) \geq c(e)$ is considered *bought*, and G_p denotes the graph of bought edges with the players offering payments $p = (p_1, \ldots, p_N)$. Since each player must connect his terminals, all of the player's terminals must be connected in G_p. However, each player tries to minimize his total payments, $\sum_{e \in E} p_i(e)$.

A Nash equilibrium of the connection game is a payment function p such that, if players offer payments p, no player has an incentive to deviate from his payments. This is equivalent to saying that if p_j for all $j \neq i$ are fixed, then p_i minimizes the payments of player i. A $(1+\varepsilon)$-approximate Nash equilibrium is a function p such that no player i could decrease his payments by more than a factor of $1+\varepsilon$ by deviating, i.e. by using a different payment function $p_i{}'$.

Basic Results Here we present several useful properties of Nash equilibria in the connection game. Suppose we have a Nash equilibrium p, and let T^i be the smallest tree in G_p connecting all terminals of player i. It easily follows from the definitions that (1) G_p is a forest, (2) each player i only contributes to costs of edges on T^i, and (3) each edge is either paid for fully or not at all.

It is not always the case that selfish agents can agree to pay for a network. There are instances of the connection game which have no deterministic Nash equilibria. In Figure 1, there are 2 players, one wishing to connect node s_1 to node t_1, and the other s_2 to t_2. Now suppose that there exists a Nash equilibrium p. By Property 1 above, in a Nash equilibrium G_p must be a forest, so assume without loss of generality it consists of the edges a, b, and c. By Property 2, player 1 only contributes to edges a and b, and player 2 only contributes to edges b and c. This means that edges

a and c must be bought fully by players 1 and 2, respectively. At least one of the two players must contribute a positive amount to edge b. However, neither player can do that in a Nash equilibrium, since then he would have an incentive to switch to the strategy of only buying edge d and nothing else, which would connect his terminals with the player's total payments being only 1.

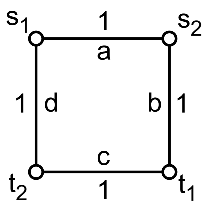

Figure 1: A game with no Nash equilibria.

We have now shown that Nash equilibria do not have to exist. However, when they exist, how bad can these Nash equilibria be? As mentioned above, the price of anarchy often refers to the ratio of the worst (most expensive) Nash equilibrium and the optimal centralized solution. In the connection game, the price of anarchy is at most N, the number of players. This is simply because if the worst Nash equilibrium p costs more than N times OPT, the cost of the optimal solution, then there must be a player whose payments in p are strictly more than OPT, so he could deviate by purchasing the entire optimal solution by himself, and connect his terminals with smaller payments than before. More importantly, there are cases when the price of anarchy actually equals N, so the above bound is tight. This is demonstrated with the following example. Suppose there are N players, and G consists of nodes s and t which are joined by 2 disjoint paths, one of length 1 and and one of length N. Each player has a terminal at s and t. Then, the worst Nash equilibrium has each player contributing 1 to the long path, and has a cost of N. The optimal solution here has a cost of only 1, so the price of anarchy is N. Therefore, the price of anarchy could be very high in the connection game. However, notice that in this example the *best* Nash equilibrium (which is each player buying $\frac{1}{N}$ of the short path) has the same cost as the optimal centralized solution. We have now shown that the price of anarchy can be very large in the connection game, but the optimistic price of anarchy remains worth considering, since the above example shows that it can differ from the (conventional) price of anarchy by as much as a factor of N.

3 Single Source Games

As we show in the Appendix, determining whether or not Nash equilibria exist in a general instance of the connection game is NP-Hard. Furthermore, even when equilibria exist, they may be significantly more expensive than the centrally optimal network. In this section we define a class of games in which there is always a Nash equilibrium, and the optimistic price of anarchy is 1. Furthermore, we show how we can use an approximation to the centrally optimal network to construct a $(1 + \epsilon)$-approximate Nash equilibrium in poly-time, for any $\epsilon > 0$.

Definition 3.1 *A* single source game *is a game in which all players share a common terminal s, and in addition, each player i has exactly one other terminal t_i.*

Theorem 3.2 *In any single source game, there is a Nash equilibrium which purchases T^*, a minimum cost Steiner tree on all players' terminal nodes.*

Proof. Given T^*, we present an algorithm to construct payment strategies p. We will view T^* as being rooted at s. Let T_e be the subtree of T^* disconnected from s when e is removed.

Algorithm 3.3

```
Initialize p_i(e) = 0 for all players i and edges e.
Loop through all edges e in T* in reverse BFS order.
  Loop through all players i with t_i ∈ T_e until e paid for.
    If e is a cut in G set p_i(e) = c(e).
    Otherwise
      Define c'(f) = p_i(f) for all f ∈ T* and
             c'(f) = c(f)   for all f ∉ T*.
      Define χ_i to be the cost of the cheapest path from s to
        t_i in G \ {e} under modified costs c'.
      Define p_i(T*) = Σ_{f∈T*} p_i(f).
      Define p(e) = Σ_j p_j(e).
      Set p_i(e) = min{χ_i − p_i(T*), c(e) − p(e)}.
```

We first claim that if this algorithm terminates, the resulting payment forms a Nash equilibrium. Consider the algorithm at some stage where we are determining i's payment to e. The cost function c' is defined to reflect the costs player i faces if he deviates in the final solution. We never allow i to contribute so much to e that his total payments exceed his cost of connecting t_i to s. Therefore it is never in player i's interest to deviate. Since this is true for all players, p is a Nash equilibrium.

We will now prove that this algorithm succeeds in paying for T^*. In particular, we need to show that for any edge e, the players with terminals in T_e will be willing to pay for e. Assume the players are unwilling to buy an edge e. Then each player has some path which explains why it can't contribute more to e. We can use a carefully selected subset of these paths to modify T^*, forming a cheaper tree that spans all terminals and doesn't contain e. This would clearly contradict our assumption that T^* had minimum cost.

Define player i's *alternate path* A_i to be the path of cost χ_i found in Algorithm 3.3, as shown in Figure 2(a). If there is more than one such path, choose A_i to be the path which includes as many ancestors of t_i in T_e as possible before including edges outside of T^*. To show that all edges in T^* are paid for, we need the following technical lemma concerning the structure of alternate paths.

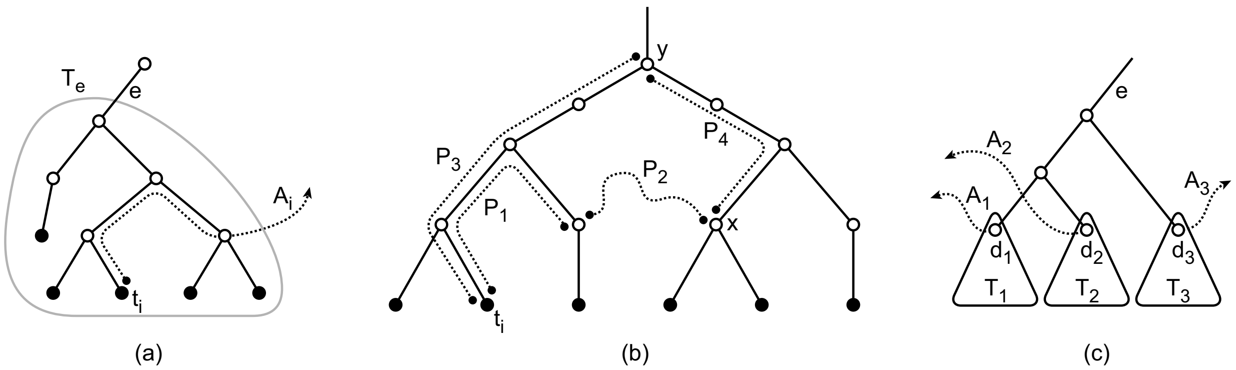

Figure 2: Alternate paths in single source games.

Lemma 3.4 *Suppose A_i is i's alternate path at some stage of the algorithm. Then there are two nodes v and w on A_i, such that all edges on A_i from t_i to v are in T_e, all edges between v and w are in $E \setminus T^*$, and all edges between w and s are in $T^* \setminus T_e$.*

Proof. Once A_i reaches a node w in $T^* \setminus T_e$, all subsequent nodes of A_i will be in $T^* \setminus T_e$, as all edges f in $T^* \setminus T_e$ have cost $c'(f) = 0$ and the source s is in $T^* \setminus T_e$. Thus, suppose A_i begins with a path P_1 in T_e, followed by a path P_2 containing only edges not in T^*, before reaching node x which is in T_e, as shown in Figure 2(b). Let y be the lowest common ancestor of x and t_i in T_e. Observe that P_1 is strictly below y. Define P_3 to be the path from t_i to y in T_e, and define P_4 to be the path from y to x in T_e. We now show that under the modified cost function c', $P_3 \cup P_4$ is at least as cheap as $P_1 \cup P_2$. Since $P_1 \cup P_2$ includes a higher ancestor of t_i than A_i (namely y), this would contradict our choice of A_i.

Consider the iterations of the algorithm during which player i could have contributed to edges in P_3. At each of these steps the algorithm computes a cheapest path from t_i to s. At any time, player i's payments are upper bounded by the modified cost of his alternate path, which is in turn upper bounded by the modified cost of any path, in particular A_i. Furthermore, at each of these steps the modified costs of all edges in A_i above x are 0. Therefore i's contribution to P_3 is always at most the modified cost of $P_1 \cup P_2$. The modified cost of P_4 is always 0, as none of the edges in P_4 are on player i's path from t_i to s in T^*. Together these imply that $c'(P_3 \cup P_4) = c'(P_3) \leq c'(P_1 \cup P_2)$. ■

Thus, players' alternate paths may initially use some edges in T_e, but subsequently will exclusively use edges outside of T_e. We use this fact in the following lemma.

Lemma 3.5 *Algorithm 3.3 fully pays for every edge in T^*.*

Proof. Suppose that for some edge e, after all players have contributed to e, $p(e) < c(e)$.

For each player i, consider the longest subpath of A_i containing t_i and only edges in T_e. Call the highest ancestor of t_i on this subpath i's *deviation point*, denoted d_i. Note that it is possible that $d_i = t_i$. Let D be a minimum set of deviation points such that every terminal in T_e has an ancestor in D.

Suppose we have every player i with a terminal t_i in D deviate to A_i, as shown in Figure 2(c), paying his modified costs to each edge. Any player i deviating in this manner does not increase his total expenditure, as player i raised $p_i(e)$ until p_i matched the modified cost of A_i. The remaining players leave their payments unchanged.

We claim that now the edges bought by players with terminals in T_e connect all these players to $T^* \setminus T_e$. To see this, first consider any edge f below a deviation point d_i in D. By Lemma 3.4, player i is the only deviating player who could have been contributing to f. If i did contribute to f, then f must be on the unique path from t_i to d_i in T_e. But by the definition of d_i, this means that f is in A_i. Thus player i will not change his payment to f.

Define T_i to be the subtree of T_e rooted at d_i. We have shown that all edges in T_i have been bought. By Lemma 3.4, we know that A_i consists of edges in T_i followed by edges in $E \setminus T$ followed by edges in $T^* \setminus T_e$. By the definition of c', the modified cost of those edges in $E \setminus T^*$ is their actual cost. Thus i pays fully for a path connecting T_i to $T^* \setminus T_e$.

We have assumed that the payments generated by the algorithm for players with terminals in T_e were not sufficient to pay for those terminals to connect to $T^* \setminus T_e$. However, without increasing any players' payments, we have managed to buy a subset of edges which connects all terminals in T_e to $T^* \setminus T_e$. This contradicts the optimality of T^*. Thus the algorithm runs to completion. ■

Since we have also shown that the algorithm always produces a Nash equilibrium, this concludes the proof of the theorem. ■

We have shown that the optimistic price of anarchy in a single source game is 1. However, the algorithm for finding an optimal Nash equilibrium requires us to have a minimum cost Steiner tree on hand. Since this is often computationally infeasible, we present the following result.

Theorem 3.6 *Suppose we have a single source game and an α-approximate minimum cost Steiner tree T. Then for any $\varepsilon > 0$, there is a poly-time algorithm which returns a $(1+\varepsilon)$-approximate Nash equilibrium on a Steiner tree T', where $c(T') \leq c(T)$.*

Proof Sketch. The proof of Theorem 3.2 suggests such an algorithm which forms a cheaper tree whenever a Nash equilibrium cannot be found. To ensure polynomial-time convergence, we force the algorithm to make only substantial improvements. See the Appendix for further details. ■

Extensions Both theorems 3.2 and 3.6 can be proven for the case where our graph G is directed, and players wish to purchase paths from t_i to s. The one difficulty arises from the fact that in proving Lemma 3.4, we assume that paths, in particular P_4, can be traversed in either direction. In the directed case, this is no longer necessarily so. We can get around this problem with a more complicated argument, showing that if e cannot be paid for, then by removing segments of alternate paths between pairs of subtrees T_i and T_j, we can connect all terminals in T_e at lower cost.

Once we have shown that our theorems apply in the directed case, we can extend our model and give each player i a maximum cost $\max(i)$ beyond which he would rather pay nothing and not connect his terminals. It suffices to make a new terminal t'_i for each player i, with a directed edge of cost 0 to t_i and a directed edge of cost $\max(i)$ to s.

4 General Connection Games

In this section we deal with the general case of players that can have different numbers of terminals and do not necessarily share the same source terminal. As stated before, in this case the price of anarchy can be as large as N, the number of players. However, even the optimistic price of anarchy may be quite large in this general case.

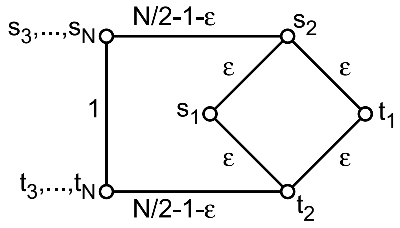

Figure 3: A game with high optimistic price of anarchy.

Consider the graph illustrated in Figure 3, where each player i owns terminals s_i and t_i. The optimal centralized solution has cost $1+3\varepsilon$. If the path of length 1 was bought, each player $i > 2$ will not want to pay for any ε edges, and therefore the situation of players 1 and 2 reduces to the example in Section 2 of a game with no Nash equilibria. Therefore, any Nash equilibrium must purchase the path of length $N-2$. In fact, if each player $i > 2$ buys $\frac{1}{N-2}$ of this path, then we have a Nash equilibrium. Therefore, for any $N > 2$, there exists a game with the optimistic price of anarchy being nearly $N - 2$.

Because of this, we cannot hope to be able to provide cheap Nash equilibria for the multi-source case. Therefore, we consider how cheap α-approximate Nash equilibria with small α can be, and obtain the following result.

Theorem 4.1 *For any optimal centralized solution T^*, there exists a 3-approximate Nash equilibrium such that the purchased edges are exactly T^*.*

Let T^* be an optimal centralized solution, which we know is a forest. Define a segment of a tree T as a path $P \subseteq T$ such that all interior nodes of P have degree 2. For simplicity of the proof, we assume that every segment of T^* is a single edge, since this proof is easily extendable to the general case where this need not hold. We also assume that T^* is a tree, since otherwise we can apply this proof to each component of T^*. Let T^i be the unique smallest subtree of T^* which connects all terminals of player i.

Definition 4.2 *A connection set S of player i is a subset of edges of T^i such that for each connected component C of the graph $T^* \setminus S$, we have that either*
(1) any player that has terminals in C has all of his terminals in C, or
(2) player i has a terminal in C.

Intuitively, a connection set S is a set such that if we removed it from T^* and then somehow connected all the terminals of i, then all the terminals of all players are still connected in the resulting graph. Since T^* is optimal, this means that any connection set S with respect to i must be cheaper than any deviation of i from a strategy where i pays for S. We now have the following lemma, the proof of which follows directly from the definition of a connection set. This lemma basically says that if each player buys at most α connection sets in full, then we have an α-approximate Nash equilibrium.

Lemma 4.3 *Let p be a payment function purchasing T^* which obeys the following properties.*
(1) If $p(e) > 0$, then e is bought fully by a single player.
(2) Each player i only buys edges which actually lie in his tree T^i.
If the set of edges that each player buys is a union of at most α connection sets, then p is an α-approximate Nash equilibrium.

Proof of Theorem 4.1. Now all that we need to prove Theorem 4.1 is a payment scheme for arbitrary games such that the conditions in Lemma 4.3 hold with $\alpha = 3$. We now exhibit such a scheme on the edges of T^*. First, each player i pays for the edges belonging only to T^i and no other tree T^j. This is clearly a connection set, so we want each player to pay for at most 2 more. We can contract the edges now paid for, forming a new tree T^* which the players must pay for, and on which each edge belongs to at least two different T^i's. For convenience, we will now talk of terminals making payments instead of players. The total payment of a player is just the sum of the payments of his terminals.

Now we recursively assign terminals to the edges of T^*. Each edge will be assigned a terminal, which will pay for it. At the end of each phase of the recursion, we generate a set of directed paths R to be paid for during the following phase. Each of these paths starts at some terminal t, and ends at a node of a path paid for in the previous phase. We will call such a path $R(t)$, since for each terminal there will be at most 1 path starting at that terminal, and we will call the last node of this path $r(t)$. Figure 4(a) shows a decomposition of T^* into these paths after this recursion is done. Initially, select $R(t)$ to be a path from an arbitrary terminal t to another terminal of the same player in T^i, direct this path away from t, and set $R = \{R(t)\}$. Each phase proceeds as follows.

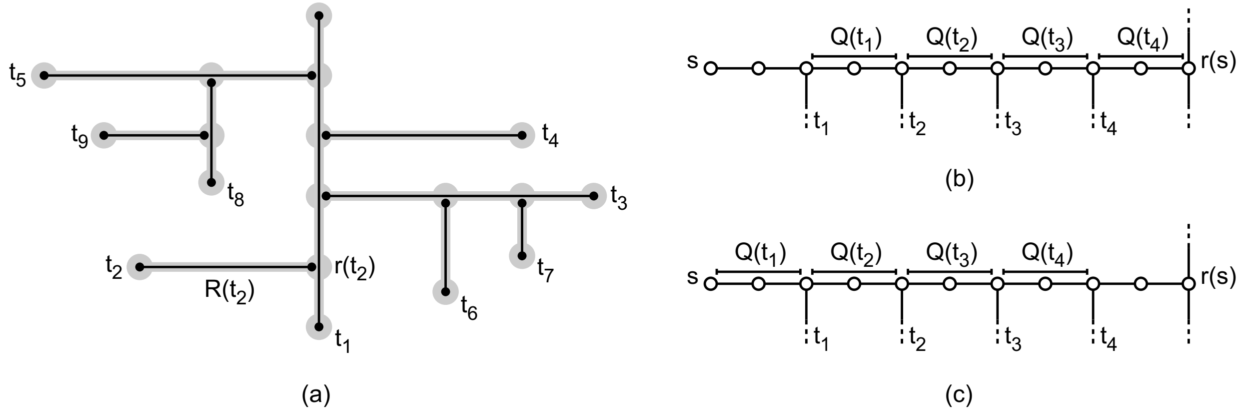

Figure 4: (a) A decomposition of T^* into paths $R(t)$; (b)(c) The paths $Q(t)$ for a single player i.

Step 1: Consider the set of directed paths R generated by the previous phase. For each path $R(s)$ in this set, do the following. Let $v_1, v_2, \ldots$ be the nodes of $R(s)$, ordered in the direction away from s (so that $s = v_1$). We assumed that T^* has no non-terminal nodes of degree 2, so each v_k must be either a terminal or have edges not in $R(s)$ incident to it. Consider the subtree rooted at v_k obtained by cutting edges (v_{k-1}, v_k) and (v_k, v_{k+1}). Define S_k to be the set of terminals in this subtree such that the player i who owns them has terminals outside this subtree as well, i.e. either (v_{k-1}, v_k) or (v_k, v_{k+1}) is in T^i. Do not include s in S_1, and set this set to be empty for $r(s)$. We form a path $Q(t)$ for each terminal $t \in S_k$ that belongs to i as follows.

- If i does not own s, pick the smallest $\ell > k$ such that S_ℓ contains a terminal of player i.
  - If such a node v_ℓ exists, set $Q(t)$ to be the path from v_k to v_ℓ.
  - If no such node v_ℓ exists, and T^i contains $r(s)$, set $Q(t)$ to be the path from v_k to $r(s)$.
  - If no such node v_ℓ exists, and T^i does not contain $r(s)$, set $Q(t)$ to be the path from $v_{k'}$ to v_k, where $v_{k'}$ is the first node of $R(s)$ such that $S_{k'}$ contains a terminal belonging to player i.
- If i owns s, pick the largest $\ell < k$ such that S_ℓ contains a terminal of player i.
  - If such a node v_ℓ exists, set $Q(t)$ to be the path from v_ℓ to v_k.
  - If no such node v_ℓ exists, set $Q(t)$ to be the path from s to v_k.

Figure 4(b) illustrates what the paths $Q(t)$ for terminals t of i look like if i does not own s and T^i contains $r(s)$. In this figure, the terminals t_1 through t_4 are all the terminals of i in any of the sets S_k. Figure 4(c) shows the same thing in the case that i owns s.

Definition 4.4 *A <u>link</u> L is a maximal set of edges of $R(s)$ such that for every edge $e \in L$, the set of paths $Q(t)$ that contain e is exactly the same, for $t \in \cup_{v_k \in R(s)} S_k$.*

A link L is really a connection set of any terminal t with $L \subseteq Q(t)$, since if we take out L and add a path connecting endpoints of $Q(t)$, then all of the endpoints of all other paths $Q(t')$ remain connected. We would like to choose exactly one terminal t from each set S_k and have these pay for the path $R(s)$ together, with each one paying for at most 1 link L, and with $L \subseteq Q(t)$. We do this by constructing the following bipartite graph (A, B).

Step 2: Let A have a node for each link in $R(s)$, and let B be the nodes of $R(s)$. Form an edge between a node $v_k \in B$ and node $L \in A$ if there exists some terminal $t \in S_k$ such that $L \subseteq Q(t)$. For $X \subseteq A$, define $\partial(X)$ to be the set of nodes in B which X has edges to. According to Hall's Matching Theorem, there exists a matching in this bipartite graph with all nodes of A incident to an edge of the matching if for each set $X \subseteq A$, $|\partial(X)| \geq |X|$. Arrange the edges of the links of X in the order they appear in $R(s)$. We want to show that between every link of X, there appears a node belonging to $\partial(X)$.

Consider some edge e of X that is not the first one in $R(s)$. Suppose this edge belongs to link L, and the previous edge e' in X to some link L'. Since these are different links, there must be some path $Q(t)$, which either ended or began between e' and e, with $t \in S_k$ and $v_k \in \partial(X)$. Suppose it ended there. If t belongs to the same player as s, then v_k is between e' and e by definition of $Q(t)$. Otherwise, there must be some terminal t' belonging to the same player as t such that $Q(t')$ begins at the place where $Q(t)$ ends. If $Q(t')$ contains e, then we are done. Otherwise, continue this argument with t' instead of t, until $Q(t')$ contains either e or e', one of which must happen by construction of $Q(t)$. Therefore, we obtain a node v_k in $\partial(X)$ that is located between e' and e. The case of $Q(t)$ beginning between e' and e is similar.

Now let L be the first link of X that appears in $R(s)$, and suppose player i owns s. It cannot be that L belongs to a single path $Q(t)$ where i owns t. Otherwise, this would mean that $L \in T^i$ but in no other tree T^j, and these edges have already been paid for and contracted. Therefore, there must be some $Q(t)$ containing L such that t belongs to a different player than s. By construction of $Q(t)$, $t \in S_k$ for some v_k that comes before L. This means that there is a node of $\partial(X)$ before L.

Therefore, $|X| \leq |\partial(X)|$, and so we can assign a terminal $t_k \in S_k$ to each node v_k such that these terminals pay for all the links of $R(s)$ while each paying for at most 1 link, with that link in $Q(t_k)$. If t_k pays for link L, assign the payment of L to the player who owns t_k.

Step 3: Finally, we must generate the set of paths R for the next phase. For each terminal t_k chosen in Step 2, let $R(t_k)$ be the path from t_k to v_k as above. Together, these paths compose R.

We have now generated a payment p which satisfies all of the conditions of Lemma 4.3. All that is left to prove Theorem 4.1 is that with this payment, $\alpha = 3$. Since the edges that belong only to T^i and no other tree T^j form a connection set, we prove this inductively by showing that the set of edges paid for by each player's terminals in the above scheme is a union of at most 2 connection sets. See the Appendix for details. ■

Extensions We have now shown that in any game, we can find a 3-approximate Nash purchasing the optimal network. As a lower bound, in the Appendix we give a simple sequence of games such that in the limit, any Nash purchasing the optimal network must be at least $(\frac{3}{2})$-approximate.

Since the proof of Theorem 4.1 is constructive, it actually contains a polynomial-time algorithm for generating a 3-approximate Nash equilibrium on T^*. We can use the ideas from Theorem 3.6 to create an algorithm which, given an α-approximate Steiner forest T, finds a $(3+\varepsilon)$-approximate Nash equilibrium which pays for a Steiner forest T' with $c(T') \leq c(T)$. However, this algorithm requires a polynomial-time optimal Steiner tree finder as a subroutine. The algorithm of Theorem 4.1 generates at most 3 connection sets for each player i. We can check if each connection set is actually cheaper than the cheapest deviation of player i, which is found by the cheapest Steiner tree algorithm. If it is, then we have a $(3+\varepsilon)$-approximate Nash equilibrium. Otherwise, we can replace this connection set with the cheapest deviation tree and run this algorithm over again. If we use a 2-approximate Steiner forest T, and an optimal Steiner tree 1.55-approximation algorithm from [16] as our subroutine, then the above algorithm actually gives a $(4.65+\varepsilon)$-approximate Nash equilibrium on T' with $c(T') \leq 2 \cdot OPT$, in time polynomial in n and ε^{-1}.

Acknowledgements

This project arose in a class taught by Eric Friedman. It began as joint work with Ranjith Rajagopalan. We would like to thank him for many useful insights and contributions.

References

[1] A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner problem on networks. SIAM Journal on Computing, 24(3) 445-456 (1995).

[2] A. Akella, R. Karp, C. Papadimitrou, S. Seshan, and S. Shenker. Selfish Behavior and Stability of the Internet: A Game-Theoretic Analysis of TCP. In Proceedings of SIGCOMM, 2002

[3] V. Bala and S. Goyal. A Non-Cooperative Model of Network Formation. In *Econometrica* 68(2000), pages 1181–1229.

[4] M. Charikar, C. Chekuri, T. Cheung, Z. Dai, A. Goel, S. Guha. Approximation Algorithms for Directed Steiner Problems. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[5] A. Czumaj, P. Krysta, and B. Vöcking. Selfish Traffic Allocation for Server Farms. In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing*, pages 287–296, 2002.

[6] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. In *Proceedings of the 13th Annual Symposium on Discrete Algorithms*, pages 413–420, 2002.

[7] D. Dutta A. Goel, and J. Heidemann. Oblivious AQM and Nash Equilibrium. In *Proceeding of the IEEE Infocom*, 2003.

[8] A. Fabrikant, A. Luthra, E. Maneva, S. Papadimitriou, and S. Shenker. On a Network Creation Game. Unpublished manuscript.

[9] U. Feige. A treshold of $\lg n$ for approximating set-cover. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 314-318, 1996.

[10] M. Goemans, and D. Williamson. A General Approximation Technique for Constrained Forest Problems. SIAM Journal on Computing, 24:296-317, 1995

[11] H. Heller and S. Sarangi. Nash Networks with Heterogeneous Agents. Working Paper Series 2001, E-2001-1, Virginia Tech.

[12] K. Jain and V. Vazirani. Applications of Approximation Algorithms to Cooperative Games. In the *Proceedings in the Annual ACM Symposium on Theory of Computing*, 2001.

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

[14] M. Mavronicolas and P. Spirakis. The price of selfish routing. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, pages 510–519, 2001.

[15] C. Papadimitriou. Algorithms, Games, and the Internet. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, pages 749–753, 2001.

[16] G. Robins and A. Zelikovsky Improved Steiner Tree Approximation in Graphs. In *Proc. 10th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pages 770–779, 2000.

[17] T. Roughgarden. The price of anarchy is independent of the network topology. In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing*, 2002.

[18] T. Roughgarden. Stackelberg scheduling strategies. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, pages 104–113, 2001.

[19] T. Roughgarden and É. Tardos. How bad is selfish routing? In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 93–102, 2000. Full version to appear in *Journal of the ACM.*

[20] A. Schulz and N. Stier Moses. On the Performance of User Equilibria in Traffic Networks. To appear in *ACM/SIAM Symposium on Discrete Algorithms*, 2003.

[21] S. Shenker. Making Greed Work in Networks: A Game-Theoretic Analysis of Switch Service Disciplines. In *IEEE/ACM Transactions on Networking*, pages 819–831, 1995.

[22] A. Vetta. Nash equilibria in competitive societies with applications to facility location, traffic routing and auctions. In *Proceedings of the Annual IEEE Symposium on the Foundations of Computer Science*, 2002.

Appendix

Proof of Theorem 3.6. To find a $(1+\varepsilon)$-approximate Nash equilibrium, we start by defining $\gamma = \frac{\varepsilon c(T)}{(1+\varepsilon)n\alpha}$. We now use Algorithm 3.3 to attempt to pay for all but γ of each edge in T. Since T 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 proof of Theorem 3.2 indicates how we can rearrange T to decrease its cost. If we modify T in this manner, it is easy to show that we have decreased its cost by at least γ. At this point we simply start over with the new tree and attempt to pay for that.

Each call to Algorithm 3.3 can be made to run in polynomial time. Furthermore, since each call which fails to pay for the tree decreases the cost of the tree by γ, we can have at most $\frac{(1+\varepsilon)\alpha n}{\varepsilon}$ calls. Therefore in time polynomial in n, α an ε^{-1}, we have formed a tree T' with $c(T') \leq c(T)$ such that the players are willing to buy T' if the edges in T' have their costs decreased by γ.

For all players and for each edge e in T', we now increase $p_i(e)$ in proportion to p_i so that e is fully paid for. Now T' is clearly paid for. To see that this is a $(1+\varepsilon)$-approximate Nash equilibrium, note that player i did not want to deviate before his payments were increased. If we let m' be the number of edges in T', then i's payments were increased by

$$\gamma \frac{p_i(T')}{c(T') - m'\gamma} m' = \frac{\varepsilon c(T) p_i(T') m'}{(1+\varepsilon)n\alpha(c(T') - m'\gamma)} \leq \frac{\varepsilon c(T) p_i(T')}{\alpha(1+\varepsilon)(1-\varepsilon)c(T')} \leq \varepsilon p_i(T').$$

Thus any deviation yields at most an ε factor improvement. ■

Proof of Theorem 4.1 (continued). Since the edges that belong only to T^i and no other tree T^j form a connection set, all we need to show is that the set of edges paid for by each player's terminals in the scheme of Section 4 is a union of at most 2 connection sets. We will prove this inductively, using the paths $R(s)$ as generated in that scheme. Let $T(s)$ be the tree containing $R(s)$ obtained by removing $r(s)$ from T^*. The inductive hypothesis is the following:

Case 1: If T^i is entirely contained in $T(s)$, then the edges paid for by terminals of i are the union of at most 2 connection sets.

Case 2: If T^i is not contained in $T(s)$, and i owns s, then the edges paid for by terminals of i in $T(s)$ is a single connection set, and $r(s)$ is connected to a terminal of i in $T(s)$ by a path on which i does not pay for anything.

Case 3: If T^i is not contained in $T(s)$, and i does not own s, the edges paid for by terminals of i in $T(s)$ is a single connection set.

We perform induction on the phases during which the paths $R(s)$ were generated, backwards. If $R(s)$ was generated in the last phase, this means that it is an empty path consisting only of the terminal s, and $T(s) = \{s\}$. The empty set here is clearly a connection set.

Let $R(s)$ and i be as in Case 2, which we will prove first. Let S be the edges of $T(s)$ for which i pays, and consider the components of $T^* \setminus S$, which we want to show satisfy one of the properties of Definition 4.2. All the components which do not intersect $R(s)$ are taken care of by the inductive hypothesis, so let C be a component of $T^* \setminus S$ which intersects $R(s)$. If C contains s, then C contains a terminal of i, so we are done. If C contains $r(s)$, then because of the way edges of $R(s)$ are paid for, there must be some path $R(t)$ ending at a node of $R(s)$, with t a terminal of i, and with $r(t)$ in C. By the inductive hypothesis, there is a path from a terminal belonging to i to $r(t)$ with no edges paid for by i, and so there is a path to $r(s)$ with the same property, as desired. The only possibility left is that there are some edges e_1 and e_2 of $R(s)$, paid for by i, that border C on left and right, cutting it off from the rest of $R(s)$. If e_1 and e_2 are in the same link, then by definition of a link, we have that C satisfies the first property of Definition 4.2. Otherwise, by construction there must be some path $R(t)$ ending at a node of $R(s)$, with t a terminal of i, and with $r(t)$ in C, and so C satisfies property (2) of Definition 4.2 by the argument above.

The proof of Case 3 is very similar to Case 2, so all that is left is to prove is Case 1. Suppose T^i is entirely contained in $T(s)$, and let S and C be as above. Assume that i pays for at least 1 edge of $R(s)$. In the proof of Case 2, we already showed that if C is cut off from $R(s)$ by two edges of the same link, then C satisfies Property (1) of Definition 4.2. Otherwise C must be cut off from $R(s)$ by edges belonging to different links of $R(s)$. By construction of the payments, there can only be one component C such that there is no path $R(t)$ ending at a node of $R(s)$, with t a terminal of i, and with $r(t)$ in C. If the links bordering that component are L and L', then if we pretend that i does not pay for either L or L', the set of edges paid for by i becomes a connection set, since C now has a terminal of i in it. Therefore, S is a union of 2 connection sets. The only case left to address is if i does not pay for any edges of $R(s)$. Then, there is only 1 component C of $T^* \setminus S$ intersecting $R(s)$, and we do not have to worry about any other components because of the inductive hypothesis. If we take any link bordering C and remove it from S, then S becomes a connection set. Therefore, the set of edges paid for by i is a union of at most 2 connection sets. ■

Lower bounds for approximate Nash on the optimal network

Claim 4.5 *For any $\epsilon > 0$, there is a game such that any equilibrium which purchases the optimal network is at least a $(\frac{3}{2} - \epsilon)$-approximate Nash equilibrium.*

Proof. Construct the graph H_N on $2N$ vertices as follows. Begin with a cycle on $2N$ vertices, and number the vertices 1 through $2N$ in a clockwise fashion. For vertex i, add an edge to vertices

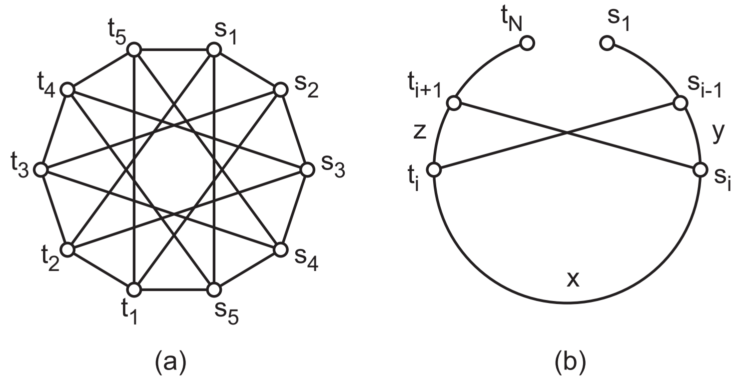

Figure 5: A game with best Nash equilibrium on OPT tending to at least a $\frac{3}{2}$-approximation.

$i+N-1 \mod (2N)$ and $i+N+1 \mod (2N)$. Let all edges have cost 1. Finally, we will add N players with 2 terminals, s_i and t_i, for each player i. At node j, add the label s_j if $j \leq N$ and t_{j-N} otherwise. Figure 5(a) shows such a game with $N = 5$.

Consider the optimal network T^* consisting of all edges in the outer cycle except (s_1, t_N). We would like to show that any Nash which purchases this solution must be at least $(\frac{6N-21}{4N-11})$-approximate. This clearly would prove our claim.

First we would like to show that players 1 and N are not willing to contribute too much to any solution that is better than $(\frac{3}{2})$-approximate. Suppose we have such a solution. Define x to be player 1's contribution to his connecting path in T^*, and define y to be his contribution to the remainder of T^*. Thus player 1 has a total payment of $x+y$. Player 1 could deviate to only pay for x. Furthermore, player 1 could deviate to purchase only y and the edge (s_1, t_N). If we have a solution that is at most $(\frac{3}{2})$-approximate, then we have that $\frac{x}{x+y} \geq \frac{2}{3}$ and similarly $\frac{y+1}{x+y} \geq \frac{2}{3}$. Taken together this implies that $\frac{1}{x+y} \geq \frac{1}{3}$, or $x+y \leq 3$. A symmetric argument shows that player N is also unwilling to contribute more than 3.

Thus we have that the remaining $N-2$ players must together contribute at least $2N-7$. Therefore there must be some player other than 1 or N who must contribute $\frac{2N-7}{N-2}$. Suppose player i is such a player. Let x be the amount that player i contributes to his connecting path in T^*. Let y be his contribution to (s_{i-1}, s_i) and let z be his contribution to (t_i, t_{i+1}). See Figure 5(b).

Now consider three possible deviations available to player i. He could choose to contribute only x. He could contribute y and purchase edge (s_{i-1}, t_i) for an additional cost of 1. Or he could contribute z and purchase edge (s_i, t_{i+1}), also for an additional cost of 1. We will only consider these possible deviations, although of course there are others. Note that if i was contributing to any other portion of T^*, then we could remove those contributions and increase x, y, and z, thereby strictly decreasing i's incentive to deviate. Thus we can safely assume that these are i's only payments, and hence

$$x + y + z \geq \frac{2N-7}{N-2}.$$

Since i is currently paying at least $x+y+z$, we know that his incentive to deviate is at least $\max(\frac{x+y+z}{x}, \frac{x+y+z}{y+1}, \frac{x+y+z}{z+1})$. This function is minimized when $x = y+1 = z+1$. Solving for x we find that

$$x \geq \frac{4N-11}{3N-6}.$$

Thus player i's incentive to deviate is at least

$$\frac{x+y+z}{x} \geq \frac{3x-2}{x} = 3 - \frac{2}{x} \geq 3 - 2\frac{3N-6}{4N-11} = \frac{6N-21}{4N-11}.$$

Therefore as N grows, this lower bound on player i's incentive to deviate tends towards $\frac{3}{2}$. Note that in this proof, we only considered one optimal network, namely T^*. If we modify G by increasing the costs of all edges not in T^* by some small $\varepsilon > 0$, then T^* is the only optimal network. Repeating the above analysis under these new costs still yields a lower bound of $\frac{3}{2}$ for the best approximate Nash on T^* in the limit as N grows and ε tends to 0. ■

NP Completeness

In this section, we present a brief proof that determining the existence of Nash equilibria in a given graph is NP-complete if the number of players is $O(n)$. We present a reduction from 3-SAT to show that the problem is NP-hard. The graph constructed will have unit cost edges.

Consider an arbitrary instance of 3-SAT with clauses C_j and variables x_i. For each variable x_i construct the gadget shown in Figure 6a. When player i buys the left path or right path, this corresponds to x_i being set to be true or false, respectively. We will call i a variable player.

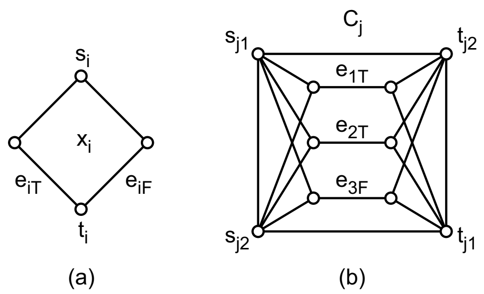

Figure 6: Gadgets for the NP-completeness reduction.

Next, for each clause C_j, construct the gadget shown in Figure 6b. Note that this example is for the case where $C_j = (x_1 \vee x_2 \vee \overline{x}_3)$. Furthermore, the edges labeled e_{1T}, e_{2T}, and e_{3F} are the same edges that appear in the variable gadgets. In other words, among all clauses and variables, there is only one edge labeled x_{iT} and only one labeled x_{iF}. We will call both players on this gadget clause players.

Suppose that there is a satisfying assignment A in our 3-SAT instance. Consider the strategy in which variable player i fully buys the left path if x_i is true in A and fully buys the right path otherwise. Since this is a satisfying assignment, by our construction each clause gadget has at least one interior edge fully paid for by a variable player. For each clause C_j, let e be one such edge, and let both players on this gadget buy the unique path of length 3 that connects their terminals which uses edge e. It is easy to see that this forms a Nash equilibrium.

Suppose now that there is a Nash equilibrium. From our example in Figure 1, we know that if we consider any clause and look at the two corresponding players, these players must use some edges other than just those on the perimeter of the gadget. In fact, by simple case checking it is clear that in a Nash equilibrium, no edges on the perimeter of the gadget are bought at all. This implies that variable players only select paths within their gadget. Furthermore, it implies that variable players must pay fully for their entire path. Suppose i is a variable player who has selected the left (true) path, but has not paid fully for the second edge in that path. The remainder of this cost must be paid for by some clause player or players. But for such a clause player to use this edge, he must also buy two other edges, which are not used by any other player. Hence such a clause player must pay strictly more than 2. But there is always a path he could use to connect of cost exactly 2, so this can not happen in a Nash equilibrium. Thus we have established that variable players pay fully for their own paths.

Now consider any clause gadget. Since we have a Nash equilibrium, we know that only internal edges are used. But since each clause player can connect his terminals using perimeter edges for a cost of exactly 2, one of the interior variable edges must be bought by a variable player in each clause gadget. If we consider a truth assignment A in which x_i is true if and only if player i selects the left (true) path, then this obviously satisfies our 3-SAT instance, as every clause has at least one variable forcing it to evaluate to true.

Therefore, this game has a Nash equilibrium if and only if the corresponding formula is satisfiable, and since this problem is clearly in NP, determining whether a Nash equilibrium exists is NP-Complete.

Connection game with two players is solvable

In this section, we outline a polynomial time algorithm to find a Nash equilibrium for a simplified two-player version of the game in which each player i has only two terminals, a source s_i and a sink t_i. For any tree T, we use T_{xy} to represent the unique path in T from node x to y. Also, $d(x, y)$ always stands for the shortest distance from x to y in the original graph G. We base our algorithm on the possible structure of the Nash equilibrium in the network. Since the Nash network is always a forest with four terminals, it either has to be two disjoint paths or a tree with T_{s_1,t_1} and T_{s_2,t_2} sharing a contiguous set of edges. Let the two endpoints of the shared subpath be called merge-nodes. The algorithm just enumerates over all possible Nash equilibrium structures. We restrict the search space to polynomial size by using the properties of a Nash tree. The following algorithm constructs a Nash tree T if one exists.

Step 1 : We first consider a special case of this problem. Here we simply want to find out whether the two players have a Nash equilibrium in which they buy node-disjoint paths. The paths taken by each player at Nash equilibrium would be shortest paths, so the only possibly profitable deviations for each player are the ones where he uses the edges purchased by the other player. Hence, in order to make sure there are no profitable deviations, we need to build the paths so that they are "far" from each other. Specifically, for some d_1, the nodes allowed for player 2 are the set of all $w \in G$ simultaneously satisfying $d(s_1, w) \geq d(s_1, t_1) - d_1$ and $d(t_1, w) \geq d_1$. The nodes allowed for player 1 also satisfy symmetrical conditions for some d_2. To test if there is such a node-disjoint Nash, we simply iterate over all possible values of d_1 and d_2. If for any choice of d_1 and d_2, the shortest paths through the restricted nodes are actually the shortest paths in the original graph, we have a Nash. Else, we claim that the graph has no node-disjoint Nash.

Step 2 : If we did not find a node-disjoint Nash equilibrium in Step 1, we choose a pair of nodes $\{u, v\}$ as our possible merge-nodes and assign one terminal from each player to each of the merge nodes. Suppose terminals s_1, s_2 are assigned to u and t_1, t_2 to v. The following steps are then iterated over all possible such assignments and over all possible choices of the merge-nodes. If we do not succeed after trying out all possible such combinations and node-pairs, we declare that there is no Nash in the game instance.

We first consider a subgame in which each terminal is a player that wants to connect to its merge-node, and look for a node-disjoint Nash equilibrium in this game. This subgame is solvable by an easy extension of the algorithm in step 1. Let the paths obtained be named T_{s_iu} and T_{t_iv} for

$i = 1, 2$.

Next, we need to construct the shared portion of the tree. We use the following criterion to create a restricted subset of nodes H. Intuitively, we do not allow the shared path to traverse nodes which might have a possibly cheaper shortcut from any of the terminals. The graph H is defined by the following recursive process.

- Initialize $H = G$.
- Consider nodes in H. For $w \in H$, find the two shortest paths Q_1 from u to w and Q_2 from v to w using only nodes currently in H. If we cannot find either of these paths, remove w from H. Else, we remove w from H if either $d(s_1, u) + d(s_2, u) + c(Q_1) > d(s_1, w) + d(s_2, w)$ or $d(t_1, v) + d(t_2, v) + c(Q_2) > d(t_1, w) + d(t_2, w)$.
- Continue this process until we cannot eliminate any more nodes from H. Each time we eliminate any node from H, we need to iterate over all the remaining nodes to see if there has been any change in the shortest paths of other nodes making them possible candidates for elimination.

Lemma 4.6 *If there is a Nash with merge-node pair $\{u, v\}$, then all nodes w in the shared portion of the Nash tree are included in the graph H corresponding to this merge-node pair and terminal assignment.*

Proof. Suppose there exists a Nash equilibrium network M such that u and v are the merge nodes. The subpaths M_{s_1u}, M_{t_1u}, M_{s_2v} and M_{t_2v} must all be shortest paths and are node-disjoint. If not all nodes of M_{uv} are in H, consider the first node w of M_{uv} to be eliminated. Then, since w is part of a Nash strategy, and all other nodes of M_{uv} were still in H, we must have $d(s_1, u)+d(s_2, u)+c(M_{uw}) \leq d(s_1, w) + d(s_2, w)$. Similarly, $d(t_1, v) + d(t_2, v) + c(M_{vw}) \leq d(t_1, w) + d(t_2, w)$. Hence, w cannot be eliminated. ■

We also observe that the graph H does not include any nodes from the paths T_{s_iu} and T_{t_iv} except for u and v. We first show that H does not have any nodes from $T_{s_1u} \cup T_{s_2u} - \{u\}$. Take any $w \in T_{s_1u}$ with $w \neq u$. We have that $d(s_1, w) < d(s_1, u)$ and $d(s_2, w) \leq d(s_2, u) + d(u, w)$ because of the triangle inequality, and the shortest path Q_{uw} in H satisfies $c(Q_{uw}) \geq d(u, w)$. Putting together the inequalities, w satisfies the condition for elimination. Similarly, we can prove that H does not have any node from $T_{t_1v} \cup T_{t_2v} - \{v\}$. This means that any path from u to v built over nodes in H will be disjoint from the paths T_{s_iu} and T_{t_iv}. We move on to the final step of the algorithm.

Step 3 : After constructing the graph H, find the shortest path T_{uv} from u to v in the induced graph of H. We claim that we can find a Nash payment scheme p on the tree $T = T_{uv} \cup \bigcup_{i=1,2}(T_{s_iu} \cup T_{t_iv})$. Each player i pays for the part of T which is used by himself only. In the shared portion, the payment by each player on any segment T_{uw} or T_{wv} is restricted by the cost of his alternate paths to w. This completes the algorithm.

The following theorem proves the correctness of the algorithm.

Theorem 4.7 *This algorithm finds a Nash equilibrium iff one exists.*

Proof. We first prove that the node-disjoint solution returned, if any, is indeed a Nash. Suppose the algorithm finds two paths P_1 and P_2 for the two players. Let $d(t_i, P_j)$ indicate the minimum distance of t_i from any node on P_j. Suppose the algorithm obtains a solution for (d_1, d_2). Now, the only possibly profitable deviations for player 1 are ones where he uses the edges paid for by player 2. All nodes w on P_2 satisfy $d(s_1, w) \geq d(s_1, t_1) - d_1$ and $d(t_1, P_2) \geq d_1$. Hence, the cost of any such deviation will be at least $d(s_1, w) + d(t_1, P_2) \geq d(s_1, t_1)$. So player 1 does not have any incentive to deviate, and neither does player 2.

Now, we show that the algorithm returns a node disjoint Nash if there was one in the original game. If the Nash paths are P_1 and P_2, set $d_1 = d(t_1, P_2)$ and $d_2 = d(t_2, P_1)$. For this d_1 and d_2, since no player has any incentive to deviate, we have that each node w on P_2 satisfies $d(s_1, w) \geq d(s_1, t_1) - d_1$. The similar condition is satisfied for each node on P_1. Thus the shortest paths lie in the restricted subsets, and our algorithm will find the Nash solution for the pair (d_1, d_2).

Next, we prove that the final algorithm is correct. In proving that the tree T is a Nash, we first observe that for each player i, it is sufficient to consider alternate paths which are node-disjoint from the tree T except at its endpoints. This is because each alternate path of player i must have subpaths which are disjoint from T and are cheaper than the payment of player i on the corresponding edges of T. To prove that T is a Nash it is enough to show that there are no such node-disjoint alternate paths which are cheaper. Further, if $\mathcal{Y}$ is an alternate path for player i, since the subpaths $T_{s_i u}$ and $T_{t_i u}$ are also Nash strategies in the corresponding subgames, it cannot be the case that $\mathcal{Y}$ is providing cheaper alternate strategies to only these subpaths. Thus it is sufficient to consider alternate paths which are shortest paths from s_i and t_i to some node w on T_{uv}. Henceforth, our arguments in showing T to be a Nash will be directed at proving that no such alternate paths exist for the given payment schema.

The reasoning that this payment scheme is a Nash relies on the shared nodes being in H with respect to the merge-node pair u and v. As argued before, the path T_{uv} is node-disjoint with subpaths $T_{s_i u}$ or $T_{t_i v}$. Consider the first node w on T_{uv} such that we cannot find sufficient payment for the segment T_{uw}. This implies both players have better alternate paths Π_1 and Π_2 which take them to w. As argued, it is enough to consider alternate paths Π_1 and Π_2 which are shortest paths from s_i to w. Thus, $c(\Pi_1) + c(\Pi_2) = d(s_1, w) + d(s_2, w)$. Also since w is in H, and T_{uw} is the shortest path in H from u to w, $d(s_1, w) + d(s_2, w) \geq c(T_{s_1 u}) + c(T_{s_2 u}) + c(T_{uw})$. Thus, Π_1 and Π_2 cannot both be cheaper alternate paths.

A similar reasoning holds for all segments T_{wv}. This shows that the entire tree is successfully paid for, and the payment scheme is a Nash equilibrium. By Lemma 4.6, if a Nash exists with merge points u and v, then the above algorithm finds it. ■