EXACT AND APPROXIMATE EQUILIBRIA FOR NETWORK FORMATION AND CUT GAMES

By

Bugra Caskurlu

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: COMPUTER SCIENCE

Approved by the
Examining Committee:

Elliot Anshelevich, Thesis Adviser

Sanmay Das, Member

Petros Drineas, Member

Malik Magdon-Ismail, Member

John Mitchell, Member

Rensselaer Polytechnic Institute
Troy, New York

June 2010
(For Graduation August 2010)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my research advisor Elliot Anshelevich, who not only introduced me the wonderful world of algorithmic game theory but also helped me to be a good researcher. I would like to send my special thanks to my committee members, Sanmay Das, Petros Drineas, Malik Magdon-Ismail and John Mitchell for their extremely useful comments and encouragements.

I would like to thank excellent friends I had in RPI whose company helped me a lot. I thank Ahmet Camtepe, Fikret Sivrikaya, Evrim Acar Ataman, Matt Edman, Jason Sanchez and Ameya Hate for being excellent office mates and friends. I will never forget the help I get while I was getting prepared for my dissertation from my precious friends Murat Semerci, Cagri Yanik, Basak Oztan and Leyla Gasimli. I would like to send my special appreciation to Cagatay Bilgin, who has been more than a friend for me.

I owe this degree as well as all the achievements of my life entirely to my family for their constant love, support and trust for me. Thank you very much Mom, Dad and Tolga for being the best family one can have. I love you!

ABSTRACT

In this thesis, we develop several interesting network formation and cutting games and study the existence and computability of exact or approximately stable solutions.

We first introduce the survivable version of the game theoretic network formation model known as the Connection Game. We prove the existence of cheap Nash equilibrium solutions and give polynomial time algorithms to compute them. We provide the first work where arbitrary sharing is used as a cost-sharing mechanism for the formation of complex, i.e., not acyclic, graphs. In order to obtain the results, we have also proven some structural properties of survivable networks, which are of interest in traditional contexts as well.

Later we define a network game called *Group Network Formation Game*, which represents the scenario when strategic agents are building a network together. In this game, players correspond to nodes of a graph and the players can have extremely varied connectivity requirements. For example, there might be several different types of nodes in the graph, and a player desires to connect to at least one of every type (so that this player's connected component forms a Group Steiner Tree). Or instead, a player might want to connect to at least k other player nodes. The first example is useful for many applications where a set of players attempt to form groups with complementary qualities. The second example corresponds to a network of servers where each server want to be connected to at least k other servers so that it can have a backup of its data; or in the context of IP networks, a set of ISPs that want to increase the reliability of the Internet connection for their customers, and so decide to form multi-homing connections through k other ISPs. Many other types of connectivity requirements fit into our framework and in this thesis we give algorithms to efficiently compute cheap exact and approximate Nash equilibria.

Finally, we define a network game called *Network Cutting Game*, which considers a dual approach to network formation games, i.e., players are not trying to connect to certain parts of a network, rather they want to disconnect themselves from certain nodes of a graph by cutting edges. This game models the scenario, where a set of players are trying to protect themselves from a diffusive process, like a computer virus, or disinformation, contamination in the water suply, etc. Cutting the edges may not mean destroying the edges, rather it corresponds to taking security actions, like putting sentries or installing firewalls on routers. In this thesis, we investigate the game-theoretic version of several standard cut problems like, s-t cut, Multicut and Multiway cut and give algorithms that return cheap exact or approximate Nash equilibrium. At the end of the thesis, we list some interesting open problems and future directions for researchers.

The work in this thesis is supported by RPI and NSF CCF-0914782.

CHAPTER 1
INTRODUCTION

Graph theory, cryptography, boolean logic, and many other major areas of computer science have been studied as subdisciplines of mathematics for centuries, but had limited applications beyond being intellectual curiosities until the invention of digital computers in 1940s. The first digital computers were both very expensive and slow. In spite of the Church-Turing thesis, this made it hard to imagine that they would have any usefulness beyond calculators. Then, computation was not an option for neither engineering nor social disciplines. Mathematical models, even important ones used for policy-making, were extremely simple (mostly linear) because of the cumbersome nature of manually solving problems and were far from being accurate representations of reality.

Since the invention of integrated circuit in 1958, the density, speed, and storage capacity of computers has increased exponentially. This tremendous speedup of computers has overjoyed computer scientists, since we already had an established theory of computing for many possible applications of computers. The early 1950s were spent mostly on increasing the usefulness of computers for the scientific community. Sophisticated numerical analysis techniques, high level programming languages, effective machine learning and simulation techniques, and robust operating systems were all introduced before 1980. This work was shocking for scientists from other engineering disciplines since because it enabled them to replace their simple models with more sophisticated and realistic ones.

The most revolutionary invention of this time was to construct a global network of computers by connecting them with wide-band communication media. This was the foundation of the internet. With the introduction of world wide web, we had the largest network constructed by the human kind, one that expands beyond all borders, and with it a new era: The "information age". All of these tremendous achievements have taken place in a very short time–only a few decades ago, even the most mundane of modern technology was science fiction for even the most promenant in the field. In the words of Cambridge professor Douglas Hartree, who had built the first differential analyzers in England: “All the calculations that would ever be needed in this country could be done on the three digital computers which are now being built – one in Cambridge, one in Teddington, and one in Manchester. No one else will ever need machines of their own, or would be able to afford to buy them.”

In 1980s and 1990s, a lot of effort was spent for creating robust, reliable, secure, and efficient networking and communication protocols which are accompanied with brand new networking and mobile technologies. These protocols set rules, and satisfied their design requirements if the components of the computing system stuck to these rules, but these extensively large networks were being constructed, maintained and developed by thousands of different companies and corporations. It was practically and politically impossible to enforce strong regulations on the network since it was both extensively large and spanned over a hundred nations. Participants could disobey the protocols if it was beneficial for them to do so. As a result of the very sudden introduction of networking technologies, we ended up having extensively large networks with no or very little theory behind their operation.

Transportation engineers have already faced a similar problem. They had a network of tens of millions of drivers that probably consider their personal comfort rather than overall performance. A computer network, however, was much less regulated and much larger, with trillions of packets traveling every day at the speed of light. It was crucial to understand the operation of networks that are governed and used by many agents.

Computer networking was not the only area of computer science that had to deal with the actions of multiple agents. The computing facilities of a company or a school are shared by the employees or the students, and the performance of the system depends on the actions of the people. Advancements in artificial intelligence and robotics was making it possible to create multi-agent systems and we had artificial systems composed of many interacting robots, expanding the need for a theory investigating the interactions of various agents beyond the networking community. Before establishing such a theory, one major question must be asked: ”How do the agents act?” An accurate answer is impossible without certain reasonble and fitting assumptions. According to Bertrand Russell "Although this may seem a paradox, all exact science is dominated by the idea of approximation."

Most researchers made the assumption that the agents are self-interested, i.e., trying to maximize their own individual comfort (or minimize their individual cost) which is sometimes referred as the "rational act". There is well-established mathematical theory called Game Theory which is the study of the interactions of rational agents. A game consists of a set of players, a set of moves (or strategies) available to those players, and a specification of payoffs for each combination of strategies. Since Game Theory was a mathematical discipline, most of the results were dealing with the *existence* of "stable points of a game", i.e., the solutions that are acceptable for all the interacting agents. For computer scientists, *computation* of these stable points was as important as their existence. The original theory was extended to include algorithms to find such stable solutions, and this new field is called Algorithmic Game Theory.

In the rest of the section, we present some basic notions of game theory necessary to understand the rest of the thesis and outline our major results with some of the earlier results relevant to our work. In Section 2, we study the survivable version of the game theoretic network formation model known as the Global Connection Game, originally introduced in [6]. In Section 3, we introduce and study Group Network Formation Game, which is a general framework where strategic agents with extremely varied connectivity requirements are building a network together. In Section 4, we introduce the Network Cutting Game which considers a dual approach, i.e., players are trying cut some nodes from certain parts of a network rather than connecting them.

1.1 Basic Concepts of Game Theory

A game is formally defined as a four-tuple $\Gamma = (N, S, M, U)$ where

- $N = \{1, 2, \ldots, n\}$ is a set of decision-making agents, called players;
- $S = \prod_{i=1}^{n} S_i$ is the set of strategy profiles where S_i is the set of decisions (strategies) player i can make (apply);

- $M : S \to O$ is a mapping from the set of strategy profiles to the possible outcomes of the game, which is denoted by O;
- $U = \prod_{i=1}^{n} U_i$ where $U_i : O \to \Re$ is a function deciding the utility or payoff of player i for each possible outcome.

In a game, we assume that each player chooses her strategy with the aim of maximizing her utility (or minimizing her payoff) which is totally characterized by U_i. Determining the output of a given game is often referred as the "solution of the game" and we next describe some popular solution concepts used in the literature.

1.1.1 Solution Concepts

For a strategy vector $s \in S$, we use s_i to denote the strategy played by player i and s_{-i} to denote the $(n-1)$-dimensional vector of the strategies played by the other players. We use the notation $U_i(s)$ to denote the utility incurred by player i for the output of the game that correspond to the strategy profile s. We will also use the notation $U_i(s_i, s_{-i})$ when it is more convenient.

Dominant Strategy Solution A strategy s_i is called a *dominant strategy* for player i if s_i is as good as (or better than) any other strategy of player i, independent of the strategies played by other players. We refer a strategy profile s as a *dominant strategy solution* if the strategy s_i of each player i is a dominant strategy. Formally, a strategy vector $s \in S$ is a dominant strategy solution, if for each player i, and each alternate strategy $s' \in S$, we have that $U_i\left(s_i, s'_{-i}\right) \geq U_i\left(s'_i, s'_{-i}\right)$. Note that if a rational agent has a dominant strategy s_i, she will certainly play s_i since it is the best strategy to play no matter what the other players do and therefore, the output of the game selected by the players will be a dominant strategy solution if each player has a dominant strategy. Unfortunately, pretty much none of the interesting games have a dominant strategy solution and therefore, we need a broader solution concept.

Nash Equilibrium We call a strategy vector $s \in S$ a *Nash equilibrium* if for all players i and each alternate strategy $s'_i \in S_i$, we have that $U_i(s_i, s_{-i}) \geq U_i(s'_i, s_{-i})$.

In other words, a Nash equilibrium is a solution where the strategy s_i of each player i is the best response of her to the strategies s_{-i} of other players. Note that this is a self-enforcing solution concept, i.e., once the players are playing such a solution, no player has any incentive to deviate. Observe that Nash equilibrium solution concept is broader than the dominant strategy solution concept since every dominant strategy solution is also a Nash equilibrium.

Unfortunately, Nash equilibrium may not exist for even very simple games. For example, there is no Nash equilibria for the traditional Paper-Stone-Scissors game where paper beats stone, stone beats scissors and scissors beat paper. None of the strategies in this game is a good strategy since for each strategy s_i of a player i, the other player has a strategy that beats s_i. The best thing a player can do in such a game is to use a random number generator and play each of her strategies with a probability of 1/3 and this observation inspires for defining a broader solution concept.

Mixed Strategy Nash Equilibrium We have given an example where there is no Nash equilibrium and the best thing a player can do is randomization over her strategies. In order to allow the players to randomize over their strategies in our framework, let us extend the choices of players so each one of them can now pick a probability distribution over her strategy set rather than a specific strategy. We call such a choice a *mixed strategy.* A mixed strategy Nash equilibrium is a solution where the probability distribution p_i selected by each player i over her strategy space S_i is the best response of her to the mixed strategies p_{-i} of other players. Observe that mixed strategy Nash equilibrium solution concept is broader than the Nash equilibrium solution concept since every Nash equilibrium solution is also a mixed strategy Nash equilibrium.

Strong Equilibrium The solution concepts mentioned so far excludes the possibility of some subset of players forming a coalition and deviating together. Strong equilibrium solution concept [10] addresses the problem of ensuring stability against deviations by subsets of players. A strategy profile s is a *strong equilibrium* if no subset of players can deviate from the solution in a way that the cost of all the players in the coalition declines.

Since providing a complete survey on the solution concepts in the literature is beyond the scope of this thesis, we would like to refer to [51] for a more elaborate discussion on other important solution concepts like correlated equilibrium, Bayesian-Nash equilibrium, and many others.

1.1.2 Existence of Equilibria

We have already defined a set of solution concepts and now we will discuss some necessary or sufficient conditions for games to have solutions. Actually, one of the most fascinating results of game theory is the following theorem proven by John Nash [49].

Theorem 1 *Any game with a finite set of players and finite set of strategies has a mixed strategy Nash equilibrium.*

Having mixed strategies extends the choices of the players and in the meantime ensures the existence of equilibrium solutions; however, mixed strategies do not really make much sense in settings such as network formation games (these will be discussed shortly). It is necessary to study the conditions for games to have Nash equilibrium. We next present a class of games guaranteed to have a Nash equilibrium. These games are known as potential games.

Potential Games Let $\Phi : S \rightarrow \Re$ be a function that maps every strategy profile s of a *finite* game Γ to a real number. Φ is called an *exact potential function* if for any strategy s_i' of a player i and any strategy vector (s_i, s_{-i}), we have $\Phi(s_i, s_{-i}) - \Phi(s_i', s_{-i}) = U_i(s_i', s_{-i}) - U_i(s_i, s_{-i})$. In order words, for any strategy profile $s \in S$, if only one of the players (without loss of generality, player i) change her strategy from s_i to some arbitrary strategy s_i' of her, the value of the exact potential function decreases as much as the increase in the utility of the deviating player. A finite game Γ is called a potential game if there exists an exact potential function defined over its set of strategy profiles. Potential games are particularly important because every potential game has at least one Nash equilibrium [48].

Congestion Games Congestion game is an important class of games studied in the game theory literature. In a congestion game, there is a set of players N and a set of resources R and each player $i \in N$ selects a subset R_i of R as her strategy. In a congestion game, the cost of each player i only depends on the resource set R_i she selected and the number of players selected the same resource. The nice mathematical properties given above for potential games also apply to congestion games since any congestion game is a potential game [52] and for any given potential game, there exists a corresponding congestion game with the same potential function [48].

1.2 Previous Work on Network Games

An elaborate discussion on network games is beyond the scope of this thesis and we will focus on network formation, routing and cut games in the rest of the section. We refer the interested reader to [51] for more detailed discussion.

1.2.1 Network Formation Games

An important class of network games is the *network formation (creation or design) games*. In network formation games, a set of players are trying to form a subnetwork on a graph satisfying some connectivity requirements. Each player has personal connectivity requirements and acts selfishly, i.e., interested only in her own connection requirements. Each edge has a cost to be paid if it is to be used as part of the subnetwork. The players want to pay as little as possible while strictly satisfying their connectivity requirements.

It has been known for long that the Nash equilibria of a game may not be unique and may not correspond to the socially optimal solutions, i.e., the solutions that maximize the total welfare (or minimize the total cost). One natural question to ask is that "What is the loss in social welfare due to selfish acts of the agents?" The two commonly used metrics to quantify this loss are *price of anarchy* [44], and the *price of stability* [5]. Price of anarchy refers to the ratio of the social welfare of the socially worst stable point to the social welfare of the socially optimal solution, which might not be stable. Price of anarchy therefore, tries to answer the question of 'how bad a stable point can be?' Price of stability tries to answer the question of 'how good a stable point can be?' by referring to the ratio of the social welfare of the socially best stable point to the socially optimal solution. Though in general price of stability is an important metric, which is analogues to the best case analysis in classical theory of algorithms, it is particularly useful if there is a mechanism which allows the agents to communicate or if it is possible to propose (but not dictate) a solution to the agents. A natural question that one would like to ask is 'how bad (or good) an average stable point is?' There is no metric used in the literature so far to address this question since it is generally pretty hard to define the 'average stable point'.

Unquestionably one of the most important decisions when modeling network design involving strategic agents is to determine how the total cost of the solution (usually the cost of the edges) is going to be split among the players. Among various alternatives [16], the "fair sharing" mechanism is the most relevant to ours [5, 14, 15, 25]. In this cost sharing mechanism, the cost of each edge of the network is shared equally by the players using that edge. This model has received much attention, mostly because of the following three reasons: Firstly, it nicely quantifies what people mean by "fair" and has an excellent economic motivation since it is strongly related to the concept of Shapley value. Secondly, fair sharing naturally models the congestion effects of network routing games, and so network design games with fair sharing fall into the well-studied class of "congestion games" [14, 17, 23, 36, 53]. Thirdly, this model has many attractive mathematical properties including guarantees on the existence of Nash equilibrium [5].

Despite all of the advantages of congestion games mentioned above, there are extremely important disadvantages as well. Firstly, although congestion games are guaranteed to have Nash equilibria, these equilibria may be very expensive. Anshelevich et al. [5] showed that the cheapest Nash equilibrium solution can be $O(\log n)$ times more expensive than the socially optimal network, and that this bound is tight. As we prove in this paper, arbitrary cost-sharing will often guarantee the existence of Nash equilibria that are as cheap as the optimal solution. Secondly, fair sharing inherently assumes the existence of a central authority that regulates the agent interactions or determines the cost shares of the agents, which may not be realistic in many network design scenarios. Arbitrary cost sharing allows the agents to pick their own cost shares, without any requirements by the central authority. Thirdly, although the players are trying to minimize their payments in fair cost sharing, they are not permitted to adjust their payments freely, i.e., a player cannot directly specify her payments on each edge, but is rather asked to specify which edges she wants to use. In the network design contexts that we consider here, we prove that giving players more freedom can often result in better outcomes. Over the last few years, there have been several new papers using arbitrary cost-sharing, e.g., [21, 34, 35]. Recently, Hoefer [33] proved some interesting results for a generalization of the game in [6], and considered arbitrary sharing in variants of Facility Location.

Fabrikant et al. [22] (see also [1]) studied the price of anarchy of a very different network design game, and [8] 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. The research on non-cooperative network design and formation games is too much to survey here, see [37, 40, 43, 51, 53, 57] and the references therein for a more comprehensive list.

1.2.2 Routing Games

The selfish routing problem is similar to the routing problem in the classical theory of networking. Given a graph, there are traffic demands among pairs of nodes and the edges have latency functions. In routing problem, one usually tries to design protocols to reduce the average or maximum latency in the network. In the selfish version of the problem, there is no authority to dictate a routing scheme, instead each packet traversing the network correspond to a player and each player chooses its own path to the destination selfishly, i.e., through the path that has minimum latency. Equilibria of routing games is first studied in the context of transportation engineering [58]. In the selfish routing game, in general, the average latency faced in the network can be arbitrarily larger than the centralized solution which we demonstrate below by a version of Pigou's example [50].

Pigou's Example Consider a simple network of two nodes, s and t, and two disjoint edges between them. The latency function for one of the edges is $c_1(x) = 1$ while for the other $c_2(x) = x^p$, where x denotes the amount of traffic on the edge and p is an arbitrary constant. In a network of selfish agents, there is a unique Nash equilibrium, all the traffic will follow the second edge and everyone will have a cost of 1. Therefore, average latency of the system will be 1. However, if there was a centralized control, an ϵ amount of the traffic would be pushed to the first edge to reduce the cost. Therefore, the average cost would be $\epsilon + (1-\epsilon)^p$. Observe that this quantity gets smaller while ϵ gets smaller or p gets larger. In the limiting case, when $\epsilon \to 0$ and $p \to \infty$, average latency tends to 0. Since the equilibrium is unique in this game, price of anarchy and the price of stability of this game are same, they are both unbounded.

Though in the variant of the Pigou's example, price of anarchy (and stability) was too discouraging, this was due to highly nonlinearity of the latency function in the second edge. In every network with affine latency functions, i.e., the latency functions can be expressed in the form $ax + b$, regardless of how large and complex the network is, price of anarchy is at most $4/3$.

Pigou's example represents the worst case due to the following theorem. First results on the price of anarchy of selfish routing was proven in [55] and later [19] gave a very elegant geometric proof for Theorem 2.

Theorem 2 *A unique Nash equilibrium always exists. Furthermore, the price of anarchy is* $4/3$ *for affine latency functions and* $\frac{p}{logp}$ *for polynomial latency functions and these bounds are tight.*

Another interesting result on routing games is that addition of new links to a network can paradoxically increase the average latency of the traffic. This phenomenon is called as the Braess's paradox [12] and see [54] for the literature on it. For a more elaborate survey of the results in routing games, see [51].

1.2.3 Cut Games

An extensively studied game related to cuts is the max-cut game [18, 23, 32]. In this game players are forming a bipartition on the graph, where the utility of each player is the total weight of the edges of the cut incident to her. The Max-cut game always admits Nash equilibria since it is a potential game [48] and it is recently shown that it always admits strong Nash equilibria [28].

There have been many interesting applications of game theory to network security models (eg. [29, 47, 56]). A notion of *interdependent security* (IDS) games was introduced by Kunreuther and Heal [45]. In these games, the decision to adopt a security measure by a player affects other players in the network. An algorithm for finding an approximate equilibrium on this model was later given by Kearns and Ortiz [41]. Work on a similar model by Aspnes et al [9] deals with players immunizing their nodes against infections that can spread in the network. There are several major differences between these models and the games we consider: (i) players in IDS games can only immunize themselves, while in our games players are allowed to add security to different parts of the network, and (ii) these models consider that an attack can occur randomly anywhere in the network, while in our games the players are trying to protect themselves from specific areas of the network which might be different for different players.

Finally, some of our questions are related to ones studied by Engelberg et al. [20], who look at budgeted versions of cut problems like Multiway Cut, Multicut, and k-cut. This work considers centrally optimal solutions, not equilibria, and while our values β_i can be thought of as budgets, they are budgets for individual players, not global budgets on the cost of the network.

1.3 Our Contributions

1.3.1 Global Connection Game

In Chapter 2, we consider the price of stability of several important extensions of the *(Global) Connection Game*, which was first defined in [6], and later studied in a variety of papers including [15, 21, 25, 34, 35]. This game represents a general framework where a network is being built by many different agents/players who have different connectivity requirements, but 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. While the Connection Game is not a congestion game, and is not guaranteed to have a Nash equilibrium, it actually behaves much better than [5] 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 [5] is $\frac{\log n}{\log \log n}$ [2]. Moreover, all such models (including cost-sharing models described above) 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 [6] 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. While the survivable network design game that we consider can be expressed as part of the framework in [33], these results do not imply ours, and our results cannot be obtained using their techniques.

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 [38].

In this paper, we consider the *Survivable Connection Game,* where each agent (or, 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.

We also consider a more restrictive version of the Survivable Connection Game. This second game is identical to the one described above, except that a player is only allowed to deviate by changing the payments on one of her paths, instead of both of them at once. This can model the case 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 extremely possible when a player represents a large company.

Our Results We consider the price of stability for both versions of the Survivable Connection Game mentioned above. 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, [6] 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 1.1 shows a game where this is the case. In Figure 1.1, 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[1] Nash equilibrium for any $\alpha < 2$.

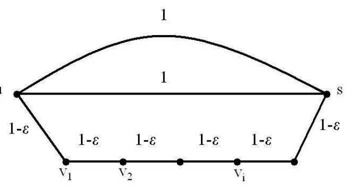

Figure 1.1: An instance with no pure Nash equilibrium.

The example in Figure 1.1 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 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.

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

- For the Survivable Connection Game where each player is only allowed to deviate by changing the payments on a single path, there exists a stable solution that is as good as the centralized optimal solution. In other words, the price of stability is 1.

- 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 [6], 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 form these techniques, we prove strong results about the laminar structure of survivable networks. These results are of independent interest, and give useful techniques for dealing with survivable networks in both game theoretic and more traditional contexts. Some of the results presented in Chapter 2 have appeared in [4].

1.3.2 Group Network Formation Game

In Chapter 3, we study a network design game that we call the *Group Network Formation Game*, which captures the essence of strategic agents building a network together in many scenarios. In this game players correspond to nodes of a graph (although not all nodes need to correspond to players), and the players can have extremely varied connectivity requirements. For example, there might be several different "types" of nodes in the graph, and a player desires to connect to at least one of every type (the centralized version of this would correspond to Group Steiner Tree [26]). Or instead, a player might want to connect to at least k other player nodes. The first example above is useful for many applications where a set of players attempt to form groups with "complementary" qualities. The second example corresponds to a network of servers where each server want to be connected to at least k other servers so that it can have a backup of its data; or in the context of IP networks, a set of ISPs that want to increase the reliability of the Internet connection for their customers, and so decide to form multi-homing connections through k other ISPs [59]. Many other types of connectivity requirements fit into our framework, and so the results we give in this paper will be relevant to many different types of network problems.

In our game, all players want to be a part of a meaningful group (which we interchangeably call a "happy group") which can correspond to many connectivity requirements, some of which are mentioned above. The optimal centralized solution (which we denote by OPT) for this game is the cheapest possible network where every connected component is a happy group, since this is the solution maximizing social welfare. For our first example above, OPT corresponds to the cheapest Group Steiner Tree, for the second to the Terminal Backup problem [7], and in general it can correspond to a variety of constrained forest problems [27]. Our goals include understanding the quality of exact and approximate Nash equilibria by comparing them to OPT, and thereby understanding the efficiency gap that results because of the players' 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

In the Group Formation Game, we don't assume the existence of a central authority that designs and maintains the network, and decides on appropriate cost-shares for each player. Instead we use a cost-sharing scheme which is sometimes referred to as "arbitrary cost sharing" [6, 21] that permits the players to specify the actual amount of payment for each edge. This cost-sharing mechanism is necessary in scenarios where very little control over the players is available, and gives more freedom to players in specifying their strategies, i.e., has a much larger strategy space. The main advantage of such a model is that the players have more freedom in their choices, and less control is required over them. A disadvantage of such a system, however, is that it does not guarantee the existence of Nash equilibria (unlike more constrained systems such as fair sharing [5]). Studying the existence of Nash equilibria under arbitrary cost sharing has been an interesting research problem and researchers have proven existence for many important games [6, 21, 33, 34]. Interestingly, in many of these problems it has been shown that the equilibrium is indeed cheap, i.e., costs as much as the socially optimal network. As we show in this paper, this tells us that in the network design contexts we consider, *arbitrary sharing produces more efficient outcomes while giving the players more freedom.*

Our Results Our main results are about the existence and computation of cheap approximate equilibria. By an α-approximate Nash equilibrium, we mean that no player in such a solution has a deviation that will improve her cost by a factor of more than α. While our techniques are inspired by [6], our problem and connectivity requirements are much more general, and so require the development of much more general arguments and payment schemes.

- In Section 3.2, we show that in the case where all nodes are player nodes, there exists a Nash equilibrium as good as OPT, i.e., the price of stability is 1.
- In Section 3.3, we show that in the general case where some nodes may not be player nodes, there exists a 2-approximate Nash equilibrium as good as OPT.
- We show that if every player is replaced by two players (or if every player node has at least two players associated with it), then the price of stability is 1. This is in the spirit of similar results from selfish routing [5, 53], where increasing the total amount of players reduces the price of anarchy.
- Starting with a β-approximation to OPT, we provide poly-time algorithms for computing an $(1+\epsilon)$-approximate equilibrium with cost no more than β times OPT, for the case where all nodes are player nodes. The same holds for the general case with the factor being $(3.1+\epsilon)$ instead.

Since for monotone happiness functions F, OPT corresponds to a constrained forest problem [27], then the last result gives us a poly-time algorithm with $\beta = 2$. Notice that we assumed that the function F is monotone, i.e., addition of more terminals to a component does not hurt. This assumption is necessary, since as we prove in Section 3.5, if F is not monotone there may not exist *any* approximate Nash equilibria. We also show that the results above are only possible in our model with arbitrary cost-sharing, and not with fair sharing.

Because of its applications to multi-homing [7, 59], we are especially interested in the behavior of Terminal Backup connectivity requirements, i.e., when a player node desires to connect to at least k other player nodes. For this special case, we prove a variety of results, such as price of anarchy bounds and the improvement of fair sharing results from [5] for this new problem. The lower bounds for Terminal Backup also hold for the general *Group Network Formation Game*, showing that while the price of stability may be low, the price of anarchy can be as high as the number of players. Some of the results presented in Chapter 3 have appeared in [3].

1.3.3 Strategic Cut Games

Networked systems for transport, communication, and social interaction have contributed to all aspects of life by increasing economic and social efficiency. However, increased connectivity also gives intruders and attackers better opportunities to maliciously spread in the network, whether the spread is of disinformation, or of contamination in the water supply [46]. Anyone participating in a networked system may therefore desire to undertake appropriate security measures in order to protect themselves from such malicious influences.

In Chapter 4 we introduce a *Network Cutting Game*, which is a game-theoretic framework where a group of self-interested players protect vertices that they own by cutting them off from parts of the network that they find untrustworthy. Cutting an edge should not be interpreted as destroying a part of the network: instead it can correspond to taking security measures on that edge such as placing sentries on lines of communication. These notions are applicable in areas such as airline security. Consider a situation where country A requires extra security screening of passengers or cargo from country B. Due to the networked multi-hop structure of international air travel, the optimal locations for carrying out such screenings may lie somewhere in between the two countries. In general, the goal of each player is to make sure that nothing can get to it from an “untrustworthy” part of the network without passing through an edge with installed security measures.

The purpose of each player is to protect themselves while spending as little money as possible for their security. We investigate the efficiency of the security actions taken by a group of agents by studying the *price of anarchy* and the *price of stability.* While the price of anarchy can be extremely high (see Section 4.5), we prove nice bounds on the price of stability.

Our Results Our main results are about the existence and computation of cheap, exact, and approximate equilibria for several important special cases of the *Network Cutting Game*. By an α-approximate Nash equilibrium, we mean that no player in such a solution can reduce her cost by a factor of more than α by deviating: Specifically, we consider the following special cases of cut requirements.

In the *Single-Source Network Cutting Game,* each player i wants to disconnect her node from a common node t, i.e., $T_i = \{t\}$ for all players i.

In the *Network Multiway Cut Game,* each player i wants to disconnect her node from the nodes of all other players, i.e., $T_i = P\backslash\{i\}$ for all players i. Notice that when β_i for all players is large enough, the socially optimal solution is exactly the *Minimum Multiway Cut.*

In the *Network Multi-Cut Game,* each player i wants to disconnect her node i from some specific node t_i. In other words, this is the case where $|T_i| = 1$ for all players. When β_i for all players is large enough, the socially optimal solution is exactly the *Minimum Multicut* for the set of pairs (i, t_i).

Our main results are as follows:

- In Section 4.2, we study the *Single Source Network Cutting Game* and show that there always exists a Nash equilibrium as cheap as OPT, i.e., the price of stability is 1. Furthermore, that Nash equilibrium can be computed in polynomial time. This analysis easily generalizes to the case when T_i are not singleton sets, but $T_i = T_j$ for all i, j.

- In Section 4.3, we study the *Network Multiway Cut Game* and show that there always exists a Nash equilibrium as good as OPT, i.e., the price of stability is 1. We also show that a Nash equilibrium whose cost is within a factor of 3/2 of OPT can be computed in polynomial time.

- In Section 4.4, we study the *Network Multi-Cut Game* and show that there always exists a 2-approximate Nash equilibrium as cheap as OPT.

In Section 4.5, we consider the above games on graphs with non-uniform edge costs, i.e., where each edge has some fixed cost $w(e)$ to cut it. If every edge must be bought entirely by a single player, we show that there are simple instances for all these games where all Nash equilibria are expensive. Because of this, we allow players to share the cost of cutting edges. If the players are allowed to pay for cutting only a portion of an edge, we prove that all the above results extend to non-uniform edge costs.

CHAPTER 2
SURVIVABLE NETWORK DESIGN

2.1 The Model and Preliminaries

We now formally define the Survivable Connection Game[2] 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 $p = (p_1, \ldots, p_N)$. In the Survivable Connection Game, each player i wants to have 2 edge-disjoint paths between i and s, i.e., each player i requires that $M(G_p, i, s) \geq 2$, where $M(G_p, i, s)$ denotes the min-cut between the nodes i and s on the graph G_p. While required to connect her terminals using at least 2 edge-disjoint connections, each player also tries to minimize her total payments, $\sum_{e \in E} p_i(e)$ (which we will denote by $|p_i|$). We conclude the definition of our game by defining the cost function for each player i as:

- $cost(i) = \infty$ if $M(G_p, i, s) < 2$
- $cost(i) = |p_i|$ otherwise.

The rest of this chapter is mainly devoted to proving that there exist exact or approximate Nash equilibrium solutions that cost as much as OPT. To prove our results we give algorithms that either form Nash equilibrium strategies for the players that buy OPT, or give us a solution G_p that is cheaper than OPT and

[2]Portions of this chapter previously appeared as: E. Anshelevich and B. Caskurlu. Price of Stability in Survivable Network Design. In *Proc. 2nd International Symposium on Algorithmic Game Theory (SAGT 2009), pg. 208-219.*

satisfies the connectivity requirements of all the players, i.e., $M(G_p, i, s) \geq 2 \forall i$, which contradicts with OPT being the socially optimal solution. We make use of the following observations in our arguments.

OPT and Nash Equilibrium OPT, by definition, is the cheapest network that satisfies the connection requirements of all the players. Therefore, for every edge e of OPT, there is a set of players whose connection requirements will be dissatisfied if e is deleted from OPT, since otherwise a cheaper network $OPT - e$ that satisfies the connectivity requirements of all the players can be obtained by simply deleting e from OPT. Note that in a Nash equilibrium, 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 nonzero payment for e. The players in this set are therefore said to *witness* e, since without them, e would not be needed.

Witness Sets Let i be a player that witnesses e, i.e., i will have only 1 path to s if e is deleted. Observe that $M(OPT, i, s) = 2$ and $M(OPT - e, i, s) = 1$. Therefore, there is a cut in OPT between i and s of size 2 such that e is 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 1.1(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.*

Smallest Witness Sets 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 edge-disjoint paths. A player witnessing $e = (i, j)$ may be using e either in the direction $i \rightarrow j$ or in the direction $j \rightarrow i$. 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. If e is witnessed in a direction, say $i \to j$, then the smallest witness set of e in that direction, $W_i(i, j)$ exists and it is unique as proven by Lemma 1, the proof of which appears in Section 2.4.1. Observe that every edge $e = (i, j)$ of OPT has either 1 or 2 smallest witness sets. In the Nash equilibrium solutions formed by our algorithms, the cost of each edge e of OPT will be shared among the players that are in a smallest witness set of e. In the rest of the paper, we will use W to denote the set of all smallest witness sets of all the edges of OPT.

Lemma 1 *If an edge $e = (i, j)$ is witnessed in the direction say $i \to j$, then the smallest witness set of e in that direction, $W_i(i, j)$, is unique.*

Laminarity A set of sets W is called a *laminar* set system if for any 2 elements of the set system $W_1, W_2 \in W$ either W_1 and W_2 are disjoint or one of them is a subset of the other. 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 example, 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. Theorem 3, proof of which appears in Section 2.4 proves that this is essentially the only way that smallest witness sets may have a non-trivial intersection, and contracting paths composed of degree 2 non-player nodes makes the set of smallest witness sets become laminar. We say that the set of smallest witness sets W of a solution G_S is *laminar with path exceptions* if the set of smallest witness sets of G_S is laminar after contracting paths composed of degree 2 non-player nodes.

Theorem 3 *Let G^* be a graph obtained by replacing each path P of OPT composed of degree 2 non-player nodes by a single edge. The set of smallest witness sets of*

G^* is a laminar set system, i.e., the set of smallest witness sets of OPT is laminar with path exceptions.

2.2 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 2 *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 4 *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 $i \rightarrow j$ or $i \rightarrow k$ to connect to s. Then, t is still feasible in OPT' by Lemma 2, 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 2, 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 3 *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 2, 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 2 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.

■

2.3 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 5.

Theorem 5 *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 2.1(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 2.1(A) as her connection paths. Please note that, though 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

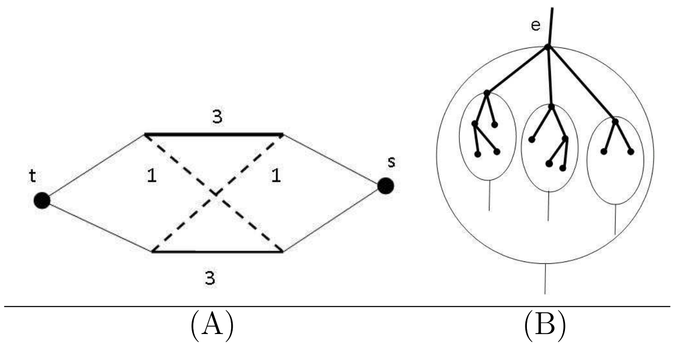

Figure 2.1: (A) An example illustrating our stable solution concept. (B) Result of the Tree Generation algorithm. The ovals represent smallest witness sets.

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 5, 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.

2.3.1 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

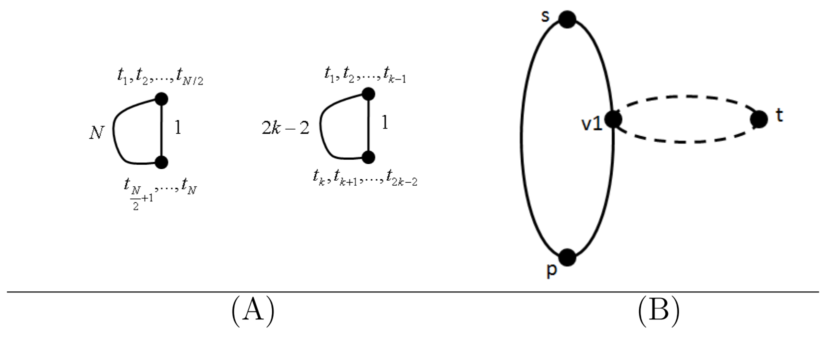
$t_1, t_2, \ldots, t_{N/2}$
N
1
$t_{\frac{N}{2}+1}, \ldots, t_N$
$t_1, t_2, \ldots, t_{k-1}$
$2k-2$
1
$t_k, t_{k+1}, \ldots, t_{2k-2}$
s
v1
t
p
(A)
(B)

Figure 2.2: (A) An instance where Price of Anarchy is $2N$. The terminal nodes are the nodes on the circle. (B) The solid paths are the connection paths of player p. The dashed lines are the connection paths of player t, which intersect the connection cycle of p at a point v_1.

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 2.2(A) 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 2.2(A), 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 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 2.2(A) gives an example where a stable solution is a 2-approximate Nash equilibrium, and no less, showing that the bound in Theorem 5 is tight.

2.3.2 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 6 *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 on the socially optimal solution. 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.

Node Splitting 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 2.

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 2. 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 as depicted in Figure 2.2(B). 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 2. 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 6, 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.

Let W be the set of all smallest witness sets of the socially optimal solution OPT, i.e., $W = \{W_i(e) | \text{for some } i, e\}$. For ease of explanation, we will first consider the case where W is a *laminar* set system, i.e., for any 2 elements of W_1 and W_2 of W, either W_1 and W_2 are disjoint or one of them is a subset of the other. We will first prove our results under the assumption that W is a laminar set system. Theorem 8 describes how our algorithm can be modified for the case W is *laminar with path exceptions* and we conclude the proof of our main result by proving Theorem 3 in Section 2.4.

Our payment scheme is formed by Algorithm 2. 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 2, 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

```
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 $p_i = \sum_{f \in (E \setminus E_i)} p_i(f)$;
        Define $p(e) = \sum_j p_j(e)$;
        Define $c'_i(f)$ to be the modified cost of $f$ for $i$;
        Define $\chi_i$ as the cost of the cheapest deviation by player $i$ in $G - e$ under $c'_i$;
        Set $p_i(e) = \min\{\chi_i - p_i, c(e) - p(e)\}$.
```

Algorithm 1: Algorithm That Generates the Payment Scheme

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.

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 2.1(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 4 *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 5 *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 2.3, 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 6, 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 6 *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 $T_j(e)$.*

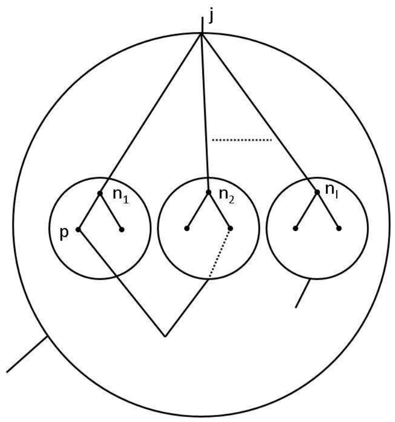

Figure 2.3: Structure of Other Paths

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 $W_{n_1}(j, n_1), \ldots, W_{n_l}(j, n_l)$ be their respective smallest witness sets. This is illustrated in Figure 2.3, 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 2.3. 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 theorem for the special case W is laminar. We prove the same result for the general case(when W is laminar with path exceptions) in Theorem 8.

Theorem 7 *If W is laminar Algorithm 2 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 7 (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 f in this tree, then she is witnessing all the edges between e and f since the tree generation algorithm is recursive. 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 6, 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 2.

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 6, so we can use Lemma 2 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 $D^i = \{d^i_1, d^i_2, \ldots, d^i_n\}$ and $D^j = \{d^j_1, d^j_2, \ldots, d^j_m\}$ of $T_i(i, j)$ and $T_j(i, j)$ respectively. Let $t^i_1, t^i_2, \ldots, t^i_n$ and $t^j_1, t^j_2, \ldots, t^j_m$ 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 $t^i_1, t^i_2, \ldots, t^i_n$ do not use any edge of $T_j(i,j)$ above the set of points D^j and also assume alternate connection cycles of $t^j_1, t^j_2, \ldots, t^j_m$ 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 2 by connecting to 2 points of the alternate connection cycle of one of $t^i_1, t^i_2, \ldots, t^i_n$ 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 $t^j_1, t^j_2, \ldots, t^j_n$ 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^i_1. Now, to construct OPT' let the players $t^i_1, t^i_2, \ldots, t^i_n$ take their alternate connection cycles similar to the previous cases, but have $t^j_1, t^j_2, \ldots, t^j_m$ 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 2 again. All we need to show is to identify 2 edge-disjoint paths to the alternate connection cycle of t^i_1. 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^i_1, and by Lemma 6 it is necessarily edge-disjoint from the other connection path, which connects to s. ■

Lemma 7 *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 2 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

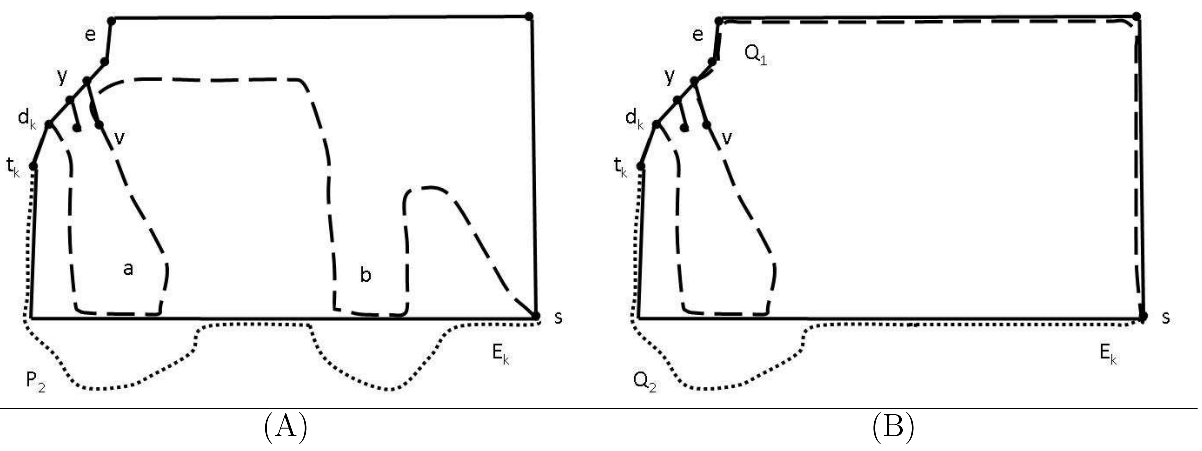

Figure 2.4: (A) This figure shows the general alternate path structure of the deviation χ_k. Notice that, despite the way they appear in the figure, in general the paths P_1 and P_2 may not be using the edges of E_k in order. (B) Shows the construction of the deviating paths Q_1 and Q_2.

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.

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 2.4(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 2.4(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 8 *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 2 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 2 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 2.4(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 6). 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 proven that the price of stability is 1 under the assumption that W is laminar. We next describe how we can modify our algorithm for the case W is not laminar.

Theorem 8 *There is a stable solution as good as OPT for the Survivable Connection Game.*

Proof. Theorem 7 proves that there is a stable solution as good as OPT for the Survivable Connection Game under the assumption that W is laminar. Now, we describe how we can modify our payment algorithm to obtain a stable solution as cheap as OPT even if W is laminar with path exceptions.

Let u and v be 2 player or high degree non-player nodes such that there is a path P on OPT between u and v, of which all intermediate nodes are degree 2 non-player nodes. Observe that if a player t witnesses any edge e of P, then it witnesses all of the edges of P in the same direction since any path between t and s that uses e has to use all the edges of P. Therefore, if an edge e of P is witnessed in both directions then all of the edges of P are witnessed in both directions. Similarly, if e is witnessed only in the direction $u \to v$ then all the edges of P are witnessed in the direction $u \to v$.

In proving Theorem 7, for each edge $e = (i, j)$ of OPT, we first generated the trees $T_i(e)$ and/or $T_j(e)$ in $W_i(e)$ and/or $W_j(e)$ such that all leaf-nodes of the trees are player nodes and all non-leaf nodes are non-player nodes. Once the payment algorithm have paid for the cost of all the edges in $W_i(e)$ and/or $W_j(e)$, the algorithm asked only the players at the leaves of $W_i(e)$ and/or $W_j(e)$ to pay for the cost of e.

For the case when W is not laminar, we will modify the generation of the trees as follows. First contract each path P of OPT composed of degree 2 non-player nodes into a single edge f. Let us call G^* to this new graph. For each edge $f = (u, v)$ of G^*(which corresponds to an edge or a path of OPT), generate the trees $T_u(f)$ and/or $T_v(f)$ in $W_u(f)$ and/or $W_v(f)$ such that all leaf-nodes of the trees are player nodes and all non-leaf nodes are non-player nodes as explained above. Observe that Lemma 5 and Lemma 6 trivially holds on G^* since the set of smallest witness sets of the edges of G^* is laminar. For each edge $f = (u, v)$ of G^* that corresponds to a path of OPT, uncontract f back with the actual path P of OPT. Note that we haven't generated the trees for actual edges of OPT yet. Let $e = (i, j)$ be an arbitrary edge of P and without loss of generality let i be the node closer to u. If the tree generation algorithm generated a tree $T_u(f)$ for f on G^* then the tree

$T_i(e)$ is obtained by appending the portion of P between u and i to the tree $T_u(f)$. Note that $T_i(e)$ is actually a tree since it is obtained by appending a path outside the smallest witness set $W_u(f)$ on G^* with a tree contained in $W_u(f)$. We obtain $T_j(e)$ similarly if the tree generation algorithm have generated $T_v(f)$ for f on G^*. Observe that the smallest witness set of $W_i(e)$ on OPT is composed of the smallest witness set $W_u(f)$ on G^* and the set of nodes of P between u and i. Therefore, the set of terminals in $W_i(e)$ on OPT are exactly the same set of terminals in $W_u(f)$ on G^*. Since the other connection path of any player in $W_u(f)$ on G^* does not use any edge of $T_u(f)$ or f, the other connection path of any player in $W_i(e)$ on OPT does not use any edge of $T_i(e)$.

To pay for the cost of the edges of OPT, we pick an arbitrary edge $e = (i, j)$ whose payment scheme has not been decided yet. If e is not an edge of a path P composed of degree 2 nonterminal nodes, we follow exactly the same steps as in Algorithm 2. Therefore, we consider the case where e is an edge of a path P on OPT between u and v, of which all intermediate nodes are degree 2 non-player nodes. Assume e is witnessed in one direction, without loss of generality in the direction $i \to j$, where i is the adjacent node to e that closer to u on path P. Then we decide the payment of all the edges of P one after another starting from the edge closest to u till the edge incident on v. Observe that all of those edges has only one tree and the trees of all these edges contain the same set of players. If the algorithm can pay for the cost of all the edges of P then it proceeds with the next edge. Otherwise, we can construct a network cheaper than OPT with exactly the same arguments in the proof of Theorem 7. If e is witnessed in both directions, then we ask the players of $T_i(e)$ to pay for the edges of P one after another starting from the edge incident to u and we ask the players of $T_j(e)$ to pay for the edges of P one after another starting from the edge incident to v. If these payments "meet in the middle" to cover the cost of all edges in P, then we have paid for e and the algorithm will proceed with selecting another unpaid edge, and if they do not, then we can construct a network cheaper than OPT with exactly the same arguments in the proof of Theorem 7. Observe that the payment scheme we generate actually has a value $p_i(e)$ for every edge of OPT, not the contracted edges of G^*.

■

2.3.3 Polynomial-Time Results

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 9 *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 $c\left(G'\right) \leq c\left(G_\alpha\right)$.*

Proof. The proof of Theorem 8 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. The algorithms algorithms and the proof of the theorem are based on two properties of OPT: every edge has a witness set, and the set of smallest witness sets is laminar with path exceptions. 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 3 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 7 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 $2\left(1+\epsilon\right)$-approximate Nash equilibrium, i.e., a solution where no player can reduce its cost by more than a factor of $2\left(1+\epsilon\right)$ 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 2 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 7, 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 the payment algorithm 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 [38], 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

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

■

2.4 Smallest Witness Sets are Laminar with Path Exceptions

We now prove Theorem 3. 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.

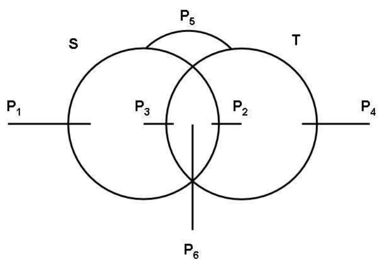

Figure 2.5: General topology of two intersecting sets

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 2.5.

Observe that;

$$p_1 + p_2 + p_5 + p_6 = 2 \tag{2.1}$$

and

$$p_3 + p_4 + p_5 + p_6 = 2 \tag{2.2}$$

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

Also;

$$p_1 + p_4 + p_6 \geq 2 \tag{2.3}$$

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 2.1 and 2.3 together imply

$$p_4 \geq p_2 + p_5 \tag{2.4}$$

where equations 2.2 and 2.3 together imply

$$p_1 \geq p_3 + p_5. \tag{2.5}$$

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 2.1 and 2.2 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 2.4 and 2.5. Therefore, the topology of these witness sets would be as described by Figure 2.6. In Figure 2.6 it is shown

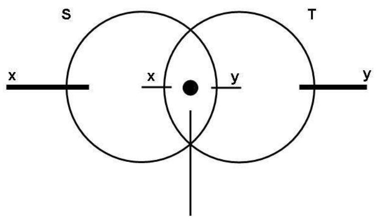

Figure 2.6: Two intersecting witness sets that have a terminal in the intersection.

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 2.6 (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, $x = y = 1$.

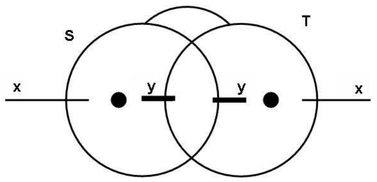

Figure 2.7: Two intersecting witness sets that have a terminal in each difference.

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 2.2 and 2.1 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 2.1 and 2.2. Observe that S cannot be the smallest witness set of any of the $p_1 + p_5$ edges between $S - T$ 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 2.7, 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 2.7, $y \geq 1$ due to feasibility of this terminal. Since we know that $x \geq y$ from equation 2.4 or 2.5, 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 2.4 or 2.5, 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 2.6, we haven't mentioned anything about the nodes in $S-T$ and $T-S$. Due to the observations in Figure 2.7, 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 2.7, 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.

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. After replacing the paths of degree 2 non-terminals with an edge the set smallest witness sets become laminar, and the desired structural properties hold, since all the nodes in the "special set" will disappear. Therefore, after replacing the paths of degree 2 non-terminals with an edge, the special set will be empty, which implies that at least one of $S-T$, $S \cap T$ or $T-S$ is empty.

2.4.1 Smallest Witness Sets are Unique

We will now prove Lemma 1, i.e., if an edge $e = (i, j)$ is witnessed in the direction say $i \rightarrow j$, then the smallest witness set of e in that direction, $W_i(i, j)$, is unique.

Let W_1 and W_2 be two smallest witness sets of an arbitrary edge $e = (i, j)$ that witness e in the same direction, say in the direction $i \rightarrow j$. As shown in the proof of Theorem 3; for any 2 smallest witness sets S and T, all $S - T, T - S$ and $T \cap S$ can be nonempty since laminarity does not hold in general, however the intersection of S and T can be in one of the two forms depicted in Figure 2.6 and Figure 2.7. However, observe that in both of these forms, S and T are smallest witness sets of different edges. Therefore, since W_1 and W_2 are smallest witness sets of the same edge, all $W_1 - W_2, W_2 - W_1$ and $W_2 \cap W_1$ cannot be nonempty, i.e., they are either disjoint or one of them is a subset of the other. Since W_1 and W_2 are 2 smallest witness sets of e in the direction $i \rightarrow j$, i is included in both W_1 and W_2(so W_1 and W_2 are not disjoint) and they contain equal number of nodes. Therefore, $W_1 = W_2$.

CHAPTER 3
GROUP NETWORK FORMATION GAME

3.1 The Model and Preliminaries

We now formally define the *Group Network Formation Game*[3] as follows. Let an undirected graph $G = (V, E)$ be given, with each edge e having a nonnegative cost $c(e)$. This graph represents the possible edges that can be built. Each player i corresponds to a single node in this graph (that we call a *player* or *terminal* node), which we will also denote by i. Similarly to [6], a strategy of player i is a payment vector p_i of size $|E|$, where $p_i(e)$ is how much player i is offering to contribute to the cost of edge e. We say that an edge e is *bought*, i.e., it is included in the network, if the sum of payments of all the players for e is at least as much as the cost of e ($\sum_i p_i(e) \geq c(e)$). Let G_p denote the subgraph of bought edges corresponding to the strategy vector $p = (p_1, \ldots, p_N)$. G_p is the outcome of this game, since it is the network which is purchased by the players.

To define the utilities/costs of the players, we must consider their connectivity requirements. *Group Network Formation Game* considers the class of problems where the players' connectivity requirements can be compactly represented with a function $F : 2^U \rightarrow \{0, 1\}$, where $U \subseteq V$ is the set of player nodes, similar to [27]. This function F has the following meaning. If S is a set of terminals, then $F(S) = 1$ if and only if the connectivity requirements of all players in S would be satisfied if S is the set of terminals of a connected component in G_p. For the example above, where each player wants to connect to at least one player from each "type", the function $F(S)$ would evaluate to 1 exactly when S contains at least one player of each type. Similarly, for the "data backup" example above, the function $F(S)$ would evaluate to 1 exactly when S contains at least $k+1$ players. In general, we will assume that the connectivity requirements of the players are represented by a monotone "happiness" function F. The monotonicity of F means that if the

[3]Portions of this chapter previously appeared as: E. Anshelevich and B. Caskurlu. Exact and Approximate Equilibria for Optimal Group Network Formation. In *Proc. 17th Annual European Symposium on Algorithms (ESA 2009), pg. 239-250.*

connectivity requirement of a player is satisfied in a graph G_p, then it is still satisfied when this player is connected to strictly more nodes. We will call a set of player nodes S a "happy" group if $F(S) = 1$. While not all connectivity requirements can be represented as such a function, it is a reasonably general class that includes the examples given above. Therefore an instance of our game consists of a graph $G = (V, E)$, player nodes $U \subseteq V$, and a function F that states the connectivity requirements of the players. We will say that player i's connectivity requirement is *satisfied* in G_p if and only if $F(S_i(G_p)) = 1$ for $S_i(G_p)$ being the terminals of i's connected component in G_p. While required to connect to a set of terminal nodes satisfying her connectivity requirement, each player also tries to minimize her total payments, $\sum_{e \in E} p_i(e)$ (which we will denote by $|p_i|$). We conclude the definition of our game by defining the cost function for each player i as:

- $cost(i) = \infty$ if $F(S_i(G_p)) = 0$
- $cost(i) = |p_i|$ otherwise.

3.1.1 Properties of the Socially Optimal Network

In this section, we will show some useful properties of the socially optimal network for the *Group Network Formation Game*, which we refer to as OPT. For notational convenience, we will extend the definition of the happiness function to subgraphs and use $F(S)$ to denote the value of the happiness function for the set of terminal nodes in a connected component S.

The cost of a network for player i in which her connectivity requirement is not satisfied is ∞. Therefore, OPT is the minimum cost network that satisfies the connectivity requirements of all the players. Furthermore, since the satisfaction of the players only depends on the terminal nodes they are connected to, then OPT is acyclic, since otherwise one can obtain a cheaper network that satisfies all the players simply by deleting any one of the edges of a cycle included in OPT.

Observation 1 *The socially optimal network for the Group Network Formation Game is the minimum cost forest that satisfies all the players.*

Let $e = (i, j)$ be an arbitrary edge of a tree T of OPT. Removal of e will divide T into 2 subtrees, namely T_i and T_j (let T_i be the tree containing node i). After removal of e, connectivity requirements of some of the players in T will be dissatisfied, i.e., either $F(T_i) = 0$ or $F(T_j) = 0$, since otherwise $OPT - e$ would be a network that is cheaper than OPT and satisfies all the players. Therefore, once e is deleted from OPT, all the players in T_i or T_j or both will be dissatisfied. The players that are dissatisfied upon removal of e are said to *witness* e. If e is witnessed by only the players in T_i or only the players in T_j then e is said to be an edge *witnessed from* 1*-side*. Analogously, we say e is *witnessed from* 2*-sides* if it is witnessed by all the players in T.

In general, some of the edges of a tree T may be witnessed from 1-side whereas some others are witnessed from 2-sides. We show that the edges of T witnessed from 2-sides form a connected component in T.

Proposition 1 *Let $e = (i, j)$ be an edge of T that is witnessed from* 1*-side, w.l.o.g. from the side of i. Then all the edges in T_i are also witnessed from* 1*-side.*

Proof. Let $f = (u, v)$ be an arbitrary edge in T_i and let v be the node closer to i in T_i. If f is removed from T then T would be divided into 2 trees, namely T_u and T_v where $T_u \subset T_i$ and $T_v \supset T_j$. Since e is witnessed only by the players in T_i, $F(T_i) = 0$ and $F(T_j) = 1$. Since F is a monotone function, $F(T_u) = 0$ and $F(T_v) = 1$ and therefore, f is witnessed from 1-side, from the side of u. ■

Corollary 1 *The edges of T witnessed from* 2*-sides form a connected component in T.*

Proof. If there is no or exactly one edge in T that is witnessed from 2-sides then the result trivially holds. Assume there exists an edge f that is witnessed from 2-sides. Let r be a node of T that is incident to f, and root T at r. Let e be an arbitrary edge in T that is witnessed from 2-sides. Observe that all the edges between e and the root r are witnessed from 2-sides, since if an edge of this path were witnessed from 1-side, then by Proposition 1 so would e. Therefore, the set of edges of T that are witnessed from 2-sides form a connected component in T that contains r. ■

3.2 When all Nodes are Terminals

For the *Group Network Formation Game*, we don't 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 terminal node, we prove that Nash equilibrium is guaranteed to exist. Specifically, there exists 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 explicitly forming the stable payments on the edges of OPT by giving a payment algorithm. The payment algorithm, which will be formally defined below, first roots the tree and then loops through all the players/nodes in reverse BFS order and decides their payments for all their incident edges. The algorithm never asks a player i to pay for the cost of an edge e that is not incident to i.

Since we are trying to form a Nash equilibrium, no player i should have an incentive of unilateral deviation from her strategy p_i when the algorithm terminates, i.e., $|p_i|$ should not be more than the cost of the best deviation of player i. Observe that a best deviation of player i, which we denote by $\chi_i\left(p_{-i}\right)$, is the cheapest strategy of player i that satisfies her connection requirement given the strategies p_{-i} of other players. While such a deviation may not be unique, for our purposes it is enough to let $\chi_i\left(p_{-i}\right)$ denote an arbitrary best response of player i to strategy p_{-i}. We will show that $p_i + p_{-i}$ buys all the edges of OPT when our payment algorithm terminates, and that those payments form a Nash equilibrium.

Notation and Invariant Let p^* denote the cheapest strategy vector that buys all the edges of OPT, i.e., $p^*(e) = c(e)$ if e is in OPT and $p^*(e) = 0$ otherwise. Let $\overline{p_i}$ denote the minimum payment to be made by other players to buy all the edges of OPT given that player i plays the strategy p_i, i.e., $\overline{p_i} = p^* - p_i$. To have an easier analysis we want our algorithm to have a stronger property: we not only want it to ensure stability at termination but also at each intermediate step. In other words, at any step of the algorithm, the inequality $|p_i| \leq |\chi_i\left(\overline{p_i}\right)|$ will hold, i.e., the payment strategy p_i assigned to i should be the cheapest strategy of i that satisfies her connectivity requirement, assuming the rest of the payments to buy all the edges of OPT are made by other players. Note that if all the edges of OPT are bought, i.e., $p_i + p_{-i} = p^*$, then $p_{-i} = \overline{p_i}$ and the invariant $|p_i| \leq |\chi_i(\overline{p_i})|$ turns into the Nash equilibrium condition. To show that our algorithm produces a Nash equilibrium as cheap as OPT, it is thus enough to prove the following two statements:

- The invariant $|p_i| \leq |\chi_i(\overline{p_i})|$ holds at every step of our algorithm for all players i.
- When the algorithm terminates, all the edges of OPT are bought by the players.

Computing deviations Here we discuss how deviations can be computed. This will be important in Section 3.4 when we talk about our polynomial-time results, but in this section we are only concerned with existence results and so include this discussion only in order to improve intuition about the nature of deviations. When computing $\chi_i(\overline{p_i})$, note that all edges of OPT such that i is not contributing any payment to them can be used by i freely to satisfy her connectivity requirement. Therefore, when computing the cheapest deviation for a player i, we should not use the actual cost of the edges in G, but instead for each edge f, we should use the cost i would face if she is to use f, which will be referred to as *modified cost of f for i,* and denoted by $c'_i(f)$ in the rest of the paper. 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 contributed anything to (i.e., $p_i(f) = 0$), we have that $c'_i(f) = 0$, since from i's perspective, she can use these edges for free because other players have paid for them. 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 her deviation from the payment strategy p_i. Using these modified costs, $\chi_i(\overline{p_i})$ is simply the cheapest set of edges which fulfill player i's connectivity requirements.

We now present an algorithm which satisfies the two properties mentioned above (the pseudocode is shown in Algorithm 2). At the beginning of the algorithm $|p_i| = 0$ for all players i, and therefore the invariant is trivially satisfied by all the players. At every step, the algorithm asks a player i to make a payment for an incident edge e of i. Let $d(e)$ denote the amount of payment i should make in order to buy e, i.e., $d(e) = c(e) - \sum_{j \neq i} p_j(e)$. Recall that the algorithm should never assign a payment for a player that violates the invariant $|p_i| \leq |\chi_i(\overline{p_i})|$. Let x denote the maximum amount of payment player i can make for e in order not to violate the invariant. If $x \geq d(e)$ then the algorithm should ask i to pay $d(e)$ for e and ask it to pay x for e otherwise, and therefore the invariant is never violated throughout the algorithm.

What is this value x, however, and how to we compute it? For a strategy p_i of player i, let $\chi_i(\overline{p_i}, e)$ denote the cheapest deviation of player i from the strategy p_i that does not use the edge e, assuming that the rest of the players are paying for $\overline{p_i}$. Observe that $|\chi_i(\overline{p_i}, e)| \geq |\chi_i(\overline{p_i})|$. Then we argue below that $x \geq \min\{|\chi_i(\overline{p_i}, e)| - |p_i|, d(e)\}$. To see this, we consider the strategy $p_i + x$ where player i pays x for edge e. We are abusing notation here, since x is a number, not a payment vector. Formally, by $p_i + x$ we will mean the payment strategy which equals p_i everywhere except at e, with $(p_i + x)(e) = p_i(e) + x$.

Lemma 9 *Given payments p_i which do not violate the invariant, player i can increase her payments on edge e by $\min\{|\chi_i(\overline{p_i}, e)| - |p_i|, d(e)\}$ and not violate the invariant.*

Proof. To see this, suppose that x is the maximum amount that i can pay for edge e without violating the invariant, and $x < d(e)$. Now suppose to the contrary that $x < |\chi_i(\overline{p_i}, e)| - |p_i|$. This means that $|p_i + x| = |p_i| + x < |\chi_i(\overline{p_i}, e)|$. By Lemma 10, we know that $|\chi_i(\overline{p_i}, e)| = |\chi_i(\overline{p_i + x})|$. Therefore, $|p_i + x| < |\chi_i(\overline{p_i + x})|$, and so we can increase x and still not violate the invariant. This gives us a contradiction with x being maximum. ■

Lemma 10 *Let p_i denote the strategy of player i right before she is asked to make a payment for e and let $x < d(e)$ be the maximum amount of payment i can make for e without violating the invariant. Then, $|\chi_i(\overline{p_i}, e)| = |\chi_i(\overline{p_i + x})|$, i.e, at least one of the best deviations of player i right after she makes a payment of x for e does not use e.*

Proof. For the purpose of contradiction, assume that all of the cheapest deviations of player i after she pays x for e, i.e., $\chi_i(\overline{p_i + x})$, uses e. Let $\mu = |\chi_i(\overline{p_i + x}, e)| >$

```
Input: The socially optimal network OPT
Output: The payment scheme for OPT that is a Nash
        equilibrium
Initialize $p_i(e) = 0$ for all players $i$ and edges $e$;
Root each tree $T$ of OPT by an arbitrary node incident to an
edge witnessed from 2-sides;
Loop through all trees $T$ of OPT;
    Loop through all nodes $i$ of $T$ in reverse BFS order;
        Let $T_i$ be the subtree of $T$ below $i$;
        Loop through all edges $e$ of $T_i$ incident to $i$;
            Let $d(e) = c(e) - \sum_{j \neq i} p_j(e)$;
            If $|\chi_i(\overline{p_i}, e)| - |p_i| \geq d(e)$
                Set $p_i(e) = d(e)$;
            Else break;
        Let $g$ to be the parent edge of node $i$;
        Set $p_i(g) = \min\{|\chi_i(\overline{p_i}, e)| - |p_i|, c(g)\}$;
```

Algorithm 2: Algorithm that generates payments on the edges of OPT

$|\chi_i(\overline{p_i + x})|$ be the cost of the cheapest deviation of player i that does not use e right after i pays x for e. Observe that if the payment of i for e is increased by $y = \min\{\mu - |\chi_i(\overline{p_i + x})|, d(e) - x\}$, the cost of all deviations of player i that use edge e increases by the same amount. The invariant is still not violated after this increase, which contradicts with the assumption that x is the maximum amount of payment she can make for e without violating the invariant. ■

Because of Lemma 9, it is clear that Algorithm 2 maintains the invariant at every step, since we never ask a player to pay for more than $\min\{|\chi_i(\overline{p_i}, e)| - |p_i|, d(e)\}$ for an edge e. We must now show that Algorithm 2 purchases all the edges of OPT. Recall that the algorithm asks the players to pay for their incident edges only. Therefore, each edge is considered for payment twice. For each edge $e = (i, j)$ where j is the parent of i, first i is asked to make her maximum amount of payment for e that will not violate the invariant. At the later iterations of the algorithm, when j is processed, the algorithm asks j to pay for the remaining cost of e. Therefore, if the payment algorithm successfully pays for the cost of all the edges of OPT, i.e., it does not execute the 'break' command, 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 executes the 'break' command at any intermediate step. We will prove this by constructing a network cheaper than OPT which satisfies all the players' connectivity requirements whenever the algorithm executes 'break', thus forming a contradiction.

Specifically, we will consider networks formed by players' deviations. Recall that by Lemma 10, when a player i cannot pay $d(e)$ but some amount $x < d(e)$ for an incident edge e without violating the invariant, she does have a deviation that costs as much as her strategy $p_i + x$ and does not use e. Define $X_i(p_i, e)$ to be the graph formed by removing the edges paid for by $p_i + x$ from OPT, and then adding the edges paid for in $\chi_i\left(\overline{p_i}, e\right)$. In other words, $X_i(p_i, e)$ is the network of bought edges formed if player i deviates from her current strategy $p_i + x$ to $\chi_i\left(\overline{p_i}, e\right)$, with the payments of the other players being $\overline{p_i + x}$. The edges added to OPT by this deviation cost at most $|p_i + x|$, and the edges removed cost strictly greater than $|p_i + x|$. The cost is *strictly* greater because e is one of the edges removed, and player i does not fully pay for edge e in the payment $p_i + x$. Therefore, we know that the graph $X_i(p_i, e)$ is strictly cheaper than OPT.

Since the algorithm roots T by a node incident to an edge witnessed from 2-sides and the edges of T that are witnessed from 2-sides form a connected component in T by Corollary 1, every edge $e = (i, j)$ of T that is witnessed from 1-side is witnessed from the side of the lower level adjacent node. Without loss of generality, assume i is the lower level adjacent node of e, i.e., $e \notin T_i$ and $e \in T_j$. The algorithm cannot execute the 'break' command while a player is asked to pay for the cost of an edge e witnessed from 1-side since, as Lemma 11 proves, i will pay for the whole cost of e when she is asked to make a payment for e. In order to show that the algorithm does not execute the 'break' command when a player is asked to pay for an edge that is witnessed from 2-sides, we need the nice properties of graphs $X_i(p_i, e)$ given by Lemma 12 and Lemma 13.

Lemma 11 *Let $e = (i, j)$ be an edge of T that is witnessed from* 1*-side, and let i be the lower level incident node. Then when the algorithm asks player i to make payment for e, she will pay for the entire cost of e.*

Proof. We will prove the lemma by induction on the number of edges in the subtree T_i. First consider the case where i is a leaf-node of T, i.e., the number of edges in T_i is 0, as the base case of induction. If the player in i does not pay an amount $c(e)$ for the edge e then she has a deviation $\chi_i\left(\overline{p_i}, e\right)$ whose cost is less than $c(e)$. Then we will ask player i to play $\chi_i\left(\overline{p_i}, e\right)$ as her strategy. The graph $X_i(p_i, e)$ obtained by deleting e from OPT and adding the edges bought by the strategy $\chi_i\left(\overline{p_i}, e\right)$ is clearly cheaper than OPT since the total cost of the edges bought by $\chi_i\left(\overline{p_i}, e\right)$ is less than $c(e)$. Since none of the players except player i was witnessing e in OPT, the connectivity requirements of all other players will be satisfied in $X_i(p_i, e)$. The connectivity requirement of player i is trivially satisfied since she is the only deviating player. Since $X_i(p_i, e)$ satisfies all the players and is cheaper than OPT, the player in i will pay $c(e)$ for e (otherwise we have a contradiction).

Now consider an arbitrary edge $e = (i, j)$ that is witnessed from 1-side. Note that all the edges in T_i are witnessed from 1-side by Proposition 1 and therefore are bought by their lower level adjacent players by the inductive assumption. So player i is not asked to pay for any edge but e by the algorithm. Assume player i does not pay an amount $c(e)$ for the edge e when she is asked to make payment for e. Then she has a deviation $\chi_i\left(\overline{p_i}, e\right)$ whose cost is less than $c(e)$. Then we will ask player i to play $\chi_i\left(\overline{p_i}, e\right)$ as her strategy. Similar to the above case, $X_i(p_i, e)$ is clearly cheaper than OPT since the total cost of the edges bought by $\chi_i\left(\overline{p_i}, e\right)$ is less than $c(e)$. Since none of the players except the ones in T_i were witnessing e in OPT and e is the only edge of OPT that is not part of $X_i(p_i, e)$, then the connectivity requirements of all the players except the ones in T_i are satisfied in $X_i(p_i, e)$. However, since all the edges of T_i are part of $X_i(p_i, e)$, all the players of T_i are in the same connected component of $X_i(p_i, e)$. Since $X_i(p_i, e)$ satisfies the connectivity requirement of player i(the happiness function F evaluates to 1 for the connected component of i in $X_i(p_i, e)$), it also satisfies the connectivity requirements of all the players in T_i as well. Since $X_i(p_i, e)$ satisfies the connectivity requirements of all the players and cheaper than OPT, we have a contradiction. Therefore, player i pays the whole cost $c(e)$ of e when the algorithm asks her to make a payment for e. ■

Lemma 12 *Let C be the connected component containing i in $X_i(p_i, e)$. Then, C contains either all players of T or another tree T' of OPT.*

Proof. By Lemma 11, all the subtrees T_u of i that are linked to i by a 1-sided edge (u, i) will be in C, since the edge (u, i) will be entirely paid for by u. Let T_v be a subtree linked to i by a 2-sided edge (v, i), and suppose to the contrary that T_v is not in C. Since (v, i) is a 2-sided edge, then $T - T_v$ is not a happy component, and so in order for the connectivity requirement of i to be satisfied, she must be connected to some tree T' of OPT, as desired. ■

Let i be a player that paid $x < c(e)$ for her upper level incident edge $e = (i, j)$ when the algorithm asked her to make a payment for e. Then, by Lemma 12, the connected component C of $X_i(p_i, e)$ that contains i, either contains all the players of T or another tree T' of OPT. If C contains all the players of T then $X_i(p_i, e)$ would be a graph cheaper than OPT and satisfying the connectivity requirements of all the players, which would be a contradiction. Assume C does not contain all the players of T but another tree T' of OPT. Let S be a subset of the players in T_i such that for every $u \in S$, u paid strictly less than $c(f)$ for her upper level incident edge f when the algorithm asked her to pay for f. The following lemma states that we can obtain a graph G' cheaper than OPT that satisfies the connectivity requirements of all the players in T_i, by replacing the strategies p_u of some elements u of S with their respective deviations $\chi_u(\overline{p_u}, f)$, none of which use any of the edges of $T - T_i$. In other words, if a player i cannot pay the whole cost of her upper level incident edge then there is a way to satisfy the connectivity requirements of all the players in T_i without increasing their cost and without relying on the payments of the players in $T - T_i$.

Lemma 13 *Let i be a player that could not pay the whole cost of her upper level incident edge e and the connected component of $X_i(p_i, e)$ that contains i does not contain all the players of T. Then, one can obtain a graph G' cheaper than OPT that satisfies the connectivity requirements of all the players in T_i, by replacing the strategies p_u of some elements u of S with their respective deviations $\chi_u(\overline{p_u}, f)$, none of which use any of the edges of $T - T_u$.*

Proof. We will prove the lemma by induction on the number of nodes in T_i. Consider the case where i is a leaf node as the base case. If player i cannot pay $c(e)$ but some amount $x < c(e)$ when the algorithm asks her to pay for her upper level incident edge e, then she has a deviation $\chi_i(\overline{p_i}, e)$ such that $|\chi_i(\overline{p_i}, e)| = x < c(e)$. Observe that the set of edges bought in $X_i(p_i, e)$ cannot be connecting i to $T - T_i$ since e is the cheapest path between i and $T - T_i$. Therefore, i will be connected to a different tree of OPT without connecting to $T - T_i$ if she plays $\chi_i(\overline{p_i}, e)$ as her strategy. Since i is the only deviating player, her connectivity requirements are trivially satisfied in $X_i(p_i, e)$ and since there is no other player in T_i, the lemma holds.

Consider the case where i is not a leaf-node. If player i cannot pay $c(e)$ but some amount $x < c(e)$ when the algorithm asks her to pay for her upper level incident edge e, then she has a deviation $\chi_i(\overline{p_i}, e)$ such that $|\chi_i(\overline{p_i}, e)| = |p_i + x| < |p_i| + c(e)$ where p_i is the strategy of player i right before she pays for e and therefore, $X_i(p_i, e)$ is strictly cheaper than OPT. By Lemma 12, the connected component C of $X_i(p_i, e)$ that contains i, either contains all terminals of T or a different tree T' of OPT. As pointed out above, C cannot contain all the terminals in T since otherwise $X_i(p_i, e)$ would be a graph cheaper than OPT that satisfies the connectivity requirements of all the players. Therefore, C contains a different tree T' of OPT. Observe that i is not connected to the terminals of $T - T_i$ in $X_i(p_i, e)$ since the shortest path between i and $T - T_i$ is e and therefore, $\chi_i(\overline{p_i}, e)$ would not be a best deviation of player i otherwise. The set of players in the subtrees of T_i that are connected to i in $X_i(p_i, e)$ (included in C) are connected to T' as well, and so their connectivity requirements are also satisfied in $X_i(p_i, e)$. Let $i_1, i_2, \ldots, i_k$ be the children of i in T such that they are not connected to i in $X_i(p_i, e)$, i.e., C does not contain them. By the inductive hypothesis, for every subtree T_{i_j}, there is a set of players S_j in T_{i_j} such that by replacing the strategies of the players in S_j with their respective best deviations, none of which uses any of the edges of $T - T_{i_j}$, we can obtain a graph cheaper than OPT where the connectivity requirements of all the players in T_{i_j} are satisfied.

Consider the graph G' that results by replacing the strategies of the players in $\cup_j S_j \cup \{i\}$ with their respective best deviations. Notice that all the players in each subtree T_{i_j} are connected to a tree of OPT other than T in G', since the best deviations of none of the players in S_j uses any edges of $T - T_{i_j}$, and so the fact that other players $(\cup_{l \neq j} S_l \cup \{i\})$ deviate and possibly remove their payments from some edges of $T - T_{i_j}$ does not affect them. The players connected to i in $X_i(p_i, e)$ are still connected to a tree T', since all the edges of the connected component C of $X_i(p_i, e)$ are also in G'. Therefore, all the players of T_i are connected to trees of OPT other than T. Since the happiness function F is monotone, this means that all the connectivity requirements of all players in T_i are satisfied. ■

We are now ready to prove that Algorithm 2 never executes the 'break' command, and thus pays for all the edges of OPT. For the purpose of contradiction, assume Algorithm 2 executed the 'break' command right after a player i is asked to pay for an incident edge $e = (v, i)$. Since the algorithm only executes the 'break' command if the sum of payments of the adjacent players v and i does not cover the total cost of the edge and the higher level incident player is always asked later than the lower level incident player, e is the lower level incident edge of player i. Observe that e is witnessed from 2-sides since otherwise v would have already paid for the total cost of e by Lemma 11. Recall that since i cannot pay for the remaining cost $d(e)$ of e but some amount $x < d(e)$ when she is asked to pay for e without violating the invariant, she does have a deviation $\chi_i(\overline{p_i}, e)$ such that $|\chi_i(\overline{p_i}, e)| = |p_i| + x$ where p_i is the strategy of player i right before she pays for e. Recall that the connected component C of $X_i(p_i, e)$ that contains i, either contains all terminals of T or a different tree T' of OPT by Lemma 12. However, C cannot contain all terminals of T since otherwise $X_i(p_i, e)$, which is cheaper than OPT, would be a network satisfying the connectivity requirements of all the players. Therefore, C contains a different tree T' of OPT, as well as $T - T_i$ (since i has not paid for any of the edges of $T - T_i$) and possibly some but not all subtrees of T_i. So, $X_i(p_i, e)$ satisfies the connectivity requirements of all the terminals except the ones that are in the subtrees of T_i which are not in C.

Let $T_{i_1}, T_{i_2}, \ldots, T_{i_k}$ be the subtrees of T_i that are not contained in C and let $f_1, f_2, \ldots, f_k$ be the upper level incident edges of the roots of these subtrees in

OPT. Observe that none of the players at the roots of $T_{i_1}, T_{i_2}, \ldots, T_{i_k}$ have paid for the entire cost of their respective upper level incident edges $f_1, f_2, \ldots, f_k$ when the algorithm asked, since otherwise player i would contribute nothing to these edges, and so the subtrees $T_{i_1}, T_{i_2}, \ldots, T_{i_k}$ would be in C. To complete the proof all we need to prove is that we can modify $X_i(p_i, e)$ without increasing its cost such that we connect the players in $T_{i_1}, T_{i_2}, \ldots, T_{i_k}$ to happy connected components. This is exactly what Lemma 13 proves.

3.3 Good Equilibria in the General Game

In Section 3.2, we saw that a good equilibrium always exists when all nodes are terminals. In this section, we consider the general Group Network Formation Game, and show that there always exists a 2-approximate Nash equilibrium that is as cheap as the centralized optimum. By a 2-approximate Nash equilibrium, we mean a strategy profile $p = (p_1, p_2, \ldots, p_n)$ such that no player i can reduce her cost by more than a factor of 2 by unilaterally deviating from p_i to $\chi_i(p_{-i})$, i.e., $|\chi_i(p_{-i})| \geq |p_i|/2$ for all players i. To prove this, we first look at an important special case that we call the *Group Network Formation of Couples Game* or *GNFCG*. This game is exactly the same as the Group Network Formation Game, except that every player node has at least two players located at that node (although not all nodes need to be player nodes).

Theorem 10 *If the price of stability for the GNFCG is* 1 *then there exists a* 2*-approximate Nash Equilibrium for the Group Network Formation Game that costs as much as OPT.*

Proof. Assume we are given an instance $\Im_1 = (N_1, G, T, F)$ of a Group Network Formation Game, i.e., we are given a set of players $N_1 = \{1, 2, \ldots, n\}$, a graph $G = (V, E)$ such that each edge $e \in E$ is associated with a nonnegative cost $c(e)$, a set of terminal nodes $T \subseteq V$ such that each player $i \in N_1$ is located at a terminal node $u \in T$ and a monotone happiness function $F : 2^T \rightarrow \{0, 1\}$. All we need to show is that $\Im_1$ has a 2-approximate Nash equilibrium as cheap as OPT assuming price of stability for the Group Network Formation of Couples Game is 1.

We will first define an instance $\Im_2 = (N_2, G, T, F)$ of the Group Network Formation of Couples Game. Observe that the graph G, the set of terminal nodes T and the monotone happiness function F for both $\Im_1$ and $\Im_2$ are the same; however, the number of players of $\Im_2$ is twice as much as the number of players of $\Im_1$, i.e., $N_2 = \{1, 2, \ldots, 2n\}$. For each player $i \in N_1$ of $\Im_1$ located at $u \in T$ there are 2 corresponding players $i, (n+i) \in N_2$ of $\Im_2$ located at u.

First observe that the socially optimal network for both games $\Im_1$ and $\Im_2$ is the same since any network satisfying the players of $\Im_1$ also satisfies the players of $\Im_2$ and vice versa. Let OPT denote the socially optimal network of both of the games. As in Section 3.2, let p^* denote the strategy vector that buys all the edges of OPT, i.e., $p^*(e) = c(e)$ if e is in OPT and $p^*(e) = 0$ otherwise.

Since we have assumed that the price of stability for Group Network Formation of Couples Game is 1, we know that there exists a stable strategy profile $p = (p_1, p_2, \ldots, p_{2n})$ for the players N_2 of $\Im_2$ that buys OPT. That is, for each player $i \in N_2$, $|\chi_i(p^* - p_i)| = |p_i|$. Furthermore, since player i and player $(n+i)$ sit at the same terminal node for $i \leq n$, the stable strategy p_i of player i is also a stable strategy for player $(n+i)$, i.e., if all the players except player $(n+i)$ pays $p^* - p_i$ then the best response of player $(n+i)$ is p_i.

To complete the proof, all we need to do is to give a strategy profile $p' = (p'_1, p'_2, \ldots, p'_n)$ for the players of $\Im_1$ such that $\sum_i p'_i = p^*$ and for each $i \in N_1$, $|p'_i| \leq 2\,|\chi_i(p^* - p'_i)|$. We define p' as follows. For each player $i \in N_1$, $p'_i = p_i + p_{(n+i)}$. Since p_i and $p_{(n+i)}$ are stable strategies for players i and $(n+i)$ of $\Im_2$ respectively, we have that $|p_i| = |\chi_i(p^* - p_i)|$ and $|p_{(n+i)}| = \left|\chi_i\left(p^* - p_{(n+i)}\right)\right|$. Since $|\chi_i(p^* - p_i)|, \left|\chi_i\left(p^* - p_{(n+i)}\right)\right| \leq |\chi_i(p^* - p'_i)|$, we have $|p'_i| = |p_i| + |p_{(n+i)}| \leq 2\,|\chi_i(p^* - p'_i)|$. ■

Because of Theorem 10, we will focus on the GNFCG in the rest of the section and prove the existence of a Nash equilibrium as cheap as OPT. This result is interesting in its own right, since it states that to form an equilibrium that is as good as the optimum solution, it is enough to double the number of players. Such results are already known for many variants of congestion games and selfish routing [5, 53], but as Theorem 11 shows, we can also prove such results for games with arbitrary sharing.

We use the same notation as in Section 3.2, including the definitions of p^*, $\overline{p_i}$, and χ_i. Given a set of bought edges T, a strategy profile p is a Nash equilibrium when $|\chi_i(p_{-i})| = |p_i|$ for all players i. To prove that price of stability is 1 for GNFCG, we give an algorithm that forms such a strategy profile on the edges of OPT. Recall that the payment strategies of all the players have to be stable when the algorithm terminates. As in Section 3.2, to have an easier analysis we not only want our algorithm to ensure stability at termination but also at each intermediate step. Specifically, we give an algorithm such that:

- The invariant $|p_i| \leq |\chi_i\left(\overline{p_i}\right)|$ holds at every step of our algorithm for all players i.
- When the algorithm terminates, all the edges of OPT are bought by the players.

The second property guarantees that the invariant is exactly the Nash equilibrium condition, since when all the edges of OPT are bought, then $p_{-i} = \overline{p_i}$. Therefore, the above two conditions imply that the price of stability is 1. In the rest of the section we prove our main theorem for the GNFCG.

Theorem 11 *For GNFCG, there exists a Nash equilibrium as cheap as the socially optimal network, i.e., the price of stability is* 1.

For ease of explanation, we will first consider the case where all the edges of OPT are witnessed from two sides and later illustrate how our algorithm can be modified for the case where some of the edges are witnessed from one side only. We start by rooting each connected component of OPT arbitrarily by a high degree non-player node. Throughout the paper, the term *high degree node* refers to the nodes with degree 3 or more. On each connected component T of OPT, we run a 2-phase algorithm. In the first phase of the algorithm, we assign players to make payments to the edges of T in a bottom-up manner, i.e., we start from a lowest level edge e of T and pick a player i to make some payment for e and continue with the next edge in the reverse BFS order. In the first phase of the algorithm, we ask a

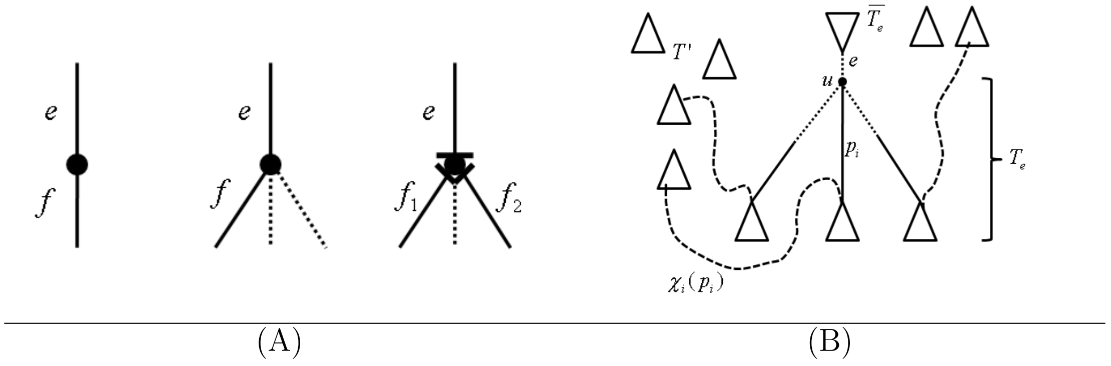

Figure 3.1: (A) Illustrates the assignment of the player to pay for the cost of e. (B) Shows how to construct a cheap network that satisfies all the players in T_e by using the deviations of a subset S of them.

player i to contribute only for the cost of edges on the unique path between her and the root and furthermore, the payment for each edge is made by only one player.

Algorithm (Phase 1) For an arbitrary edge $e = (u, v)$ where u is the lower level incident node of e, the assignment of the player to pay for e is as follows. If u is a terminal node, we ask a player i located at node u to make maximum amount of payment on e that will not make p_i violate the invariant, i.e., we set $p_i(e) = \min\{\chi_i(\overline{p_i}, e) - |p_i|, c(e)\}$. This will not violate the invariant by the same reasoning as in Section 3.2 (i.e., Lemma 9 still holds). If u is a degree 2 nonterminal node then we ask the player who has completely bought the other incident edge of u, i.e., made a payment equal to the cost of that edge, to make maximum amount of payment on e that will not make her strategy unstable (i.e., violate the invariant) as shown on the left of Figure 3.1(A). Note that it may be the case that no player has bought the other incident edge of u in which case we don't ask any player to pay for e and the payment for e will be postponed to the second phase of the algorithm. If u is a high degree nonterminal then the selection of the player to pay for e is based on the number of lower level incident edges of u that are bought in the previous iterations of the algorithm. If none of the lower level incident edges of u are bought then we postpone the payment on e to the second phase of the algorithm. If exactly one of the lower level incident edges of u, namely f, is bought then we ask the player who bought f to make maximum amount of payment on e that will not make her strategy unstable (i.e., violate the invariant) as shown in the middle of Figure 3.1(A). If 2 or more of the lower level incident edges of u are already bought, namely $f_1, f_2, \ldots, f_l$, then we fix the strategies of the players $i_1, i_2, \ldots, i_l$ that bought those edges, i.e., the players $i_1, i_2, \ldots, i_l$ are not going to pay any more and therefore the strategies of those players that will be returned at the end of the algorithm are already determined. Since there are two players located at every terminal, pick an arbitrary player located at the same terminal as one of $i_1, i_2, \ldots, i_l$ that has not made any payments yet, and assign her to make maximum amount of payment for e that will not make her strategy unstable as shown on the right of Figure 3.1(A). We later prove that such a player always exists, i.e., not all of $i_1, i_2, \ldots, i_l$ are the last players to make payment at their respective terminal nodes.

Notation We now define some helpful notation in order to prove some lemmas about the first phase of the algorithm which is fully specified above. When we are talking about a player i, let T denote the connected component of OPT containing i and let T' denote the set of other connected components of OPT. The strategy of a player is denoted by p_i which is a vector of length m, the total number of edges in G, where each entry of p_i indicates the payment player i is making for the corresponding edge. Recall that we use p^* for the strategy vector that buys OPT, i.e., $p^*(e) = c(e)$ if e is an edge of OPT and $p^*(e) = 0$ otherwise. Observe that when the algorithm terminates it should be that $\sum_i p_i = p^*$. We use the notation $G(p)$ for the subgraph of bought edges by strategy p, i.e., the subgraph composed of the edges for which $p(e) = c(e)$. For instance, $G(p^*)$ denotes OPT, $G(p^* - p_i)$ denotes the subgraph composed of edges bought by players other than i and $G(p^* - p_i + \chi_i(\overline{p_i}, e))$ denotes the subgraph of bought edges if player i deviates from her strategy p_i to her best deviation that does not use e, $\chi_i(\overline{p_i}, e)$. Finally, for an arbitrary edge e of a rooted tree T, we use T_e in order to refer to the subtree of T below e and $\overline{T_e}$ to refer to the rest of the tree $T - T_e$. The notation for the subtree of T rooted at a node u is analogously T_u.

We now present the analysis of the first phase of the algorithm by giving a series of lemmas that successively proves the following. For every edge e that could not be bought in the first phase of the algorithm by the assigned player to make payment for it, we can connect all the terminal nodes in T_e to the connected components of T' *without using any of the edges of* $\overline{T_e}$ by simply setting $p_i = \chi_i(\overline{p_i})$ for a subset S of players in T_e. The deviations of the subset S of the players are depicted in Figure 3.1(B). The condition that no edges of $\overline{T_e}$ are used by the deviations is crucial, since that is what allows us to have a set of players all deviate at once and still be satisfied afterwards. The fact that such a "re-wiring" exists allows us to argue in our proofs that at least one of the incident edges of the root of T will be bought during the first phase of the algorithm. In all the lemmas below, the payment p will refer to the payment at the end of Phase 1 of the algorithm.

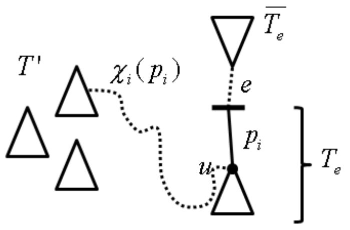

Figure 3.2: Shows the deviation $\chi_i(p_i)$ of a player i that is located at terminal u and could not buy the edge e.

Lemma 14 *Let u be an arbitrary terminal node and i be the first player to make payments that is located at u. Let e be the first edge between u and the earliest ancestor of u which is either a terminal or a high degree nonterminal node such that i could not buy all of e without violating the invariant. Then all the players in T_u will be satisfied in the subgraph $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right) - \overline{T_e}$.*

Proof. Since every edge f between u and e is bought by player i, i.e., $p^*(f) = p_i(f) = c(f)$ and player i did not pay for any other edge in G, then the subgraph $G(p^* - p_i)$ consists of the connected components T', T_u and $\overline{T_e}$. Observe that the players in T_u, including i, are not satisfied in $G(p^* - p_i)$ since according to our assumption, any edge of OPT is witnessed from two sides, and so any subset of T_e

cannot be a happy component. Since player i is satisfied in $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$ and not satisfied in $G(p^* - p_i)$, then the subgraph $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right) - G(p^* - p_i)$ must include a path P_u between T_u and one of the connected components in T' or $\overline{T_e}$.

For the purpose of contradiction, assume P_u is a path between T_u and $\overline{T_e}$. Since $T - T_u - \overline{T_e}$ is a path of degree 2 nonterminal nodes between T_u and $\overline{T_e}$, then all the players of T will be in the same connected component of $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$ and therefore the subgraph $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$ satisfies all the players. This is because i must be satisfied in $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$, and so it is in a happy component. Observe that $G\left(p^* - p_i\right)$ has the same set of edges as OPT except e and the edges bought by player i. Since the cost of $\chi_i(\overline{p_i}, e)$ is equal to the cost of p_i by Lemma 10 (which still holds), then $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$ is a subgraph satisfying all the players and cheaper than OPT. More precisely, the cost of $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$ is less than the cost of OPT by an amount $c(e) - p_i(e)$. Since OPT is the cheapest network satisfying all the players, this is a contradiction and therefore there cannot exist a path between T_u and $\overline{T_e}$ in $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$. Since the players in T_u are in a connected component that is disjoint from all the nodes and the edges of $\overline{T_e}$ in $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$, then they are all satisfied in $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right) - \overline{T_e}$ as shown in Figure 3.2. ■

Lemma 15 *Let i be a player such that i did not buy e, i.e., $p_i(e) < c(e)$, even though the algorithm assigned i to make payment for e. Then there exists a subset of players $S = (s_1, \ldots, s_k)$ in T_e with deviations $\chi_1(\overline{p_1}, f_1), \ldots, \chi_k(\overline{p_k}, f_k)$, where f_l is the edge player l could not fully buy, from their respective strategies $p_1, \ldots, p_k$ such that all the players in T_e will be satisfied in the subgraph $G\left(p^* - \sum_{l \in S} p_l + \sum_{l \in S} \chi_l(\overline{p_l}, f_l)\right) - \overline{T_e}$.*

Proof. We will prove this result by induction on the number of nodes in T_e. If T_e has only 1 node u, then the lemma holds by Lemma 14. Below we will use the notation $R(S)$ to denote $G(p^* - \sum_{l \in S} p_l + \sum_{l \in S} \chi_l(\overline{p_l}, f_l))$ and $R(S, e)$ to denote $R(S) - \overline{T_e}$.

Let us assume that the lemma holds for all instances such that the number of nodes in T_e is at most k and let's prove that the lemma also holds when the number of nodes is $k+1$. Let Γ be the set composed of connected components of $G\left(\sum_{i \in T_e} p_i\right)$ that involves at least one terminal node and let C be the connected component involving i. Observe that C is the highest level connected component, i.e., the one that is adjacent to e.

Let u be the highest level terminal or high degree nonterminal node in C. If u is a terminal node then the result directly follows from Lemma 14, with $S = \{i\}$, since the terminals in T_u are the same as in T_e. Therefore, assume u is a high-degree nonterminal node. Observe that at least one of the lower level incident edges of u is a bought edge since otherwise C would not be a connected component of $G\left(\sum_{i \in T_e} p_i\right)$ that involves at least one terminal node. Recall that if at least 2 of the lower level incident edges of u are bought, i is only assigned to make payment for the edges above u, i.e., player i has not made any payment on the edges below u. Therefore, all we need to do is to exactly repeat the proof of Lemma 14, once again giving us the desired result with $S = \{i\}$. Let us now consider the final case, which is depicted in Figure 3.1(B), where u is a high-degree nonterminal node such that exactly one lower level incident edge of u is bought.

In this case, let $C_1, C_2, \ldots, C_k$ be the elements of Γ that are one level lower than C and let $e_1, e_2, \ldots, e_k$ be the immediate higher level unpaid edges of $C_1, C_2, \ldots, C_k$ respectively. By the inductive hypothesis, each of these edges e_j already has a desired set of players S_j in T_{e_j}.

The connected component containing i in the subgraph $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$ may also contain a player j in T_{e_j}. Since player i did not make any payment for the edges in T_{e_j}, then all the edges of T_{e_j} are also part of $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$ and therefore all the players in T_{e_j} are in the same connected component with i in $G\left(p^* - p_i + \chi_i(\overline{p_i}, e)\right)$. Let Λ be the edges of $e_1, e_2, \ldots, e_k$ such that T_{e_j} is *not* in the same connected component of $G(p^* - p_i + \chi_i(\overline{p_i}, e))$ as i. Then, we set the set S to be $\cup_{e_j \in \Lambda} S_j \cup \{i\}$.

We must now prove that all the players in T_e are satisfied in the subgraph $R(S, e) = G(p^* - \sum_{l \in S} p_l + \sum_{l \in S} \chi_l(\overline{p_l}, f_l)) - \overline{T_e}$. First, we prove this for the players in T_{e_j} for $e_j \in \Lambda$. By the inductive hypothesis, they are all satisfied in the subgraph $R(S_j, e_j) = G\left(p^* - \sum_{l \in S_j} p_l + \sum_{l \in S_j} \chi_l(\overline{p_l}, f_l)\right) - \overline{T_{e_j}}$. Let g be an arbitrary edge in a connected component of a player in T_{e_j} in the subgraph $R(S_j, e_j)$. This edge cannot be in $\overline{T_e}$, since $\overline{T_e} \subseteq \overline{T_{e_j}}$. This edge cannot be paid for by a player outside T_{e_j}, since those players only pay for edges in $\overline{T_{e_j}}$. Therefore, g must still be present in $R(S, e)$, and so all players in T_{e_j} are still satisfied in $R(S, e)$.

Now consider the players in C and in T_{e_j} for $e_j \notin \Lambda$. They are satisfied in $G(p^* - p_i + \chi(\overline{p_i}, e))$ since i is satisfied and they are in the same connected component. Let g be an arbitrary edge in the connected component of $G(p^* - p_i + \chi(\overline{p_i}, e))$ containing player i. These players are satisfied in $R(S) = G\left(p^* - \sum_{l \in S} p_l + \sum_{l \in S} \chi_l(\overline{p_l}, f_l)\right)$, since g is still present in $R(S)$. This is because if g were being paid for by a player j, then it would be part of some subtree T_{e_j} with $e_j \notin \Lambda$, and so $j \notin S$, and those payments on g would remain unchanged. We need to prove that players in C and in T_{e_j} for $e_j \notin \Lambda$ are satisfied in $R(S, e)$, and so it is enough to show that $g \notin \overline{T_e}$. If this were not the case, then $\overline{T_e}$ is in the same connected component of $R(S)$ as i. Since i is satisfied in $R(S)$, then so are all the players in $\overline{T_e}$. We already proved that all players in all subtrees T_{e_j} are satisfied in $R(S)$, and so $R(S)$ is a feasible solution (all the players in the graph are satisfied). Notice, however, that the solution $R(S)$ is cheaper than OPT, by the same argument as in Lemma 14, and so this is not possible.

Therefore, all the players in T_e are satisfied in $R(S, e)$, as desired. ■

Lemma 16 *Let $C_1, C_2, \ldots, C_k$ be the highest level connected components of the subgraph $G\left(\sum_i p_i\right)$ that include at least one terminal node, and suppose that these do not include the root of T. Let $e_1, e_2, \ldots, e_l$ be the immediately above incident edges of $C_1, C_2, \ldots, C_k$ respectively. Let $\overline{T_{e_1, e_2, \ldots, e_l}} = T - T_{e_1} - T_{e_2} - \ldots - T_{e_k}$. Then for each C_j there exists a corresponding set S_j of players in T_{e_j} such that all players in T are satisfied in $G\left(p^* - \sum_{l=1}^{k}\left(\sum_{j \in S_l} p_j - \sum_{j \in S_l} \chi_j(\overline{p_j}, f_j)\right)\right) - \overline{T_{e_1, e_2, \ldots, e_k}}$, where f_j is the edge player j could not fully buy.*

Proof. Let i be an arbitrary player in T. W.l.o.g. assume it is in T_{e_j}. There is a set S_j such that player i is satisfied in $G\left(p^* - \sum_{j \in S_j} p_j - \sum_{j \in S_j} \chi_j(\overline{p_j}, f_j)\right) - \overline{T_{e_j}}$ due to Lemma 15. To prove the lemma, all we need to show is that every edge of $G\left(p^* - \sum_{j \in S_j} p_j + \sum_{j \in S_j} \chi_j(\overline{p_j}, f_j)\right) - \overline{T_{e_j}}$ is also an edge of the graph $G\left(p^* - \sum_{l=1}^{k}\left(\sum_{j \in S_l} p_j - \sum_{j \in S_l} \chi_j(\overline{p_j}, f_j)\right)\right) - \overline{T_{e_1, e_2, \ldots, e_k}}$.

First let us consider the edges of T. Observe that none of the edges in $\overline{T_{e_j}}$ are in $G(p^* - \sum_{j \in S_j} p_j + \sum_{j \in S_j} \chi_j(\overline{p_j}, e)) - \overline{T_{e_j}}$. Therefore all we need is to check the edges in T_{e_j}. Since the algorithm never assigns a player i to pay for the edges that are not on the unique path between the terminal i and the root of T, none of the the players in $T - T_{e_j}$ made payment for any edge in T_{e_j}. Since the players in $\bigcup_{l \neq j} S_l$ did not make payment on the edges of T_{e_j}, then an edge of T_{e_j} in $G\left(p^* - \sum_{j \in S_j} p_j + \sum_{j \in S_j} \chi_j(\overline{p_j}, f_j)\right) - \overline{T_{e_j}}$ is also an edge of the graph $G\left(p^* - \sum_{l=1}^{k}\left(\sum_{j \in S_l} p_j - \sum_{j \in S_l} \chi_j(\overline{p_j}, f_j)\right)\right) - \overline{T_{e1,e2,\ldots,e_k}}$.

Now let us consider the edges of $\overline{T}$, i.e., the edges outside of T. Since the algorithm never asks the players to pay for the edges outside of T, an edge not in T is in $G\left(p^* - \sum_{j \in S_j} p_j + \sum_{j \in S_j} \chi_j(\overline{p_j}, f_j)\right) - \overline{T_{e_j}}$ if and only if it is also in $G\left(\sum_{j \in S_j} \chi_j(\overline{p_j}, f_j)\right)$. So, any edge of $\overline{T}$ in $G\left(p^* - \sum_{j \in S_j} p_j + \sum_{j \in S_j} \chi_j(\overline{p_j}, f_j)\right) - \overline{T_{e_j}}$ is also in the graph $G\left(p^* - \sum_{k=1}^{l}\left(\sum_{j \in S_k} p_j - \sum_{j \in S_k} \chi_j(\overline{p_j}, f_j)\right)\right) - \overline{T_{e1,e2,\ldots,e_l}}$ as well. ∎

Lemma 17 *At least one of the incident edges of the root of T will be bought at the end of the first phase of the algorithm.*

Proof. For the purpose of contradiction assume none of the incident edges of the root of T is bought at the first phase of the algorithm and let's obtain a contradiction by constructing a subgraph that is cheaper than OPT and satisfies all the players. According to Lemma 16, the graph

$$R = G(p^* - \sum_{l=1}^{k}\left(\sum_{j \in S_l} p_j - \sum_{j \in S_l} \chi_j(\overline{p_j}, f_j)\right)) - \overline{T_{e1,e2,\ldots,e_k}}$$

satisfies all the players. Therefore, all we need to show is that the cost of R is less than the cost of OPT. Since the cost of $\chi_i(\overline{p_i}, f_i)$ is equal to the cost of p_i by Lemma 10, then we know by the same argument as in the proof of Lemma 14 that R is strictly cheaper than OPT. ∎

Algorithm (Phase 2) In the second phase of the algorithm, we ask the players that have not made any payments yet to make stable payments for the remaining edges and buy them. Let Γ be the set composed of connected components of $G(p) - T'$ that include at least one terminal node. In other words, Γ consists of connected components of the edges in T purchased so far by the algorithm (a single terminal node with no adjacent bought edges would also be a component in Γ). We call a connected component $C_1 \in \Gamma$ *immediately below* a connected component $C \in \Gamma$ if after contracting the components in Γ, C is above C_1 in the resulting tree and there are no other components of Γ between them. In the second phase of the algorithm, we form payments on the edges in a top-down manner as we explain next. We start from the connected component $C \in \Gamma$ that includes the root of T (this must exist by Lemma 17) and assign a player i in C that has not made any payments yet to buy *all* the edges between C and the connected components that are immediately below C. The set of edges i should buy are shown in Figure 3.3. We prove that such a player i always exists in Lemma 18. Observe that once i buys all the edges between C and the connected components $C_1, C_2, \ldots, C_k$ that are immediately below C, all these $k + 1$ connected components form a single connected C that contains the root. We repeat this procedure, i.e., pick a player i in the top-most connected component C that has not made a payment yet to buy all the edges between C and the connected components that are immediately below C, until all the players in T are in the same connected component and all of T is paid for.

Proof of Theorem 11. To show that our algorithm forms an equilibrium payment, we need to prove that all of OPT is paid for when it terminates, and that the invariant is never violated. It is clear that all of OPT is fully paid for, since Phase 2 of the algorithms pays for every edge of OPT that was not paid for in Phase 1. It is also clear that the invariant is never violated during the first phase by construction, and so the final payments of players used in Phase 1 are stable. To finish the proof, we need to show that a strategy p_i that buys all the edges between a connected component C and the connected components $C_1, C_2, \ldots, C_k$ that are immediately below C is a stable strategy for any player in C, which we show in Lemma 19.

This concludes the proof of Theorem 11. Recall that for ease of explanation, we only considered the case where all edges of OPT are witnessed from two sides until now. In Lemma 20, we modify this algorithm to return a Nash equilibrium

that purchases OPT even if some of the edges of OPT are witnessed from one side.
■

Lemma 18 *At any stage of the algorithm, each connected component of* Γ *has a player* i *such that* $p_i(e) = 0$ *for all edges* e.

Proof. The statement is trivially true at the start of the algorithm. In the first phase, consider a time when we ask a player i, which is not the first to pay among the players with the same terminal node of i, to make payment for some edges. This only occurs when at least 2 of the lower level incident edges of a high-degree nonterminal u are bought. But then 2 or more connected components of Γ merge. Since each of these connected components had at least one player who hasn't made any payment yet, and only one of them is being asked to pay at this moment, then every component of Γ still has at least one player that has not made any payments.

In the second phase of the algorithm, one player i buys all the edges between a connected component C and the connected components $C_1, C_2, \ldots, C_k$ that are immediately below C as shown in Figure 3.3. Similar to the above case, after the player i makes her payment, at least 2 connected components of Γ merge. Therefore, the lemma holds at the end of the second phase of the algorithm. ■

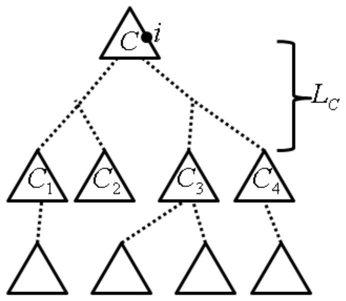

Figure 3.3: Illustrates the set of edges to be bought by a player i located at the connected component that contains the root.

Lemma 19 *Let* $C \in \Gamma$ *be a connected component of bought edges containing the root and let* $C_1, C_2, \ldots, C_k \in \Gamma$ *be the connected components of bought edges that are* *immediately below C. Then for any player i in C the strategy p_i that buys all the edges between C and $C_1, C_2, \ldots, C_k$ is stable, i.e., $|p_i| \le |\chi_i(\overline{p_i})|$.*

Proof. Let L_C denote the set of edges between C and $C_1, C_2, \ldots, C_k$. Observe that even though all the edges in L_C are not bought, a player j may have made a payment $p_j(e) < c(e)$ for some edge $e \in L_C$ in the first phase of the algorithm. Therefore, the cost of the strategy p_i of player $i \in C$ that buys all the edges of L_C, which we denote by l_i, may be less than $\sum_{e \in L_C} c(e)$. More precisely, $l_i = \sum_{e \in L_C} \Big(c(e) - \sum_j p_j(e) \Big)$.

We claim that a strategy p_i of a player $i \in C$ that buys all the edges in L_C is stable. For the purpose of contradiction, assume p_i is not a stable strategy, i.e, i could not pay the remaining cost of all the edges in L_C. Then, player i has a deviation $\chi_i(\overline{p_i})$ from p_i such that the cost of $\chi_i(\overline{p_i})$ is strictly less than l_i, and therefore the subgraph $G\left(p^* - p_i + \chi_i(\overline{p_i})\right)$ is cheaper than OPT. Since player i did not make any payment for the edges in C, all the players in C are in the same connected component of $G\left(p^* - p_i + \chi_i(\overline{p_i})\right)$ as i and therefore are satisfied. If the players in $C, C_1, C_2, \ldots, C_k$ are also in the same connected component with i in $G\left(p^* - p_i + \chi_i(\overline{p_i})\right)$, then $G\left(p^* - p_i + \chi_i(\overline{p_i})\right)$ satisfies all the players and is cheaper than OPT. Therefore, the players in some of the connected components $C_1, C_2, \ldots, C_k$ are not in the same connected component with i in $G\left(p^* - p_i + \chi_i(\overline{p_i})\right)$.

Let K be the subset of connected components $C_1, C_2, \ldots, C_k$ the players of which are not in the same connected component with i in $G\left(p^* - p_i + \chi_i(\overline{p_i})\right)$. Let $e_1, \ldots, e_d$ be the edges that were unpaid for in the first phase of the algorithm directly above the components in K. Then by Lemma 16, there is a set of players S such that all the players in $T_{e_1}, \ldots, T_{e_d}$ are satisfied in the graph $G(p^* - \sum_{l \in S} p_l + \sum_{l \in S} \chi_l(\overline{p_l}, f_l)) - \overline{T_{e_1, e_2, \ldots, e_d}}$. We claim that the subgraph $R = G(p^* - \sum_{j \in S \cup i} p_j + \sum_{j \in S \cup i} \chi_j(\overline{p_j}, f_j))$ satisfies all the players in T and is cheaper than OPT. The latter is clear since $|\chi_i(\overline{p_i})| < l_i = |p_i|$. All players in C and in the subtrees below $C_j \notin K$ are satisfied in R, since they are satisfied in $G\left(p^* - p_i + \chi_i(\overline{p_i})\right)$, and by construction, the payments p_j for $j \in T_{e_1}, \ldots, T_{e_d}$ do not contain edges of $\overline{T_{e_1, e_2, \ldots, e_d}}$. All players in components of K are satisfied in R as well, since the only edges missing from R that were in $G(p^* - \sum_{j \in S} p_j + \sum_{j \in S} \chi_j(\overline{p_j}, f_j)) - \overline{T_{e_1, e_2, \ldots, e_d}}$ are edges of p_i, which are all edges of $\overline{T_{e_1,e_2,\ldots,e_d}}$. Therefore, all the edges are still there that are needed to make the players in K (and the subtrees below them) be satisfied. Since we constructed a subgraph that satisfies all players and is cheaper than OPT, we have a contradiction. ■

Lemma 20 *Price of stability is* 1 *for the Group Network Formation of Couples Game even if some of the edges of OPT are witnessed from one side.*

Proof. We will show the result by slightly modifying the first phase of the algorithm. Recall that by Corollary 1, the edges witnessed from 2-sides form a connected component of T, which we will refer to as D, and so we root the tree T at a node in D. Observe that there exists a subset S of nodes of D such that all the edges that are witnessed from 1-side are subtrees of T rooted at the nodes of S. If OPT has edges witnessed from 1-side, i.e., not all nodes of S are leaf nodes, we ask the players to buy the edges witnessed from 1-side first. Specifically, for each $u \in S$, we ask the players in the subtree of 1-sided edges rooted at u to buy all the edges in their subtree, using the algorithm from [6] for the Single Source Connection Game. However, we ask only one player per terminal node to form payments in this algorithm. The proof that this set of players can indeed buy all the edges of this subtree using stable payments is exactly the same as the proof of the Single Source Connection Game payment algorithm so we will not repeat it here. Specifically, we can reduce this problem to the Single Source Connection Game by contracting $\overline{T_u}$ and T' into a single node, since every player in T_u must connect either to $\overline{T_u}$ or T' in order to be satisfied.

Once we have formed the payment on the edges witnessed from 1-side, we use our payment algorithm. If u is a terminal node, one of the players at u pays for the higher level incident edge of u in D. Let us now consider the case where u is a high degree nonterminal node. If the subtree composed of edges that are witnessed from 1-side has more than one terminal nodes, we contract the subtree and ask a second player j in one of these terminal nodes to pay for the higher level incident edge of u. The key observation here is that there are at least 2 players in the subtree (one per each terminal node) that did not make any payment yet, and so Lemma 18 still holds right after player j makes her payments. If the subtree has only one terminal node then the payment for the higher level incident edge depends on the number of lower level incident edges of u that are bought. If only one of the lower level incident edges of u is bought, the one that belongs to the subtree of edges witnessed from 1-side, then we ask this player that bought all the edges of the subtree to pay for the higher level incident edge of u. If at least 2 of the lower level incident edges of u are bought then 2 connected components of bought edges merge at u and therefore we ask a player that has not made payment yet to pay for the higher level incident edge of u. ■

3.4 Computing Equilibria in Polynomial Time

The proof of our 2-approximate Nash equilibrium result suggests an algorithm which forms a cheaper network whenever a 2-approximate Nash equilibrium cannot be found. Using techniques similar to [6], this allows us to form efficient algorithms to compute approximate equilibria:

Theorem 12 *Suppose we are given a feasible solution G_α whose cost is within a factor α of OPT. Then for any $\epsilon > 0$, there is a polynomial time algorithm which returns a $3.1(1+\epsilon)$-approximate Nash equilibrium on a feasible graph G', where $cost\,(G') \leq cost\,(G_\alpha)$. Furthermore, if all the nodes are player nodes, there is a polynomial time algorithm which returns a $(1+\epsilon)$-approximate Nash equilibrium on a feasible graph G', where $cost\,(G') \leq cost\,(G_\alpha)$.*

Proof. We will first prove that given an α-approximation to the socially optimal graph G_α for an instance of the Group Network Formation Game where all nodes are player nodes and any $\epsilon > 0$, there is a polynomial time algorithm which returns a $(1+\epsilon)$-approximate Nash equilibrium on a feasible graph G', where $cost\,(G') \leq cost\,(G_\alpha)$.

To define our algorithm, recall that the proof in Section 3.2 followed by constructing a network cheaper than the given one and the proof ended up with contradiction since the network at hand was optimal. The proof for obtaining a $(1+\epsilon)$-approximate Nash equilibrium in polynomial time on a given α-approximate socially optimal network G_α is based on following this suggested algorithm to obtain a cheaper network whenever a Nash equilibrium 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 $(1+\epsilon)$-approximate Nash equilibrium, i.e., a solution where no player can reduce its cost by more than a factor of $(1+\epsilon)$ 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 G. We now use our payment algorithms 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, a player may not pay for the cost of an incident one-sided edge or remaining cost of all her lower level incident edges, i.e., that Algorithm 2 could execute the 'break' command. However, the proofs of Lemma 12 and Lemma 11 indicate 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.

Observe that each call to our payment algorithm takes polynomial time if we can compute the best deviation of a player $\chi_i(\overline{p_i}, e)$ in polynomial time. Recall that $\chi_i(\overline{p_i}, e)$ is the cheapest set of edges (using modified costs) that fulfills i's connectivity requirements (see the beginning of Section 3.2 for a discussion of modified costs). By Lemma 12, $\chi_i(\overline{p_i}, e)$ is the cheapest set of edges (using modified costs) that connects player i to either to a different connected component T' of G_α or to all other terminals of T. To find the best deviation of player i, all we need to do is to find the cheapest deviation that connects i to a different connected component T' of G_α and the cheapest deviation of player i than connects i to all other terminals of T separately, and then take the cheaper one. The former one is essentially computing the shortest path from player i to any of the different connected components T' (using modified costs for the edges) and can be done in polynomial time. Computing the cheapest deviation of player i that connects i to all other terminals of T we do the following. We obtain a new graph by merging the terminal i with all other trees T' of G_α. Computing the cheapest deviation of player i that connects i to all other terminals of T corresponds to connecting all connected components of this new graph, since every node of the graph is a terminal, which can be done in polynomial time since this problem is the minimum spanning tree problem.

Thus we know that the Algorithm 2 can be made to run in poly-time, and that at every call to this algorithm it either forms a Nash equilibrium, or returns a solution that is cheaper by at least γ. Since each call which fails to form a Nash equilibrium reduces the cost by γ, we can have at most $\frac{\alpha(1+\epsilon)m}{\epsilon}$ calls. And since each call to our payment algorithm can be made to run in polynomial time, we obtained a network G' with $c(G') \leq c(G_\alpha)$ such that we have a Nash equilibrium on G' if the cost of its edges were decreased by γ in time polynomial in m and ϵ^{-1}.

For all payment strategies p_i 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. To show that we obtained a $(1+\epsilon)$-approximate Nash equilibrium, all we need to show is that each stable strategy p_i became $(1+\epsilon)$-approximately stable after they are proportionally increased.

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 $(1+\epsilon)$-approximate Nash equilibrium, note that p_i was a stable payment before it was increased and therefore $\chi_i(\overline{p_i})$, deviation of player i with respect to p_i, was as expensive as p_i. Therefore, by deviating with respect to p_i, player i can gain at most a factor of

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

We have given a polynomial-time algorithm which returns a $(1+\epsilon)$-approximate Nash equilibrium on a feasible graph G', where $\mathit{cost}(G') \leq \mathit{cost}(G_\alpha)$ for the Group Network Formation Game where all nodes are terminals.

The same algorithm and the proof holds for the Group Network Formation of Couples Game except that computing the cheapest deviation does not reduce to minimum spanning tree but to minimum Steiner tree problem and therefore, the best deviation of a player cannot be computed in polynomial time. But since the Steiner tree problem can be provably approximated within a factor of $1+\frac{\ln 3}{2} \approx 1.55$ by a polynomial algorithm[60], we will use the output of this algorithm instead of the cheapest deviation to decide the payment on the edges. Note that we obtain a network G' with $c(G') \leq c(G_\alpha)$ such that we have a 1.55-approximate Nash equilibrium on G' if the cost of its edges were decreased by γ in time polynomial in m and ϵ^{-1}. Similarly to the case when all nodes are terminals, the payment of each player will be increased by a factor of at most $(1+\epsilon)$ after the payments on the edges are raised proportionally to cover the whole cost of the edges of G'. Therefore, we have given a polynomial-time algorithm which returns a $(1.55+\epsilon)$-approximate Nash equilibrium on a feasible graph G', where $\mathit{cost}\,(G') \leq \mathit{cost}\,(G_\alpha)$ for the Group Network Formation of Couples Game.

Given an instance $\Im_1 = (N_1, G, T, F)$ of Group Network Formation Game, we can obtain an instance $\Im_2 = (N_2, G, T, F)$ of Group Network Formation of Couples Game that has twice as many players on the same graph G, with the same set of terminals T and the same happiness function F such that each player $i \in N_1$ has 2 corresponding players $j, k \in N_2$ as illustrated in the proof of Theorem 10. The strategy $p_i = p_j + p_k$ is $3.1(1+\epsilon)$-approximately stable for player i since the costs of both p_j and p_k are within a factor of $(1.55+\epsilon)$ of the cost of $\chi_i(p_i)$. ■

Since for all monotone functions F, finding OPT is a constrained forest problem [27], then Theorem 12 gives us a poly-time algorithm for $\alpha = 2$. In other words, we can find a $(3.1+\epsilon)$-approximate Nash equilibrium with cost at most $2 \cdot OPT$ in polynomial time.

3.5 Inapproximability Results and Terminal Backup

Recall that in this paper, we consider games where the happiness functions are monotone. Theorem 13 shows that this property of happiness functions is critical for even approximate stability.

Theorem 13 *For the Group Network Formation Game where the happiness functions may not be monotone, there is no α-approximate Nash equilibrium for any α.*

Proof. To prove that there is no approximate Nash equilibrium for the Group Network Formation Game we give such an instance of the problem as shown in Figure 3.4(A). All nodes are player nodes. We define a component to be happy if

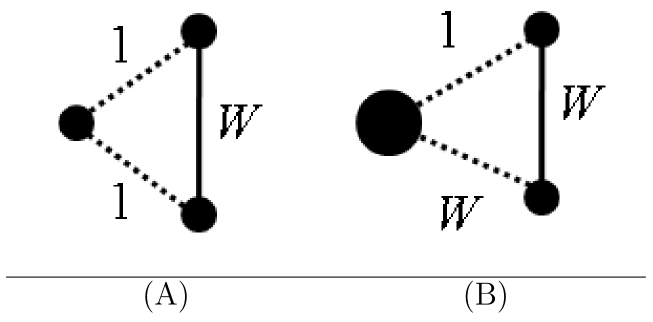

Figure 3.4: (A) An instance of a game where there is no approximate Nash equilibria if the happiness function is not monotone. (B) An instance of a Group Network Formation Game where there is no approximate Nash equilibrium on OPT if the fair sharing cost-sharing mechanism is used.

all the players in it are happy according to the following connectivity requirements: The player on the left is happy if it is connected to at most one other terminal. The 2 players on the right are happy if and only if they are connected to exactly 1 other terminal. The cost of each edge is as given in Figure 3.4(A), where W is very large. Observe that there is only one feasible solution of the game, i.e., all the players are happy and where only the edge whose cost is W is purchased. The player on the left cannot contribute to the cost of the edge in any approximate Nash equilibrium since it is already happy and therefore has a deviation of cost 0. Since the other 2 players have to be sharing the cost of W, at least one of them should be paying at least $W/2$. Since that player has a deviation of cost 1, at least one of the players can reduce its cost by a factor of $W/2$ by deviating and therefore there is no α-approximate Nash equilibrium for this game where $\alpha < W/2$. Since W can be arbitrarily large and it is independent of the number of players, there is no approximate Nash equilibrium for that instance of the general Group Network Formation Game. ■

Recall that congestion games, including our game with fair sharing, are guaranteed to have Nash equilibria, although they may be expensive. The following theorem studies the quality (cost) of approximate Nash equilibrium and shows that there may not be any approximately stable solution that is as cheap as the socially optimal network.

Theorem 14 *For the Group Network Formation Game, there may not be any approximate Nash equilibrium whose cost is as much as OPT if the fair cost-sharing mechanism is used.*

Proof. In Figure 3.4(B), all nodes are player nodes. We define a component to be happy if all the players in it are happy according to the following connectivity requirements: The large node on the left is happy always, and the 2 nodes on the right want to connect to at least one other terminal. In OPT, the 2 players on the right would have to buy the edge between them which has a cost of W. When the fair sharing cost scheme is used, both of the nodes have to make a payment of $W/2$ on that edge even though the top player has a deviation of cost 1. Since the top player can reduce its cost by a factor of $W/2$ by deviating, there is no α-approximate Nash equilibrium for this game where $\alpha < W/2$. Since W can be arbitrarily large and it is independent of the number of players, there is no approximate Nash equilibrium on OPT. ■

Because of its applications to multi-homing [7, 59], we are especially interested in the behavior of Terminal Backup connectivity requirements, i.e., when a player node desires to connect to at least $k-1$ other player nodes for a fixed k.

Theorem 15 *For the Group Network Formation Game and the Terminal Backup problem, the Price of Anarchy is n and $2k-2$ respectively. Furthermore, these bounds are tight.*

Proof. The *price of anarchy* refers to the ratio of the worst (most expensive) Nash equilibrium and the optimal centralized solution. In the Group Network Formation 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 form a connected component that makes her happy with smaller payments than before. More importantly, there are cases when the price of anarchy actually equals N. This is demonstrated with the example in Figure 3.5(A).

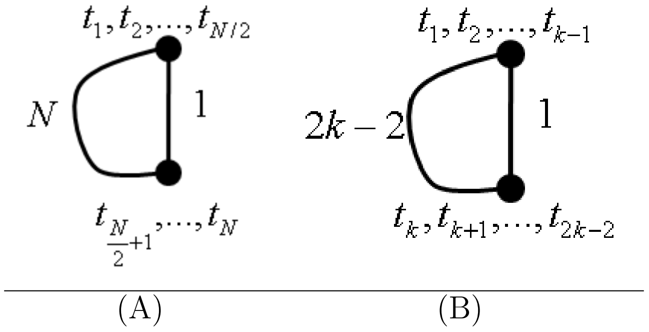
$t_1, t_2, \ldots, t_{N/2}$
N
1
$t_{\frac{N}{2}+1}, \ldots, t_N$
(A)
$t_1, t_2, \ldots, t_{k-1}$
$2k-2$
1
$t_k, t_{k+1}, \ldots, t_{2k-2}$
(B)

Figure 3.5: (A) An instance of a Group Network Formation Game that has a Nash equilibrium whose cost is N times more than the socially optimal network. (B) An instance of Terminal Backup problem that has a Nash equilibrium whose cost is $2k-2$ times more than the socially optimal network.

Suppose there are N players, and G consists of 2 nodes which are joined by 2 disjoint paths, one of cost 1 and and one of cost N. Half of the players have their terminal at one node and the other half of the players have their terminal at the other node. The only happy components must include both nodes. This corresponds to Terminal backup requirements where every player wants to be connected to at least $N/2+1$ terminals. 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 Group Network Formation Game.

We are also interested in the price of anarchy of the Terminal Backup problem which is a special case of Group Network Formation Games. In Figure 3.5(B), we give an example where price of anarchy is $2k-2$ and we prove that this bound is tight below. In this case, k is the connectivity requirement: components are happy if and only if they contain at least k terminal nodes.

To prove the result all we need to do is to show that no equilibrium will cost more than $2k-2$ times OPT since Figure 3.5(B) shows an instance of a Terminal Backup problem with an equilibrium whose cost is exactly $2k-2$ times the cost of OPT.

For the purpose of contradiction, assume there is an instance of a Terminal

Backup problem that has an equilibrium whose cost is more than $2k-2$ times the cost of OPT. Let EEQ denote this expensive equilibrium solution. Observe that there must exist a connected component T of OPT such that the total payments of the players of T in the expensive equilibrium solution EEQ is more than $2k-2$ times the cost of T. Since EEQ is a Nash equilibrium, no player in T pays more than the cost of T. Therefore, T includes more than $2k-2$ terminals. However, as we have shown in the proof of Theorem 16, there exists a set of edges C_i for every player i such that $C_i \subseteq T$, C_i contains at least k terminals including i, and every edge of T is contained in no more than $2k-2$ sets C_i.

We define α_i be the cost of C_i, which is at least the cost of connecting a player i in T to at least $k-1$ other players in T. Since no edge of T is used by more than $2k-2$ sets C_i, we know that $\sum_{i \in T} \alpha_i \leq (2k-2)\, c(T)$ where $c(T)$ denotes the cost of the component T.

However, in EEQ, the total payment of all the players in T is more than $2k-2$ times the cost of T, i.e., $\sum_{i \in T} |p_i| > (2k-2)\, c(T)$. Therefore, $\sum_{i \in T} |p_i| > \sum_{i \in T} \alpha_i$ which implies that there exists a player i in T such that $|p_i| > \alpha_i$. Therefore, EEQ cannot be a Nash equilibrium since player i can reduce its cost to α_i by deviating.

The lower bounds for Terminal Backup also hold for the general Group Network Formation Game, showing that while the price of stability may be low, the price of anarchy can be as high as the number of players. ■

Theorem 16 *For the Terminal Backup problem, with the Shapley cost-sharing (fair sharing) payment scheme, the price of stability is at most $H(2k-2)$ where $H(2k-2)$ denotes the $(2k-2)^{nd}$ harmonic number.*

Proof. The Terminal Backup problem, under the Shapley cost-sharing model (or fair sharing), falls into a class of games called congestion games. In this model, the strategy of a player i is just a set of edges S_i that contains at least k terminals including i, and the edges that are built are $\cup_i S_i$. The players split the cost of every edge evenly, i.e., a player i using edge $e \in S_i$ must pay $c(e)/x_e$, where $x_e = |\{S_i | e \in S_i\}|$. [5] showed that in a network design game with the Shapley-cost sharing mechanism, the cost of the Nash equilibrium obtained by the best-response dynamics starting from OPT is at most $H(n)$ times more expensive than OPT, where $H(n)$ is the n^{th} harmonic number and n is the maximum number of players using a single edge on OPT for their connectivity requirements. Therefore, to obtain a price of stability bound for the Terminal Backup problem, all we need to do is to select sets S_i for the players in OPT while ensuring that no edge appears in too many sets S_i.

To prove the result, we need to show that we can select sets S_i for the players on OPT such that no edge is used by more than $2k-2$ players. We first start by replacing all the edges of OPT by 2 directed edges. We can now form an Euler tour in each connected component of OPT since every vertex has an equal number of incoming and outgoing edges incident to it. Then, we ask each player i to follow the Euler tour in the clockwise direction until it encounters $k-1$ other distinct terminals. We set S_i to be this set of edges. Observe that each directed edge appears in at most $k-1$ sets S_i belonging to distinct closest players in the counterclockwise direction. Since each undirected edge of OPT had been replaced with 2 directed edges, each edge of OPT is used by at most $2k-2$ sets S_i. ■

CHAPTER 4
STRATEGIC CUT GAMES

4.1 Model and Preliminaries

We now formally define the *Network Cutting Game* as follows. We are given an undirected graph $G = (V, E)$, and a set P of k players. Each player i corresponds to a single *player node* in the graph G, which we will also denote by i. The strategy set of every player i is 2^E; a strategy $S_i \subseteq E$ of player i is the set of edges that player i will cut. The outcome of the game is G_S, which is a subgraph of G obtained by removing the edges of $\bigcup_i S_i$.

The objective of each player i is to protect her node i from a given subset of nodes T_i of V. We say that player i *satisfies her cut requirement* if i is disconnected from all nodes of T_i in G_S. Every player wants to satisfy her cut requirement, but also wants to minimize the number of edges she cuts, which we denote by $|S_i|$. If a player i does not satisfy her cut requirement, she faces a penalty cost of β_i. We can think of β_i as the maximum number of edges that i would be willing to cut in order to satisfy her cut requirement. We conclude the definition of our game by defining the cost function for each player i as:

- $cost(i) = |S_i|$ if player i satisfies her cut requirements,
- $cost(i) = |S_i| + \beta_i$ otherwise.

Nash Equilibrium and OPT A pure Nash equilibrium (NE) of the *Network Cutting Game* is a strategy vector $S = (S_1, \ldots, S_k)$ such that no player i has an incentive for unilateral deviation from her strategy, i.e., no player can reduce her cost by changing her strategy from S_i to another strategy S'_i, assuming all other players stay with their existing strategies. Notice that in an equilibrium no player will cut more than β_i edges, since this player could change her strategy to $S'_i = \emptyset$, and reduce her cost to at most β_i. By the same reasoning, all players that do not satisfy their cut requirements will play $S_i = \emptyset$ at equilibrium. Therefore, in a Nash equilibrium, all edges must be cut by players that satisfy their cut requirements.

We analyze the quality of Nash equilibrium solutions by comparing them with the cost of the socially optimal solution, which we refer to as OPT. The socially optimal solution is an outcome of the *Network Cutting Game* that minimizes the total cost of all the players (equivalently, maximizes social welfare). Let $Q(S)$ denote the set of players whose cut requirements are not satisfied in G_S. The cost of solution S is given by:

$$\text{cost}(S) = \left|\bigcup_i S_i\right| + \sum_{j \in Q(S)} \beta_j,$$

which is the same as $\sum_i cost(i)$ since for any equilibrium solution we can assume that all sets S_i are disjoint. When all β_i are large, OPT is exactly the smallest set of edges that satisfies all the cut requirements.

Given a graph $G = (V, E)$, and two disjoint subsets $A \subseteq V$ and $B \subseteq V$, denote by $M(G, A, B)$ the set of minimum-size $A - B$ cuts. By an $A - B$ cut, we mean a set of edges E' such that A and B are disconnected by removing E' from G. Denote by $m(G, A, B)$ the size of a minimum $A - B$ cut in graph G. We will also abuse notation slightly, and for a strategy vector $S = (S_1, \ldots, S_k)$, we will let S_{-i} denote $\bigcup_{j \neq i} S_j$, i.e., the set of edges cut by players other than i.

Proposition 2 *A strategy vector $S = (S_1, \ldots, S_k)$ of the Network Cutting Game is a Nash equilibrium if and only if S_i are pairwise disjoint, and*

- $|S_i| = m(G - S_{-i}, i, T_i) \leq \beta_i$ *if cut requirement of player i is satisfied on G_S,*
- $S_i = \emptyset$ *and* $m(G - S_{-i}, i, T_i) \geq \beta_i$ *otherwise.*

Proof. It is easy to see that all S_i must be disjoint, since otherwise a player could reduce her cost without altering the outcome G_S.

Consider the best response of player i to the strategy vector S. The minimum number of edges that i must cut in order to fulfill its cut requirements is exactly $m(G - S_{-i}, i, T_i)$. Therefore, the cost of i's best response is exactly $\min\{m(G - S_{-i}, i, T_i), \beta_i\}$. This is because if $\beta_i < m(G - S_{-i}, i, T_i)$, then player i could achieve smaller cost by cutting nothing and incurring a cost of β_i.

Let i be a player such that G_S satisfies the cut requirement of i. First consider the case where $m(G - S_{-i}, i, T_i) \leq \beta_i$. Then, player i will be stable in solution S if and only if $|S_i| = m(G - S_{-i}, i, T_i)$, as desired. If instead $\beta_i < m(G - S_{-i}, i, T_i)$, then S cannot possibly be a Nash equilibrium, since for i's cut requirements to be satisfied in S, it must be that $|S_i| \geq m(G - S_{-i}, i, T_i) \geq \beta_i$, which means that player i could switch her strategy to $S_i' = \emptyset$ and reduce her cost to at most β_i.

Now, let i be a player such that G_S does not satisfy their cut requirement. As argues above, i will not cut any of the edges of G in an equilibrium, so it must be that $S_i = \emptyset$. Moreover, i is stable in solution S if and only if $m(G - S_{-i}, i, T_i) \geq \beta_i$, since otherwise player i can reduce her cost to $m(G - S_{-i}, i, T_i)$ from β_i by changing her strategy from S_i to $S_i' \in M(G - S_{-i}, i, T_i)$. ■

To add intuition about the structure of Nash equilibria, notice that when β_i are large (say $\geq |E|$) for all players, then the above proposition says that a solution S is a Nash equilibrium exactly when $|S_i| = m(G - S_{-i}, i, T_i)$ for all players.

4.2 Single-Source Network Cutting Game

In this section, we study the *Single Source Network Cutting Game*, which is a special case of the Network Cutting Game, where each player i wants to cut her node from a common node t, i.e., $T_i = \{t\}$ for all players i. While the results and arguments in this section are not difficult, they introduce some basic ideas and techniques that will be greatly expanded on in the later sections. This analysis easily generalizes to the case when T_i are not singleton sets, but $T_i = T_j$ for all i, j.

Theorem 17 *For the Single Source Cutting Game, Nash equilibrium is guaranteed to exist and the price of stability is* 1.

Proof. We show that there exists an assignment of the edges in OPT to the players such that it is a Nash Equilibrium. Add a super-source s to the input graph G, so that s is connected only to each player node $i \in P$. If we set the capacities of edges (s, i) to be β_i, and the capacities of all other edges to be 1, then it is easy to see that OPT is an $s - t$ min-cut in this new graph. More precisely, if we let S^* be the set of edges in OPT in the original graph G, and Q^* be the set of players that are still connected to t in OPT, then $S^* \cup \{(s,i)|i \in Q^*\}$ is a minimum s-t cut. This is simply because if we take a set of edges S in G so that the players Q are still connected to t after removing S, then the cost of the s-t cut $S \cup \{(s,i)|i \in Q\}$ is exactly the social cost of solution S.

Let the set of edges in this s-t min-cut be $M = S^* \cup \{(s,i)|i \in Q^*\}$ and let $|M|$ denote the cardinality of M. This also means that an integral flow of size $|M|$ can be pushed from s to t saturating all edges of M, since all capacities are integral (we can assume that all β_i are integral without loss of generality). The assignment of edges of S^* to the players is as follows. Note that we only assign edges to terminals i if M did not contain the edge of weight β_i. Now consider the flow leaving the super-terminal s. Because of the construction, all of this flow passes through the terminal nodes. Since all the flows are integral we can decompose them into unit flows. Now we can categorize these unit flows depending upon the terminals they pass through. A flow passing through edge (s,i) is marked as f_i. Since all edges of the original graph G are of unit capacities, any edge $e \in S^*$ will receive flow from exactly one terminal in $P \backslash Q^*$. We set S_i to be the set of edges of S^* that receive a flow f_i. We now show that such an assignment results in a NE.

We use Proposition 2 to show that the resulting strategy vector $S = (S_1, \ldots, S_k)$ is a Nash equilibrium. First consider a player $i \in Q^*$, i.e., a player that is connected to t in $G - S^*$. We did not assign it to pay for any edges of OPT, so $S_i = \emptyset$, as desired. Now suppose to the contrary that $m(G - S_{-i}, i, T_i) = m(G - S^*, i, t) < \beta_i$. Then consider the set of edges $S^* \cup C$, for $C \in M(G - S^*, i, t)$. The social cost of this set of edges is at most $|S^*| + |C| + \sum_{j \in Q^* \backslash i} \beta_j$ which is cheaper than OPT and hence a contradiction. Therefore, by Proposition 2, i will not deviate in solution S.

Now consider a player $i \notin Q^*$. To show that it will not deviate, we must prove that $|S_i| = m(G - S_{-i}, i, T_i) \leq \beta_i$. Consider $m(G - S_{-i}, i, t)$. This is the size of the maximum flow that can be sent from i to t in the graph $G - S^* + S_i$. Since M is a min-cut, and thus saturated by a maximum flow, we know that the flow f_i does not use any edges of $M - S_i$, and so it is still a valid i-t flow in $G - S^* + S_i$. Therefore, $|S_i| \leq m(G - S_{-i}, i, t)$. On the other hand, S_i is a i-t cut in the graph $G - S^* + S_i$, since S^* is an i-t cut in G. Therefore, $|S_i| \geq m(G - S_{-i}, i, t)$, as desired. Finally, notice that $|S_i| \leq \beta_i$, since $|S_i|$ is the size of a flow passing through edge (s, i), and the capacity of that edge is β_i. Therefore, by Proposition 2, i will not deviate in solution S. Thus this assignment results in a NE and the price of stability is 1. ■

Notice that the equilibrium constructed in the proof of Theorem 17 can be easily found in poly-time using standard flow algorithms.

4.3 Network Multiway Cut Game

In the Network Multiway Cut Game (NMCG) each player i wants to disconnect itself from every other player provided that the cost of cutting edges does not exceed the player's budget β_i. As mentioned earlier, when β_i for every player is very large the socially optimal solution is the well-studied Multiway Cut (MWC) problem. For the sake of simplicity in exposition, we will derive the main results assuming β_i to be infinitely large. We will later extend these results to case where values of β_i are bounded. Notice that in this scenario where $\beta_i = \infty$, each player ***must*** disconnect itself from every other player node. In order to prove results about price of stability of the Network Multiway Cut Game, we make use of the following observations.

4.3.1 Properties and Terminology

The problem input consists of graph $G = (V, E)$, and a set of k terminals/players $T \subseteq V$. For every player i, $T_i = T \backslash i$. We define the following terms with respect to an instance of a multiway cut problem (G, T). Any valid MWC X must divide G into at least k components. Let the component containing terminal i be termed as $C_i(X)$. We define $\delta_i(X)$ as the set of edges (u, v) such that $|C_i(X) \cap \{u, v\}| = 1$ and $\delta_{ij}(X)$ as the set of edges (u, v) such that $u \in C_i(X)$ and $v \in C_j(X)$. Terminal j is a neighbor of terminal i (denoted by $j \in N_i(X)$) if $\delta_{ij}(X) \neq \emptyset$. Let $G_i(X)$ be the subgraph defined as follows: $G_i(X) = G - X + \delta_i(X)$.

Let OPT be the set of edges in the optimal Multiway Cut. For the sake of brevity, when defined for OPT we will refer to the above terms simply as C_i, δ_i, δ_{ij}, N_i and G_i respectively. It is well known that when an optimal MWC is made, the graph G is divided into exactly k components each containing one terminal. Also, given a graph G, let $E[G]$ correspond to the set of edges of G.

Lemma 21 *Recall that $M(G_i, \{i\}, N_i)$ is the set of minimum edge cuts between terminal i and its neighboring terminals in graph G_i. Then, there exists $M \in M(G_i, \{i\}, N_i)$, such that the set of edges in M is a subset of $E[C_i] \cup \delta_i$.*

Proof. Consider $M_i' \in M(G_i, \{i\}, N_i)$ to be a cut such that it is not a subset of $E[C_i] \cup \delta_i$. Since M_i' does not completely lie in the subgraph $C_i \cup \delta_i$, there is at least one $j \in N_i$ such that $M_i' \cap E[C_j] \neq \emptyset$. Consider such a terminal j. Let M_i' divide the graph G_i into W_i and W_j such that W_i contains terminal i and W_j contains terminal j (see Figure 4.1). Let set $A = \delta_{ij} \cap E[W_i]$ be the set of edges of δ_{ij} that lie in side W_i, and set $B = M_i' \cap E[C_j]$ be the set of edges of M_i' that lie in the component C_j. We make the following observations on graph G_i.

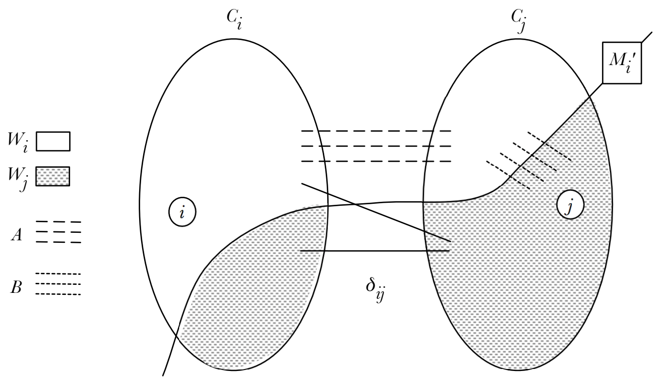

Figure 4.1: Illustration of the Cuts and Components

Claim 1 *The set of edges $Y = (M_i' \backslash B) \cup A$ is a valid $(\{i\}, N_i)$ cut in G_i.*

Proof. Consider an arbitrary path p from j to i in G_i. As we trace this path starting from terminal j, consider the edge e when p enters the component W_i for the last time. By definition $e \in M_i'$.

Case 1: If e belongs to the set $M_i' \backslash B$ then e also belongs to Y. This means that all such paths p will be cut by Y.

Case 2: If e belongs to B then the path p enters the component $C_j \cap W_i$. Since there is no edge in G_i between the components C_j and C_d for $d \in N_i \backslash j$ and path p

does not enter side W_j again, then the path has to travel across an edge of set A in order to reach terminal i. But $A \subseteq Y$ and hence all such paths p will also be cut by Y.

Therefore all (i, j)-paths are cut by Y. Let d be a non-j neighbor of i (i.e. $d \in N_i \backslash j$). We still need to prove that Y also cuts all (i, d)-paths. As we trace a path starting from terminal d, consider the edge e when it enters the component W_i for the last time. Then the two cases considered above hold in the same fashion thus proving that Y is a valid (i, N_i) cut. ■

By a symmetric argument we can also prove the following claim.

Claim 2 *The set of edges $OPT' = (OPT \backslash A) \cup B$ is a valid multiway cut for the problem instance (G, T).*

Let us now compare the size of the sets A and B. If $|A| < |B|$ then by using Claim 1 we can construct a $(\{i\}, N_i)$ cut that is smaller in size than M_i'. This contradicts our assumption that M_i' is a minimum $(\{i\}, N_i)$ cut. If $|B| < |A|$ then by using Claim 2 we can construct a valid multiway cut for the instance (G, T) that is smaller in size than OPT which is also clearly a contradiction.

These observations lead us to the conclusion that $|A| = |B|$. As described in Claim 1, replacing the edges of B with A in the cut M_i' gives a valid (i, j) cut. And since $|A| = |B|$, the size of the new cut is same as the original cut. This procedure can be applied to all $j \in N_i$ thereby resulting in a cut which belongs to the set $M(G_i, \{i\}, N_i)$ and also lies in the subgraph $C_i \cup \delta_i$. ■

Following is a useful observation about the optimal solution of the Multiway Cut.

Observation 2 *If OPT is the optimal multiway cut for a graph G with terminals T, then for any $S \subseteq OPT$, $OPT - S$ is the optimal multiway cut for the graph $G - S$ with terminals T.*

Proof. Let $OPT_S \neq OPT - S$ be the optimal multiway cut in the graph $G - S$ with terminals T such that $|OPT_S| < |OPT - S|$. Now consider the multiway cut

$OPT_S + S$ for the original multiway cut problem (G, T). Since OPT_S is a valid multiway cut for $(G - S, T)$,is easy to see that OPT_S will also be a valid multiway cut. It is also obviously smaller in size than OPT. But this is a contradiction, hence no such OPT_S exists. ■

4.3.2 Price of Stability

Theorem 18 *The price of stability for the Network Multiway Cut problem is* 1*, i.e. there exists a payment strategy for the socially optimal solution that forms a Nash Equilibrium.*

In order to prove the theorem we make use of the following construction. For a multiway cut X of an instance (G, T), we construct a directed flow graph $F(G, T, X)$ as follows: Construct source node s and sink node t. For every edge $e \in X$, construct vertices m_e, n_e and a directed edge (m_e, n_e) of capacity 1. Construct a directed edge (n_e, t) of capacity ∞ for every such n_e. Add components $C_i(X)$ to this graph and make all of their edges bi-directed with capacity 1. Add an edge (s, t_i) for every terminal i with capacity ∞. For every vertex $v \in C_i(X)$ that has an edge e of $\delta_i(X)$ incident on it, construct a directed edge (v, m_e) with capacity ∞.

Lemma 22 *Any bounded $s - t$ cut in $F(G, T, X)$ corresponds to a valid multiway cut for (G, T) of the same size.*

Proof. First of all it should be clear that there always exists a bounded $s - t$ cut since X has bounded size and is a valid $s - t$ cut. Observe that all edges of $F(G, T, X)$ with bounded capacities are present in the original graph G. This means that all edges of a bounded $s-t$ cut in $F(G, T, X)$ can be mapped to original graph G. In $F(G, T, X)$ consider a terminal i and edge $e = (u, v) \in \delta_i(X)$. Since the edges (s, i) and (v, t) have infinite capacities, any bounded $s-t$ cut will remove at least one edge from the path (i, v). When mapped to the original graph G, this observation implies that for any terminal i and any $e(u, v) \in \delta_i(X)$, every (i, v) path will be cut. Since for any two terminals i, j, an (i, j) path traverses through at least one edge in $\delta_i(X)$, all (i, j) paths will be cut by the bounded $s - t$ cut on $F(G, T, X)$. ■

If $X = OPT$, then from the construction we can see that the edges representing OPT in $F(G, T, OPT)$ form a bounded $s - t$ cut of the same size. If the minimum $s-t$ cut on $F(G, T, OPT)$ is smaller than size of OPT then Lemma 22 would imply that there exists a MWC of size smaller than OPT, which is a contradiction. So the minimum bounded $s - t$ on $F(G, T, OPT)$ has to be of the same size as OPT. This means that the maximum $s - t$ flow in $F(G, T, OPT)$ will saturate edges of OPT. Then the assignment of edges S_i to terminal/player i can be made as shown in Algorithm 3.

```
**Input**: Flow Graph $F(G, T, OPT)$
**Output**: Assignment of edges to players
Mark the flows that originate through terminal $i$ as $f_i$;
Assign all edges of $OPT$ that carry flows marked $f_i$ to $S_i$;
```

Algorithm 3: Algorithm that assigns edges of OPT to players

Since OPT forms the minimum bounded $s - t$ cut, we know that all edges of OPT will be assigned.

Lemma 23 *The assignment made by Algorithm 3 forms a Nash Equilibrium.*

Proof. We first prove this for the case where all β_i are large; see Section 4.3.4 for the extension of this result to arbitrary β values. Consider the graph $G'_i = G-OPT+S_i$. It should be noted that the best response for player i, given a strategy decided by the above algorithm, will be a cut $M \in M(G'_i, \{i\}, N_i)$. The assignment algorithm chooses S_i that is a subset of OPT and consequently Observation 2 tells us that S_i will be an optimal solution for the MWC instance (G'_i, T). Let δ_i and C_i defined for the MWC instance (G'_i, T) be represented as $\delta_i(G'_i, T)$ and $C_i(G'_i, T)$ respectively. It should be clear that $\delta_i(G'_i, T)$ is in fact the set S_i and so $C_i(G'_i, T)$ is same as original component C_i of OPT. Using Lemma 21, we can now show that there exists an $M \in M(G'_i, \{i\}, N_i)$ which lies in the subgraph $\delta_i(G'_i, T) \cup C_i(G'_i, T)$. But according to the assignment algorithm there exists a flow of size $|S_i|$ from terminal i to δ_i in the subgraph $C_i \cup S_i$. Hence $m(G'_i, \{i\}, N_i)$ is at least $|S_i|$. It follows that if player i chooses to deviate then it will have to pay at least $|S_i|$ to disconnect itself. Therefore the assignment forms a Nash Equilibrium. ■

4.3.3 Poly-time computable Nash Equilibrium

Our algorithm in the previous section depends on the knowledge of the optimal Multiway cut for a given problem. Since this is often computationally infeasible due to the NP hardness of the Multiway Cut problem, we give an algorithm that efficiently computes a Nash equilibrium whose cost is no larger than any given Multiway Cut. For example, we can begin with a $3/2-$approximate MWC solution that can be obtained in polynomial time [13], and using the following lemma obtain a Nash equilibrium that is no more expensive than this solution.

Lemma 24 *For any valid multiway cut X for the problem instance (G, T), we can form a NE payment strategy for X given that the following two conditions hold:*

(i.) Size of the minimum $s-t$ cut in flow graph $F(G, T, X)$ is same as the size of X.

(ii.) If $S(X)$ is the payment strategy devised by Algorithm 3 on X, then for every terminal i there exists a cut $M \in M(G - S_{-i}(X), i, N_i(X))$ such that M lies completely within the component $C_i(X) \bigcup S_i(X)$.

Proof. If the first condition holds true then we know from the proof of Theorem 18 that Algorithm 3 will be able to devise a payment strategy for all edges of X. The size of the best response strategy of terminal i for strategy $S(X)$ is $m(G - S_{-i}(X), i, N_i(X))$. But because there exists a flow of size $|S_i(X)|$ from terminal i to $S_i(X)$ within the component $C_i(X) \bigcup S_i(X)$ and given the fact that the second condition holds, it follows that $|S_i(X)| = m(G - S_{-i}(X), i, N_i(X))$. Therefore the payment strategy will form a Nash Equilibrium. ■

The following algorithm can now be used to form a cheap Nash equilibrium solution in polynomial time.

1. Check for condition $(i.)$

   1.1 If $(i.)$ does not hold then find a smaller MWC X and go to step 1.

2. Check for condition $(ii.)$

2.1 If (*ii.*) does not hold then find a smaller MWC X and go to step 1.

3. Since both conditions of Lemma 24 hold, use Algorithm 3 on MWC X to form a Nash equilibrium

To finish the algorithm, we must show how to efficiently create a smaller Multiway Cut when one of the conditions in Lemma 24 does not hold. If condition (*i.*) does not hold true for X, then we can use Lemma 22 to construct a valid MWC which is smaller in size than X. Condition (*ii.*) can be checked in polynomial time using standard flow methods. If condition (*ii.*) does not hold then we can use the same arguments as in Claim 2 to construct another MWC which is smaller in size than X. When both conditions are satisfied, we know from Lemma 24 that a NE strategy can be obtained.

Every time a new MWC is constructed by the algorithm, its size is smaller than the existing cut. This means that the algorithm can consider at most $|E|$ MWCs. Since for every MWC it takes polynomial time to compute a new smaller MWC or to find the final assignment, the algorithm terminates in polynomial time. This completes the proof of the following theorem:

Theorem 19 *Given a multiway cut X, we can find a Nash equilibrium that is cheaper than X in polynomial time.*

4.3.4 Bounded values of β_i

We now extend the results of the previous sections to the case where the values of β_i are arbitrary. The socially optimal solution becomes a *prize-collecting* version of the Network Multiway Cut Game (NMCG) where no player experiences a cost more than its budget. A player will not pay for cutting any edge in a payment strategy which does not satisfy its cut requirement. It will instead incur a cost of β_i for not being isolated. Given a socially optimal solution for the NMCG, let R be the set of players whose cut requirement is fulfilled. It is easy to extend Lemma 21 for all $i \in R$. Observation 2 also holds in this scenario. Notice that the optimal solution still divides the graph into at most k components. Players belonging to R

do not share a component with any other player whereas those players that are not isolated share a component with one or more players like them.

In order to prove Theorem 18 for this general case we need to make some changes to the construction of the flow graph $F(G, T, X)$. Instead of adding all components we add only those components that contain a player belonging to R. Also, instead of assigning edge (s, i) infinite capacity, we give it capacity β_i. Now any valid $s - t$ cut in this new flow graph can be mapped to a valid solution for the NMCG problem of the same size, since no $s - t$ cut will cost more than $\sum_{i \in R} \beta_i$. An edge (s, i) being cut corresponds to player i incurring a cost of β_i because its cut requirements are not fulfilled. Hence when the flow graph is constructed for the optimal solution: $F(G, T, OPT)$, the min $s - t$ cut is the same size as OPT, because otherwise we would be able construct a cheaper OPT. We then proceed to assign edges to players belonging to set R according to Algorithm 3. Since Observation 2 and Lemma 21 still hold for this general case, we can use essentially the same argument as in Lemma 23 to prove that the payment strategy produced by Algorithm 3 forms a Nash Equilibrium. The arguments in Section 4.3.3 can similarly be extended.

4.4 Network Multicut Game

In the *Network Multicut Game* each player i wants to be cut from a particular node t_i of G provided that the cost of the strategy S_i of player i does not exceed β_i. Recall that Network Multicut Game is a special case of the Network Cutting Game, where $T_i = \{t_i\}$ is singleton for every player. As pointed out before, the socially optimal solution is the minimum cost multicut for the pairs $\{(1, t_1), (2, t_2), \ldots, (k, t_k)\}$ if β_i values are large enough. In general, OPT is a solution that minimizes the total cost of all the players.

For the Network Multicut game, we don't know whether there always exists a Nash equilibrium or not. In this section, we first give some properties of Nash equilibria that are as cheap as OPT for the Network Multicut Game, which enables us to give a simple greedy algorithm that returns a 2-approximate Nash equilibrium as cheap as OPT.

Let C_j and C_k be arbitrary two components of OPT and let δ_{jk} denote the set of edges of G that are between C_j and C_k. Since C_j and C_k are cut apart in OPT, all the edges of δ_{jk} are cut in OPT. We call that C_j and C_k are neighbor components if $\delta_{jk} \neq \emptyset$.

Lemma 25 *For any 2 neighbor components C_j and C_k of OPT, there exists a player i such that either $i \in C_j$ and $t_i \in C_k$ or vice versa.*

Proof. For the purpose of contradiction, assume the contrary of the statement, i.e., there exists neighbor components C_j and C_k of OPT such that there is no player i for which either $i \in C_j$ and $t_i \in C_k$ or vice versa. But then one can obtain a new solution that cuts less edges than OPT by simply merging the components C_j and C_k. Notice that any player l that is cut from t_l in OPT is also cut from t_l in the new solution. Therefore, the new solution is cheaper than OPT. ■

Lemma 26 *Let i be a player such that $i \in C_j$ and $t_i \in C_k$ for 2 arbitrary neighbor components C_j and C_k. Then player i cannot cut any edge that is not in δ_{jk} in any Nash equilibrium that cuts OPT.*

Proof. For the purpose of contradiction, assume player i cuts an edge $e \notin \delta_{jk}$ as part of her strategy S_i in a Nash equilibrium solution (S_i, S_{-i}) that cuts OPT. Assume player i unilaterally deviates from her strategy S_i to a different strategy $S'_i = S_i - e$ of her, i.e., she cuts exactly the same set of edges in S_i but e. Notice that in the solution (S'_i, S_{-i}), i and t_i are still cut apart and player i cuts less edges than she does in (S_i, S_{-i}) and therefore, (S_i, S_{-i}) is not a Nash equilibrium solution. ■

Corollary 2 *Let i be a player such that $i \in C_j$ and $t_i \in C_k$ for 2 arbitrary components C_j and C_k of OPT that are not neighbors. Then player i is a free-rider in any Nash equilibrium solution that cuts OPT (i.e., cost$(i) = 0$).*

To prove the existence of a 2-approximate Nash equilibrium as cheap as OPT, we give an algorithm that assigns all the edges of OPT to the players and prove that no player can reduce her cost by more than half by unilaterally deviating from the strategy where she cuts the edges assigned to her by the algorithm. If a player i is not cut from t_i in OPT, then our algorithm does not assign any edge to player i, i.e., $S_i = \emptyset$. If i and t_i are in different connected components of OPT, say C_j and C_k respectively, then our algorithm assigns a subset of δ_{jk} to player i. Notice that if C_j and C_k are not neighbor components then $S_i = \emptyset$.

A player i may have an incentive for unilateral deviation from her strategy S_i only if i and t_i are in different neighboring connected components of OPT because of Lemma 27. Let $G - S_{-i}$ denote the subgraph of G where the edges cut by other players are removed and let i be a player such that $0 < |S_i| \leq \beta_i$. Observe that a best deviation of player i from her strategy S_i, which is denoted by $\chi_i(S_i)$, is a cheapest strategy that cuts i from t_i in $G - S_{-i}$. The cost of the best deviation of player i from strategy S_i, i.e., $|\chi_i(S_i)|$, is as much as $m(G - S_{-i}, i, t_i)$. Let G_{jk} be the subgraph of G, which is composed of the connected components C_j and C_k of OPT and edges δ_{jk}. Since player i does not cut any edge of OPT that is not an element of δ_{jk}, then $OPT - \delta_{jk} \subset S_{-i}$ and therefore, $m(G - S_{-i}, i, t_i) = m(G_{jk}, i, t_i) = |\chi_i(S_i)|$.

Lemma 27 *If a player i is not assigned any edge by the algorithm, i.e., $S_i = \emptyset$, then i cannot reduce her cost by unilateral deviation.*

Proof. The statement is trivially true if i and t_i are in different connected components of OPT, since $cost(i) = 0$. Therefore, assume both i and t_i are in the same connected component C_j of OPT and $cost(i) = \beta_i$. If $m(C_j, i, t_i) \geq \beta_i$ player i cannot reduce her cost by unilateral deviation since she has to cut at least $m(C_j, i, t_i)$ to satisfy her cut requirement. If $m(C_j, i, t_i) < \beta_i$, player i can reduce her cost by playing the strategy S_i' that cuts the edges of a minimum-size $i - t_i$ cut instead of strategy $S_i = \emptyset$. Observe that (S_i', S_{-i}) satisfies the cut requirements of all the players whose cut requirement is satisfied in (S_i, S_{-i}). Therefore, no player's cost will increase if player i deviates to S_i', which contradicts with OPT being the socially optimal solution. ■

Algorithm Let L_{jk} be the set of players i such that either $i \in C_j$ and $t_i \in C_k$ or vice versa. Without loss of generality let $L_{jk} = \{1, 2, ..., |L_{jk}|\}$. Recall that $L_{jk} \neq \emptyset$ by Lemma 25. Notice that the socially optimal solution of the Network Multicut

Game for players L_{jk} on G_{jk} (which we denote by $OPT(G_{jk}, L_{jk})$) is δ_{jk}. This is by the same argument as in Observation 2. For every δ_{jk}, we make one pass over the players L_{jk}. For player 1 of L_{jk}, we send a max-flow from node 1 to t_1 on G_{jk}(If the size of the max-flow is more than β_1 then we just send an arbitrary flow of size β_1). Let m_1 denote the subset of edges of δ_{jk} that are used by that flow. Notice that $|m_1| = \min\{m(G_{jk}, 1, t_1), \beta_1\}$. The algorithm asks player 1 to cut the edges of m_1, i.e., sets $S_1 = m_1$ and proceeds with player 2. We send a max-flow from 2 to t_2 on $G_{jk} - m_1$(Similarly, if the size of the max-flow is more than β_2 then we just send an arbitrary flow of size β_2) and ask player 2 to cut the edges m_2, the subset of $\delta_{jk} - m_1$ that are used by that flow and so on.

Let M denote the subset of the edges of δ_{jk} that are cut by the players at the end of the one pass described in detail above, i.e., $M = \bigcup_{i \in L_{jk}} m_i$. Notice that if $M = \delta_{jk}$ for all neighbor components C_j and C_k then the above algorithm will return a Nash equilibrium. This is because $|m_i| = m(G - OPT + m_i, i, t_i)$ for all i since the flow we used to construct m_i does not use any edges of OPT except m_i, and thus $|m_i| \leq \min\{m(G - S_{-i}, i, t_i), \beta_i\}$ which by Proposition 2 implies that i is stable.

However, it may be that $M \neq \delta_{jk}$ and so the greedy algorithm given above does not always return a Nash equilibrium. Fortunately, we know that $|M| \geq |\delta_{jk}|/2$ by Lemma 28. We next assign the remaining edges of δ_{jk} to the players L_{jk} proportionally to the number of edges they are assigned so far. Since $|M| \geq |\delta_{jk}|/2$, then any player $i \in L_{jk}$ which has been assigned $|m_i|$ edges is now assigned at most $|m_i|$ extra edges. The assignment given by the algorithm is a 2-approximate Nash equilibrium, since the cost of the best deviation of each player $i \in L_{jk}$ is $\min\{m(G - OPT + S_i, i, t_i), \beta_i\}$, which is at least $|m_i|$ by the above argument.

Lemma 28 $|M| \geq |\delta_{jk}|/2$.

Proof. The critical observation is that $|OPT(G_{jk} - M, L_{jk})| - |OPT(G_{jk} - M, L_{jk} - \{1\})| \leq m_1$. For the purpose of contradiction, assume the inequality does not hold. If the max-flow between 1 and t_1 on G_{jk} is less than β_1, then $m(G_{jk}, 1, t_1) = |m_1|$. Observe $m(G_{jk} - M, 1, t_1) \leq |m_1|$ since player 1 cannot send a bigger flow in a smaller graph. Then one can obtain a cheaper solution for the Network Multicut Game for the set of players L_{jk} on the graph $G_{jk} - M$ than $OPT(G_{jk} - M, L_{jk})$ by first cutting the edges of $OPT(G_{jk} - M, L_{jk} - \{1\})$ and then cutting a min-cut between 1 and t_1 on the remaining edges. If the max-flow between 1 and t_1 on G_{jk} is at least β_1, then $|m_1| = \beta_1$. Then one can obtain a cheaper solution for the Network Multicut Game for the set of players L_{jk} on the graph $G_{jk} - M$ than $OPT(G_{jk} - M, L_{jk})$ by only cutting the edges of $OPT(G_{jk} - M, L_{jk} - \{1\})$. Note than in this solution player 1 does not cut any edges and faces a cost of β_1.

With the same argument one can show $|OPT(G_{jk}-M, L_{jk}-\{1\})|-|OPT(G_{jk}-M, L_{jk}-\{1,2\})| \leq m_2$ and so on. Finally, we have $|OPT(G_{jk}-M, L_{jk}-\{1,2,...,(|L_{jk}|-1)\})| - |OPT(G_{jk} - M, \emptyset)| \leq m_{|L_{jk}|}$. Summing up all the telescoping inequalities, we obtain $|OPT(G_{jk} - M, L_{jk})| \leq |M|$. It is also clear that $|OPT(G_{jk}, L_{jk})| - |OPT(G_{jk} - M, L_{jk})| \leq |M|$. Therefore, $|\delta_{jk}| = |OPT(G_{jk}, L_{jk})| \leq 2|M|$. As desired, this proves that at least half of the edges of δ_{jk} are cut after one pass over the players of L_{jk}. ■

4.5 Edges with Non-Uniform Costs

In the previous sections, the cost to player i for cutting edges S_i was was just the number of edges $|S_i|$. We now consider a generalized version of our games where the edges have positive edge weights/costs $w(e)$. In order for an edge e to be cut, players will have to pay the weight $w(e)$ of the edge. That is, the cost to player i is no longer just the number of edges $|S_i|$, but their total weight which we represent by $w(S_i)$. Similarly, the cost of a strategy vector S is now $\text{cost}(S) = w(S) + \sum_{j \in Q(S)} \beta_j$, where $Q(S)$ is defined as in the Introduction.

For this more general model with non-uniform edge costs, we first show that if the weight of an edge cannot be split between players, then the price of stability can be very high. By allowing players to share the cost of edges, however, we are able to extend most of our results to this general case.

No Cost Sharing The game that we defined above does not allow players to share the cost of an edge, since if an edge $e \in S_i$, then the cost of player i increases by the full weight $w(e)$ of the edge. Now consider an example of the Single-Source Cutting game in Figure 4.2. The children represent the player nodes and the root t is the node that all players want to disconnect from. We assume that $\beta_i > 1$ for all i. This game has only one Nash Equilibrium, the cost of which is k, whereas the socially optimal solution costs $1 + \epsilon$. The price of stability is clearly k. This result also holds for the Network Multi-cut Game since it is a generalization of the Single-Source Cutting Game.

Notice that even if the edges did not have weights, this example shows that the *price of anarchy* can be as high as k.

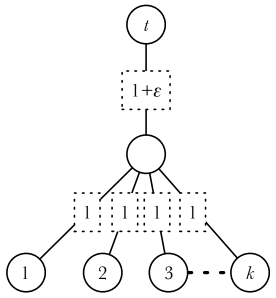

Figure 4.2: Example showing lower bounds on the Price of Anarchy for uniform edge weights, and Price of Stability for non-uniform edge weights.

If we allow players to share costs of cutting edges, then the results become much nicer. Specifically, we consider fair sharing and arbitrary sharing schemes.

Fair Sharing In this model, the players that cut an edge e split the cost of this edge equally among themselves. Specifically, for a given strategy vector S, define k_e to be the number of players i such that $e \in S_i$. Then, player i only pays $w(e)/k_e$ for cutting edge e, i.e., player i's cost is $\sum_{e \in S_i} w(e)/k_e$ when i's cut requirements are satisfied. All the games that we considered in sections 4.2, 4.3, and 4.4 are congestion games [5] under the fair sharing scheme. Using standard techniques [5], it can be shown that the price of stability is $O(\log k)$. Unfortunately, it can also be as high as $\Omega(\log k)$: just consider the example in Figure 4.2 with weights $1, \frac{1}{2}, \frac{1}{3}, \ldots, \frac{1}{k}$ on the bottom edges instead of 1.

Arbitrary Cost Sharing In this model players can choose to pay for arbitrary fractions of edge weights. Specifically, the strategy of each player i is a payment function S_i where $S_i(e)$ is the amount player i pays for the cost of edge e. An edge e is considered cut if the total payment for e exceeds its weight, i.e., $\sum_i S_i(e) \geq w(e)$. The cost of each player i is the total amount of payment it makes for cutting the edges, i.e., $\sum_e S_i(e)$. Observe that the arbitrary sharing model gives the players much larger freedom in selecting their strategies since the strategy space of each player is the positive orthant of the m-dimensional Euclidean space where $m = |E|$.

The algorithms presented in previous sections work for edges with unit cost. In order to extend those results to non-uniform weighted edges in the arbitrary cost sharing scheme, we make the following changes: Scale up the weights of edges so that all weights are integers. Simultaneously increase the value of β_i by the same factor. Now replace any edge having weight $w(e)$ with $w(e)$ parallel edges of unit cost. We can now use the algorithms in sections 4.2, 4.3, and 4.4 to assign these edges with unit cost to players. This assignment can be easily mapped to the original graph with weighted edges where the players pay for fractions of edge weights. This gives us a Nash equilibrium solution where the edge weights are arbitrarily shared.

Poly-time computable NE for Network Multiway Cut Game under Arbitrary cost sharing Since the algorithm for poly-time computable NE for the NMWG considered in Section 4.3.3 may take $O(E)$ steps, for non-uniform edge weights this may result in exponential running time. So the same algorithm does not work for this case. However we show that it is possible to form an approximate NE for NMWG in polynomial time.

Theorem 20 *Suppose we have a weighted NMCG and an MWC S^α that is within a factor α of OPT. Then for any $\epsilon > 0$, there is a poly-time algorithm which returns a $(1+\epsilon)$-approximate NE for a MWC S', where $w(S') \leq w(S^\alpha)$.*

Proof. To find a $(1+\epsilon)$-approximate NE, we start by defining $\lambda = \frac{\epsilon w(S^\alpha)}{\alpha(1+\epsilon)|E|}$. We now use the algorithm of section 4.3.3 to pay for all but λ for each edge in S^α. In this algorithm, every time a condition cannot be satisfied, we construct a new MWC with smaller weight. But now we are trying to pay for λ amount less for each edge in S^{α}. So every time a smaller weighted MWC is found, its weight must be at least λ smaller than the previously considered MWC.

Each step in the algorithm can be performed in polynomial time. Every time either of the conditions (i) and (ii) of 24 fail, the cost of the new MWC reduces at least by λ amount. This means that the algorithm can consider at most $\alpha\frac{(1+\epsilon)}{\epsilon}|E|$ multiway cuts. Therefore, in time polynomial in $|E|$, α and ϵ^{-1}, we have formed a MWC S' with $w(S') \leq w(S^{\alpha})$ such that the players are willing to buy S' if its edges have costs decreased by λ.

For all players and for each edge e in S', we now increase $S'_i(e)$ in proportion to S'_i so that e is fully paid for. Now S' is clearly paid for. To see that this is a $(1+\epsilon)$-approximate NE, note that player i was stable before its payments were increased. Player i's payments were increased by:

$$\begin{aligned}\lambda\frac{w(S'_i)}{w(S') - |S'|\lambda}|S'| &= \frac{\epsilon w(S^{\alpha})w(S'_i)|S'|}{\alpha(1+\epsilon)|E|(w(S') - |S'|\lambda)} \\ &\leq \frac{\epsilon w(S')w(S'_i)}{(1+\epsilon)(w(S') - |S'|\lambda)} \leq \epsilon w(S'_i)\end{aligned}$$

Thus any best response yields at most an ϵ factor improvement for any player i, and so this is a $1+\epsilon$ approximate Nash equilibrium. ■

CHAPTER 5
IMPROVEMENTS, GENERALIZATIONS AND FUTURE DIRECTIONS

5.1 Improvements Over Our Results

In this thesis, we have analyzed the approximate stability of some important network formation and cut games. In Chapter 2, we studied the Survivable Single-Source Connection Game where each player wants to have 2 edge-disjoint paths between her terminal and the root, rather than a single path. In Chapter 3, we introduced and studied a network formation game where players are forming a network together. In both chapters, we have shown the existence of 2-approximate Nash equilibrium that is as cheap as OPT and identified an interesting and natural subclass of those games (the case when all nodes are terminals) for which Nash equilibria is guaranteed to exist and the price of stability is 1. However, we don't know whether there always exist a Nash equilibrium or not for the general versions of these games.

Open Question 1: *Does there always exist a Nash equilibrium for the Survivable Single-Source Connection Game?*

Open Question 2: *Does there always exist a Nash equilibrium for the Group Network Formation Game?*

Discussion Observe that the proofs we have are *constructive*, i.e., we have proven the existence of approximate Nash equilibrium as cheap as OPT by giving an algorithm that explicitly form the strategies of the players on the edges of OPT. So, even if Nash equilibrium exist for the general versions of the games we analyzed, existence of them cannot be proven with our techniques unless price of stability for them is 1. Therefore, a more novel approach is to be followed to answer these open questions.

The analysis of the algorithms we have given for the general versions of both Survivable Single-Source Connection Game and the Group Network Formation Game in Chapter 2 and Chapter 3 are tight, i.e., one can construct respective instances of those games $\Im_1$ and $\Im_2$ such that both $\Im_1$ and $\Im_2$ has at least one player i for which the cost of the cheapest deviation $\chi_i(p_i)$ from the strategy p_i formed by the algorithm is exactly half of the cost of p_i. However; the algorithms themselves may not be tight, i.e., there may be algorithms that forms solutions with lower approximation ratios while ensuring that the total cost of the solution is no more than the cost of OPT.

Open Question 3: *Is there an $\alpha < 2$ such that there always exist an α-approximate Nash equilibrium as cheap as OPT for the Survivable Single-Source Connection Game?*

Open Question 4: *Is there an $\alpha < 2$ such that there always exist an α-approximate Nash equilibrium as cheap as OPT for the Group Network Formation Game?*

Discussion Recall that in Chapter 2, we have defined a stability notion that restricts the deviations of the players and shown that price of stability is 1 with respect to this stability notion. This stability notion was related to Nash equilibria, i.e., every stable solution was a 2-approximate Nash equilibrium as declared by Theorem 5 and this bound is tight. Therefore, one cannot show the existence of an α-approximate Nash equilibrium as cheap as OPT where $\alpha < 2$ for the Survivable Single-Source Connection Game by using the technique in Chapter 2. For the Group Network Formation Game, however, we haven't used a notion of stability other than Nash equilibria. The technique we used was similar to the technique used in [6] but did not form a Nash equilibrium but a 2-approximate Nash equilibrium, since whenever 2 or more of the lower level incident edges of a nonterminal node u are bought in the previous iterations of the algorithm, we fixed the strategies of the players that bought these edges and asked a new player to pay for the upper level incident edge of u. That property of the algorithm enabled us to prove Lemmas 14, 15 and 16 at the expense of an approximation factor of 2. Existence of an α-

approximate Nash equilibrium as cheap as OPT where $\alpha < 2$ for the Group Network Formation Game can be proven by using our techniques by coming up smarter rewiring techniques that does not require the algorithm to have aforementioned property.

For the Multi-Source Connection Game, it has been shown in [6] that there always exist a 3-approximate Nash equilibrium as cheap as OPT and there are instances of the game for which there is no $(1.5 - \epsilon)$-approximate Nash equilibrium as cheap as OPT. The results in [6] have not been improved yet and the following interesting problems are still unanswered.

Open Question 5: *Is there an $\alpha \geq 1.5$ such that there is no α-approximate Nash equilibrium as cheap as OPT for the Multi-Source Connection Game?*

Open Question 6: *Is there an $\alpha < 3$ such that there always exists an α-approximate Nash equilibrium as cheap as OPT for the Multi-Source Connection Game?*

As we have shown in Chapter 2 for the Survivable Single-Source Connection Game and in Chapter 3 for the Group Network Formation Game, it is often possible to prove better results with much easier analysis for the special case where all nodes are terminals. Therefore, we believe that the following open questions can be answered soon.

Open Question 7: *Is there an $\alpha \geq 1.5$ such that there is no α-approximate Nash equilibrium as cheap as OPT for the Multi-Source Connection Game when all nodes are terminals?*

Open Question 8: *Is there an $\alpha < 3$ such that there always exists an α-approximate Nash equilibrium as cheap as OPT for the Multi-Source Connection Game when all nodes are terminals?*

Though we have focused on approximate stability results that are as cheap as OPT, existence and computability of stable solutions that cost slightly more are also problems of interest. More precisely, the problems could have been stated as bicriteria approximation problems where an (α, β)-approximate Nash equilibrium means an α-approximate Nash equilibrium whose cost is within a β factor of OPT.

Open Problem 9: *Is there an (α, β)-approximate Nash equilibrium for any of the above problems where both α and β are small?*

5.1.1 Generalizations and Future Directions

In the Group Network Formation Game, we have assumed monotonicity of the happiness function F, i.e., if $F(S) = 1$ for some set of terminals then $F(T) = 1$ for any $T \supset S$. Recall that if there is no restriction on F, then there is no approximate Nash equilibrium as cheap as OPT. Though the model we have presented in Chapter 3 is quite general and covers quite a lot of interesting applied problems, there are many other interesting problems that does not fit in the model. Therefore, it is both a theoretical and a practical interest to investigate stability of group network formation with more general happiness functions.

As a starting point, let us define "monotonicity" from a different viewpoint. Let S and T be two arbitrary disjoint subsets of the terminal nodes. Then F is monotone if and only if $F(S \cup T) \geq \max\{F(S), F(T)\}$. In other words, the union of 2 sets at least one of which is happy is also a happy set, while the union of two unhappy sets may or may not be happy. In the rest of the discussion, we call such functions type 1 monotone and define 2 other notions of monotonicity as illustrated in Table 5.1.

Case	Type 1	Type 2	Type 3
$F(S) = 1, F(T) = 1$	$F(S \cup T) = 1$	$F(S \cup T) = 1$	$F(S \cup T) = 1$
$F(S) = 1, F(T) = 0$	$F(S \cup T) = 1$	$F(S \cup T) = 0$	$F(S \cup T)$ is free
$F(S) = 0, F(T) = 0$	$F(S \cup T)$ is free	$F(S \cup T)$ is free	$F(S \cup T)$ is free

Table 5.1: Definition of Various Notions of Monotonicity

Our second notion of monotonicity, which we call type 2 monotone and precisely define in Table 5.1 tries to model the scenarios where the players have several types and they try to form groups such that the ratio of the number of players of any

2 types t_1 and t_2 is constant for each happy set. To have a better understanding of this notion of monotonicity consider the following easy example. Each player node of the graph is colored to either red or blue and a component is happy if and only if it has equal number of red and blue nodes. Notice that the union of 2 happy sets is always happy, the union of a happy and an unhappy set is always unhappy while the union of two unhappy sets may or may not be happy. Notice also the set of type 2 monotone functions are disjoint from the set of type 1 monotone functions.

Our last notion of monotonicity, which we call type 3 monotone functions and define precisely in Table 5.1 is an extremely general model that imposes only one restriction on the happiness functions, i.e., the union of 2 happy sets is always happy. Notice that, type 3 monotone functions are all functions that satisfy maximality condition [31] and therefore cannot be approximated by the primal-dual approximation algorithm given by Goemans and Williamson given in [27].

Open Question 10: *Does the Group Network Formation Game always have a Nash equilibrium if the happiness functions are type* 2 *monotone? If so, what is the price of anarchy and the price of stability?*

Open Question 11: *Does the Group Network Formation Game always have* (α, β)*-approximate Nash equilibrium for small* α *and* β *if the happiness functions are type* 3 *monotone? If so, what are the bounds on* α *and* β*?*

Different cost-sharing mechanism Among various cost-sharing mechanisms [16], we have used the arbitrary cost-sharing mechanism in our earlier work because of the nice properties it has which are listed in Chapter 1. In this cost-sharing scheme, no player pays for an edge she does not witness in a Nash equilibrium. However, this statement is not true for approximate equilibrium solutions and unfortunately often the best approximate Nash equilibrium solution requires players to pay for edges that they don't even use. Therefore, even though such solutions ensures that the players can gain very little if they deviate (or gain nothing in scenarios where deviation is costly), the solutions does not seem viable. Furthermore, this unpleasant property complicates the analysis of approximate equilibrium solutions by overly extending the space of possible best deviations of the players. For that reason, we propose the following slight restriction on arbitrary cost-sharing mechanism. In this new cost-sharing scheme, the players can still decide the amount they will pay for each edge but they only pay for the edges they witness. Observe that this new cost-sharing scheme still gives a lot of freedom to the players and does not require the existence of an overseeing authority. We think that one can obtain tighter (especially lower) bounds of approximability in this cost-sharing scheme.

Open Question 12: *What are the answers of the above open questions if players can only pay for the edges they witness?*

A general framework for Connection Games We now propose a very general framework for network creation games where each player wants to connect her terminal to a set of source nodes. Let A be a matrix that has a row for each player and a column for each source. Let A_{ij} denote the entry of A at row i and column j. Each entry of A is a real number between 0 and 1. The connectivity requirement of each player i is to connect to an arbitrary set of sources J such that $\sum_{j \in J} A_{ij} \geq 1$. That is a quite general framework since each matrix in the specified domain correspond to a different game. For example, if A is a $0-1$ matrix such that each row has exactly one entry that is equal to 1 then this game is precisely the Multi-Source Connection Game. Furthermore, if the entries that are equal to 1 are at the same column then this game is the Single-Source Connection Game. As another example, let each entry of A be $\frac{1}{k}$ where k is the number of columns of A. Then, in this game each player wants to connect to all sources. We next ask some interesting questions for the games in this framework.

Open Question 13: *For which matrices A, Nash equilibrium is guaranteed to exist?*

Open Question 14: *Is the existence of Nash equilibrium related with linear algebraic properties of A?*

LITERATURE CITED

[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 and B. Caskurlu. Exact and Approximate Equilibria for Optimal Group Network Formation. In *Proc. 17th Annual European Symposium on Algorithms* (ESA 2009).

[4] E. Anshelevich and B. Caskurlu. Price of Stability in Survivable Network Design. In *Proc. 2nd International Symposium on Algorithmic Game Theory* (SAGT 2009).

[5] 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.

[6] 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.

[7] E. Anshelevich and A. Karagiozova. Terminal Backup, 3D Matching, and Covering Cubic Graphs. In *Proc. 39th ACM Symposium on Theory of Computing* (STOC 2007).

[8] E. Anshelevich, B. Shepherd, G. Wilfong. Strategic Network Formation through Peering and Service Agreements. FOCS 2006.

[9] J. Aspnes, K. Chang, A. Yampolskiy. Innoculation Strategies for Victims of Viruses and the Sum-of-Squares Problem. In *Journal of Computer and System Sciences*, 72(6):1077–1093, September 2006

[10] R. Aumann. Acceptable Points in General Cooperative n-person Games. *Contributions to Theory of Games*, 1959.

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

[12] D. Braess. On a paradox of traffic planning. In *Transport Sci.*, 39(4): 446-450, 2005.

[13] G. Calinescu, H. Karloff, and Y. Rabani. An improved approximation algorithm for multiway cut. In *Proc. 30th Ann. ACM Symp. on Theory of Comp.*, ACM-SIAM (1998), 48–52.

[14] 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.

[15] H. Chen, T. Roughgarden. Network Design with Weighted Players. In *SPAA* 2006.

[16] H. Chen, T. Roughgarden, and G. Valiant. Designing Networks with Good Equilibria. *SODA 2008.*

[17] G. Christodoulou and E. Koutsoupias. On the price of anarchy and stability of correlated equilibria of linear congestion games. ESA, 2005.

[18] G. Christodoulou, V.S. Mirrokni, A. Sidiropoulos. Convergence and Approximation in Potential Games. In *STACS* 2006.

[19] J.R. Correa, A.S. Schulz and N.E. Stier-Moses. A Geometric Approach to the Price of Anarchy in Nonatomic Congestion Games. In *Games and Economic Behavior*, 64, 457-469, 2008.

[20] R. Engelberg, J. Könemann, S. Leonardi, J. Naor. Cut Problems in Graphs with a Budget Constraint. In *Journal of Discrete Algorithms*, Volume 5 , Issue 2 (June 2007).

[21] A. Epstein, M. Feldman, and Y. Mansour. Strong Equilibrium in Cost-Sharing Connection Games. *EC* 2007.

[22] A. Fabrikant, A. Luthra, E. Maneva, S. Papadimitriou, and S. Shenker. On a Network Creation Game. In *PODC*, 2003.

[23] A. Fabrikant, C. Papadimitriou, K. Talwar. The complexity of pure Nash equilibria. In *STOC*, 2004.

[24] J. Feigenbaum, C. Papadimitriou, S. Shenker. Sharing the Cost of Multicast Transmissions. *Journal of Computer and System Sciences* 63 (2001), 21–41.

[25] 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.

[26] N. Garg, G. Konjevod, R. Ravi. A polylogarithmic approximation algorithm for the group Steiner tree problem. *SODA 2000.*

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

[28] L. Gourves, J. Monnot. On Strong Equilibria in the Max Cut Game. In *WINE* 2009.

[29] S.N. Hamilton, W. L. Miller, A. Ott, O. S. Saydjari. Challenges to applying game theory to the domain of information warfare. In *4th Information survivability workshop (ISW-2001/2002)*, Vancouver, Canada, 2002.

[30] Shai Herzog, Scott Shenker, Deborah Estrin. Sharing the "Cost" of Multicast Trees: An Axiomatic Analysis. *IEEE/ACM Transactions on Networking*, Dec. 1997.

[31] Dorit S. Hochbaum. *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, 1997.

[32] M. Hoefer. Cost Sharing and Clustering under Distributed Competition, Ph.D Thesis, Universitat Konstanz (2007).

[33] M. Hoefer. Non-cooperative Facility Location and Covering Games. In *ISAAC* 2006.

[34] M. Hoefer. Non-cooperative Tree Creation. In *MFCS* 2006.

[35] M. Hoefer, P. Krysta. Geometric Network Design with Selfish Agents. *COCOON* 2005.

[36] R. Holzman, N. Law-Yone. Strong Equilibrium in congestion games. *Games and Economic Behavior,* 21, 1997.

[37] 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.

[38] K. Jain. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem. *Combinatorica,* 21, 2001.

[39] K. Jain and V. Vazirani. Applications of Approximation Algorithms to Cooperative Games. In *STOC*, 2001.

[40] R. Johari and J. Tsitsiklis. Efficiency loss in a network resource allocation game. *Mathematics of Operations Research,* 29 (3): 407–435.

[41] M. Kearns, L. Oritz. Algorithms for Interdependent Security Games. In *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[42] J. Könemann, S. Leonardi, and G. Schäfer. A group-strategyproof mechanism for Steiner forests. *SODA* 2005.

[43] Y.A. Korillis, A.A. Lazaar, and A. Orda. Achieving Network Optima Using Stackleberg Routing Strategies. *IEEE/ACM Transactions on Networking,* 5(1), 1997.

[44] E. Koutsoupias, C. Papadimitriou. Worst-Case Equilibria. In *Proc. 16th Annual Symposium on Theoretical Aspects of Computer Science*, 1999.

[45] H. Kunreuther, G. Heal. Interdependent Security. *Journal of Risk and Uncertainity (Special Issue on Terrorist Risks)*, 2003.

[46] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, N. Glance. Cost-effective Outbreak Detection in Networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD), 2007.

[47] M. Mavronicolas, L. Michael, V. G. Papadopoulou, A. Philippou and P. G. Spirakis. The Price of Defense. In *31st International Symposium on Mathematical Foundations of Computer Science*, 2006.

[48] D. Monderer, L. Shapley. Potential Games. *Games and Economic Behavior* 14(1996), 124–143.

[49] J. Nash. Noncooperative Games. *Annals of Mathematics*, 54:289-295, 1951.

[50] A.C. Pigou. The Economics of Welfare. *Macmillan*, 1920.

[51] N. Nisan, T. Roughgarden, É. Tardos, and V. V. Vazirani (eds.), *Algorithmic Game Theory,* Cambridge University Press.

[52] R. W. Rosenthal. The network equilibrium problem in integers. *Networks*, 3:53-59, 1973.

[53] T. Roughgarden. *Selfish Routing and the Price of Anarchy,* MIT Press.

[54] T. Roughgardon. On the severity of Braess's Paradox: Designing networks for selfish users is hard. In *J. Computer System Sci.*, 72(5):922-953, 2006.

[55] T. Roughgardon and E. Tardos. How bad is selfish routing? In *J. ACM*, 49(2):236-259, 2002.

[56] P. F. Syverson. A different look at secure distributed computation. In *IEEE Computer Security Foundations Workshop (CSFW 10)*, June 1997.

[57] É. Tardos. Network Games. In *Proceedings of the 36th Annual ACM Symposium on the Theory of Computing*, 2004.

[58] J.G. Wardrop. Some theoretical aspects of road traffic research. In *Proc. Institute of Civil Engineers, Pt.2*, volume 1, pp. 325-378, 1952.

[59] D. Xu, E. Anshelevich, and M. Chiang. On Survivable Access Network Design: Complexity and Algorithms. *INFOCOM 2008.*

[60] G. Robins, A. Zelikovsky. Improved Steiner Tree Approximation in Graphs. SODA 2000, pg. 770-779.