Buyer-Supplier Games: Core Characterization and Computation

Nedialko B. Dimitrov[*] C. Greg Plaxton[†]

April 2006

Abstract

In a buyer-supplier game, a distinguished player, called the buyer, wishes to purchase some combinatorial object. A set of players, called suppliers, offer pieces of the object for sale. In this paper, we provide a transformation from most combinatorial minimization problems to buyer-supplier games; a characterization of the core of buyer-supplier games in the transferable and non-transferable utility cases; a polynomial time algorithm for optimization over the core of buyer-supplier games for which the underlying minimization problem is solvable in polynomial time; and an impossibility result showing that it is hard to distinguish core vectors if the base minimization problem is not solvable in polynomial time. We also introduce and study the concept of focus point price, which answers the question: If we are constrained to play in equilibrium, how much can we lose by playing the wrong equilibrium?

[*]Supported by an MCD Fellowship from the University of Texas at Austin.
[†]Supported by NSF Grants CCR–0310970 and ANI–0326001.

1 Introduction

A game is a strategic interaction between a group of players, where the aim of each player is to maximize the player's own payoff. In game theory, a solution concept predicts the outcome of a game; the predictions made by the solution concept are called equilibria. Differing models of game play lead to different solution concepts. For example, the famous Nash equilibrium solution concept was developed with the notion that players make strategic decisions simultaneously, without any inter-player communication [10]. The correlated equilibrium solution concept, on the other hand, was developed with the notion that players may use some common randomness as the sole form of communication, and as a source of collaboration [2].

In contrast, the absence of restrictions on inter-player communication leads to the core solution concept. We give a formal definition of the core of a game in Section 2.2. Informally, the core is a subset of the payoff vectors possible from outcomes to the game. If no collusion were allowed, only some of the payoff vectors would be rational, since an individual player would never accept a payoff that the player can unilaterally and selfishly improve. Allowing some sets of players to collude further restricts the set of rational vectors, since a set of players would never accept collective payoffs that they can unilaterally and selfishly improve. The core is the set of payoff vectors that are still rational when we allow every subset of the players to collude.

In this paper, we study the core of a large set of games, which we term buyer-supplier games. We are concerned both with the characterization of the core, as well as efficient computations over the set of vectors belonging to the core. Before diving into an overview of buyer-supplier games, we present some connections between our work and the existing literature.

1.1 Related Work

Though suggested by Edgeworth as early as 1881 [3], the notion of the core was formalized by Gillies and Shapley [6, 20], extending von Neumann and Morgenstern's work on coalitional game theory [22]. Recently, Goemans and Skutella studied the core of a cost sharing facility location game [7]. In their paper, Goemans and Skutella are primarily interested in using core vectors as a cost sharing indicator, to decide how much each customer should pay for opening the facility used by the customer. Goemans and Skutella show that, in general, the core of the cost sharing facility location game they study is empty. In contrast, for the buyer-supplier games we study, we show that the core is always nonempty. In another contrast, in our work we do not view vectors in the core as an indication of cost shares but rather as rational outcomes of negotiation amongst the players in the buyer-supplier game. Pál and Tardos extend the work of Goemans and Skutella by developing a mechanism for the cost sharing facility location game which uses the concept of an approximate core [13].

There has been a great interest in theoretical computer science on comparing the game's best outcome to the best equilibrium outcome, where the term best is based on some desired criterion. For example, one may wish to compare the outcome maximizing the net utility for all players in the game against the best possible Nash equilibrium, from the point of view of net utility. Papadimitriou termed one such comparative measure as the price of anarchy [14]. Roughgarden and Tardos have studied the price of anarchy in the context of routing [17, 18, 19].

In this paper, we introduce and study the focus point price, a quantity with a similar motivation to that of the price of anarchy. Solution concepts often yield multiple predictions, or equilibria. In actual game play, however, only one of the equilibria can be chosen by the game's players. Experiments show that conditions outside the game, such as societal pressures or undue attention to a specific player, focus the players' attention on the point of a single equilibrium. The focus point price measures the value of a good focus point. In other words, the focus point price is the difference between the best and worst equilibrium outcome, where the terms best and worst are based on some desired criterion. Stated succinctly, the focus point price answers the question: If we are constrained to play in equilibrium, how much can we lose by playing the wrong equilibrium?

One of the central concerns in the applications of game theory is the problem of private information. The problem arises because players may be unwilling to share their private evaluations of the game's outcomes for strategic reasons. Since the utility functions of the players are private, making optimal strategic decisions in the game becomes impossible. In the literature, there have been two main approaches to dealing with private information.

One approach to dealing with private information is through the design of truthful mechanisms. We call a game truthful if it is to every player's advantage to reveal all private information. A truthful mechanism is a truthful game in which the equilibrium outcomes have some desired property. In a truthful mechanism, once all private information is revealed, choosing an outcome with the desired property is reduced to a computational problem. There has been a great amount of work in theoretical computer science on truthful mechanism design, starting with the work by Nisan and Ronen, which introduced algorithmic mechanism design to the computer science theory community [12]. Feigenbaum, Papadimitriou, and Shenker have used truthful mechanisms in the context of multicast transmissions [4]. Tardos and Archer have created a general framework for designing truthful mechanisms where the private information of each player is a single positive real number [1].

While numerous advances have been made in the area of truthful mechanisms, significant challenges still remain. First, if the players are involved in just two consecutive, identical executions of a truthful mechanism, in general, it is no longer the case that revealing all private information is the optimal strategy. The brittleness of truthfulness under repetition holds even with the most widely known truthful mechanism, the Vickrey auction [23]. A second challenge for truthful mechanisms is related to the difficulty of enforcing a desirable property known as budget balance. To maintain truthfulness, many mechanisms make excessively large payments to the players. The payments made by the mechanism are often larger than the total wealth present at the beginning of the game, implying that the mechanism is not budget balanced. Thus, budget balance often conflicts with truthfulness and the mechanisms' desired property. For example, in general, it is impossible to design truthful, budget balanced mechanisms that maximize the net utility to all players [16].

A second approach to dealing with the problem of private information is the method of types developed by Harsanyi [8]. Informally, the method of types removes the problem of private information by introducing a mathematically meaningful notion of beliefs into the game. Thus, any particular player has beliefs, represented by probability distributions, about the private information of the other players. With this model, the player can make optimal decisions by pretending to play a random game with complete information, where the game is drawn from a distribution determined by the player's beliefs about the private values of the other players. Thus, the key to the method's application is the ability to deal with a wide range of games of complete information. While mathematically elegant, Harsanyi's method of types does have the practical drawback that a player may be entirely unsure of their own beliefs. However, because of its elegance and robust applicability, in this paper, we take Harsanyi's approach to dealing with the problem of private information. We present computational results for a wide range of buyer-supplier games of complete information. Our results can be directly applied to obtain results in the private information setting.

1.2 Overview of Buyer-Supplier Games

The definition of a buyer-supplier game, given in Section 2.1, is self-contained and generic. However, it is also possible to transform a combinatorial minimization problem into a buyer-supplier game. Consider a combinatorial minimization problem of the following form. We have some finite set of elements C. We designate some subsets of C as feasible. To capture feasibility, we use a predicate $P : 2^C \rightarrow \{0, 1\}$, where the predicate is one on all feasible subsets of C. With each feasible set $\mathcal{A} \subseteq C$, we associate a nonnegative cost $f(\mathcal{A})$. The combinatorial minimization problem can then be captured by the function MinProb : $2^C \rightarrow \Re_+$

defined by

$$\text{MinProb}(\mathcal{B}) = \min_{\substack{\mathcal{A} \subseteq \mathcal{B} \\ P(\mathcal{A}) = 1}} f(\mathcal{A})$$

where $\mathfrak{R}_+$ denotes the nonnegative real numbers.

To transform the above minimization problem into a buyer-supplier game, we associate a player with each element of C; we call such players suppliers. We also add another player whom we call the buyer. In the game, the buyer wishes to purchase a feasible subset of C. The suppliers, on the other hand, are offering their membership to the buyer's set at a price.

To fully specify the game's model of a realistic interaction, we let M designate the maximum investment the buyer is willing to spend on a feasible set. We decompose f such that $f(\mathcal{A}) = \text{Bcost}(\mathcal{A}) + \sum_{a \in \mathcal{A}} \tau(a)$, where $\tau(a)$ is the internal cost for supplier a to be present in the buyer's set and $\text{Bcost}(\mathcal{A})$ is the internal cost to the buyer for purchasing this specific feasible set. In general, many such decompositions are possible, and they produce different games. However, when specifically applying the core solution concept, Lemma 3.22 shows that all such decompositions are equivalent. Though it is not necessary, to remove special cases in our statements, it is convenient to let $\text{Bcost}(\mathcal{A}) = M$ when $\mathcal{A} = \emptyset$ or $\mathcal{A}$ is not feasible.

Now that we have determined the internal costs for the buyer and the suppliers, we can specify the game. The buyer-supplier game is specified by the tuple (C, τ, Bcost). The strategy set for the buyer is the power set of C. By playing $\mathcal{A} \subseteq C$, the buyer chooses to purchase the membership of the suppliers in $\mathcal{A}$. The strategy set for every supplier $a \in C$ is the nonnegative real numbers, indicating a bid or payment required from the buyer for the supplier's membership.

For any supplier $a \in C$, we let $\beta(a)$ denote the associated bid. Let $\mathcal{A}$ be the set of suppliers chosen by the buyer. The payoff for the buyer is $M - \text{Bcost}(\mathcal{A}) - \sum_{a \in \mathcal{A}} \beta(a)$. The payoff for a supplier not in $\mathcal{A}$ is 0. The payoff for a supplier a in $\mathcal{A}$ is $\beta(a) - \tau(a)$.

Since we are applying the solution concept of the core, one may think of the game play as follows. All the players in the game sit down around a negotiating table. All the players talk amongst themselves until they reach an agreement which cannot be unilaterally and selfishly improved upon by any subset of the players. Once such an agreement is reached, game play is concluded. Since no subset of the players can unilaterally and selfishly improve upon the agreement, rationality binds the players to follow the agreement.

The fully formal definition of a buyer-supplier game is given in Section 2.1. The transformation process described above can be used to create buyer-supplier games from most combinatorial minimization problems. For example, minimum spanning tree, Steiner tree, shortest path, minimum set cover, minimum cut, single- and multi-commodity flow can all be used to instantiate a buyer-supplier game. As a concrete example and interpretation of a buyer-supplier game, consider the buyer-supplier minimum spanning tree game. In this game, a company owns factories on every node of a graph. The company wishes to connect the factories by purchasing edges in the graph. Each edge is owned by a unique supplier player. Each supplier has an internal cost associated with the company's usage of the edge. The company has a maximum amount of money it is willing to spend on purchasing edges. Depending on the transportation conditions of a particular edge, the company may have some internal cost associated with choosing that particular edge. The buyer-supplier game paradigm yields similarly natural games when applied to other minimization problems.

In this paper we will be concerned with efficient computation over the set of core vectors. For the rest of the paper, when we say polynomial time, we mean time polynomial in the size of the parameter C, which is also polynomial in the number of players of the buyer-supplier game.

1.3 Main Contributions

There has been increased interest from the theoretical computer science community in game theory. While problem-specific solutions may give us insight, to leverage the full power of decades of study in both research areas, we must find generic computational solutions to game theoretic problems. Indeed, others have already realized this need [1, 15]. In this paper, we continue this line of work by deriving generic results for computing with core solutions in a large class of games.

First, we provide a framework for constructing a natural game given a common minimization problem. We call the constructed game a buyer-supplier game.

Second, we give a characterization of the core of buyer-supplier games. Surprisingly, we show that the characterization of the core remains unchanged when the game allows side payments. Stated differently, the core is the same in the transferable and non-transferable utility versions of the game.

Third, we provide a generic algorithm, based on the ellipsoid method, for optimizing over the core. If the original minimization problem is solvable in polynomial time, we show that it is possible to optimize linear functions of core vectors in polynomial time.

Fourth, we use a polynomial time reduction to show that if the original minimization problem is not solvable in polynomial time, it is impossible, in polynomial time, to determine if an arbitrary vector is in the core of the buyer-supplier game.

Fifth, we introduce the concept of focus point price. Our positive computational results give a polynomial time algorithm for computing the focus point price when the original minimization problem is solvable in polynomial time. When the original minimization problem is not solvable in polynomial time, we show that it is impossible to approximate the focus point price to within *any* multiplicative factor.

We finally conclude the paper by giving a problem-specific combinatorial algorithm, complementing our generic algorithm, for computing the focus point price of the buyer-supplier game based on the minimum spanning tree problem. The combinatorial algorithm results from the favorable properties of minimum spanning trees and is a minor extension of Kruskal's algorithm.

1.4 Organization of the Paper

In Section 2, we define buyer-supplier games and the core of a game. In Section 3, we characterize the core of buyer-supplier games. In Section 4, we give positive computational results, namely the generic algorithm for optimizing over the set of core vectors. In Section 5, we give negative computational results by showing polynomial time equivalence between several related problems. In Section 6, we give a small simplification to a linear program arising from the focus point price problem. In Section 7, we give the problem-specific combinatorial algorithm for the buyer-supplier game arising from the minimum spanning tree problem. In Section 8, we show several applications of the results in this paper to a range of buyer-supplier games.

2 Definitions

2.1 Buyer-Supplier Games

Let C be a finite set and M be a nonnegative real number. Let τ be a function from C to $\Re_+$. Let Bcost be a function from 2^C to $\Re_+$ such that $\text{Bcost}(\emptyset) = M$. The simplifying condition that $\text{Bcost}(\emptyset) = M$ is not required. We explain the condition's purpose later in this section. For $\mathcal{A} \subseteq C$, let $\text{Eval}(\mathcal{A}, \tau, \text{Bcost}) = \text{Bcost}(\mathcal{A}) + \sum_{a \in \mathcal{A}} \tau(a)$. For $\mathcal{A} \subseteq C$, let $\text{MinEval}(\mathcal{A}, \tau, \text{Bcost}) = \min_{\mathcal{B} \subseteq \mathcal{A}} \text{Eval}(\mathcal{B}, \tau, \text{Bcost})$. We will omit the parameters τ and Bcost from the functions $\text{Eval}(\mathcal{A}, \tau, \text{Bcost})$ and $\text{MinEval}(\mathcal{A}, \tau, \text{Bcost})$ when the value of the parameter is clear from context.

Given a tuple (C, τ, Bcost), we proceed to define a buyer-supplier game. Associate a player with each element of C. Call the players in C suppliers. Let there also be another player, μ, whom we call the buyer. Let $\mathcal{P} = C \cup \{\mu\}$ be the set of players for the buyer-supplier game.

The strategy for supplier a is a tuple $(\beta(a), p_a)$ with $\beta(a) \in \Re_+$ and $p_a : \mathcal{P} \to \Re_+$. The first element, $\beta(a)$, represents supplier a's bid to the buyer, requiring the buyer to pay $\beta(a)$ for using the supplier's services.

The second element, p_a, represents the nonnegative side payments supplier a chooses to make to the game's players. By $p_a(b)$ we denote the side payment a makes to player b.

The strategy for the buyer, μ, is a tuple $(\mathcal{A}, p_\mu)$ where $\mathcal{A} \in 2^C$ and $p_\mu : \mathcal{P} \to \Re_+$. The first element, $\mathcal{A}$, represents the suppliers chosen by the buyer for a purchase. Similarly to a supplier, the second element, p_μ, represents the nonnegative side payments the buyer chooses to make to the game's players.

For each player $a \in \mathcal{P}$ we denote the player's strategy set by $\mathcal{S}_a$. For a set of players $\mathcal{A} \subseteq \mathcal{P}$, we denote the set of strategies $\bigotimes_{a \in \mathcal{A}} \mathcal{S}_a$ by $\mathcal{S}_\mathcal{A}$. We call elements of $\mathcal{S}_\mathcal{A}$ strategy vectors. We index strategy vectors from $\mathcal{S}_\mathcal{A}$ by the elements of $\mathcal{A}$. Given a tuple (C, τ, Bcost), we proceed to define a buyer-supplier game. Associate a player with each element of C. Call the players in C suppliers. Let there also be another player, μ, whom we call the buyer. Let $\mathcal{P} = C \cup \{\mu\}$ be the set of players for the buyer-supplier game.

The strategy for supplier a is a tuple $(\beta(a), p_a)$ with $\beta(a) \in \Re_+$ and $p_a : \mathcal{P} \to \Re_+$, where $\Re_+$ represents the nonnegative real numbers. The first element, $\beta(a)$, represents supplier a's bid to the buyer, requiring the buyer to pay $\beta(a)$ for using the supplier's services. The second element, p_a, represents the nonnegative side payments supplier a chooses to make to the game's players. By $p_a(b)$ we denote the side payment a makes to player b.

The strategy set for μ is a tuple $(\mathcal{A}, p_\mu)$ where $\mathcal{A} \in 2^C$ and $p_\mu : \mathcal{P} \to \Re_+$. The first element, $\mathcal{A}$, represents the suppliers chosen by the buyer for a purchase. Similarly to a supplier, the second element, p_μ, represents the nonnegative side payments the buyer chooses to make to the game's players.

For each player $a \in \mathcal{P}$ we denote the player's strategy set by $\mathcal{S}_a$. For a set of players $\mathcal{A} \subseteq \mathcal{P}$, we denote the space of strategies $\bigotimes_{a \in \mathcal{A}} \mathcal{S}_a$ by $\mathcal{S}_\mathcal{A}$. We call elements of $\mathcal{S}_\mathcal{A}$ strategy vectors. We index strategy vectors from $\mathcal{S}_\mathcal{A}$ by the elements of $\mathcal{A}$.

We now define the utility function for each player. Suppose strategy $s \in \mathcal{S}_\mathcal{P}$ is played. Specifically, suppose that $s_\mu = (\mathcal{A}, p_\mu)$ and $s_a = (\beta(a), p_a)$ for each $a \in C$. The utility functions for the game's players are

$$\begin{aligned}
u_\mu(s) &= M - \left[\text{Bcost}(\mathcal{A}) + \sum_{a \in \mathcal{A}} \beta(a)\right] + \left[\sum_{b \in \mathcal{P}} p_b(\mu) - \sum_{b \in \mathcal{P}} p_\mu(b)\right] && \\
u_a(s) &= \beta(a) - \tau(a) + \left[\sum_{b \in \mathcal{P}} p_b(a) - \sum_{b \in \mathcal{P}} p_a(b)\right] && \text{for } a \in \mathcal{A} \\
u_a(s) &= \left[\sum_{b \in \mathcal{P}} p_b(a) - \sum_{b \in \mathcal{P}} p_a(b)\right] && \text{for } a \in C - \mathcal{A}.
\end{aligned}$$

Interpreting, the buyer begins with M utility and chooses to make a purchase from each supplier in $\mathcal{A}$. The buyer gives $\beta(a)$ to each supplier $a \in \mathcal{A}$ and loses an extra $\text{Bcost}(\mathcal{A})$ from the initial M utility. Each supplier a in $\mathcal{A}$ receives the bid payment from the buyer and loses $\tau(a)$ because the supplier must perform services for the buyer. The distribution of side payments completes the utility functions. The requirement that $\text{Bcost}(\emptyset) = M$ lets the strategy $\emptyset$ stand as a "don't play" strategy for the buyer. To remove the requirement, we could introduce a specific "don't play" strategy to the buyer's strategy set, however this creates a special case in most of our proofs.

Let the side payment game we have defined be denoted SP. Let NOSP denote the same game with the additional requirement that all side payments be fixed to zero. In other words, in NOSP we restrict the strategy set for each $a \in \mathcal{P}$ so that p_a is identically zero.

2.2 Game Theoretic Definitions

All of the definitions in this section closely follow those of Shubik [21, Chapter 6].

We call a vector in $\Re^{|\mathcal{P}|}$, indexed by $a \in \mathcal{P}$, a *payoff vector*. We say a payoff vector π is *realized* by a strategy vector $s \in \mathcal{S}_\mathcal{P}$ if $\pi_a = u_a(s)$ for all $a \in \mathcal{P}$.

Let π be a payoff vector and s be a strategy vector in $\mathcal{S}_\mathcal{A}$ for $\mathcal{A} \subseteq \mathcal{P}$. Let t be any strategy vector in $\mathcal{S}_\mathcal{P}$ such that the projection of t onto the coordinates in $\mathcal{A}$ is equal to s. If for all t and for all $a \in \mathcal{A}$ we have $\pi_a \leq u_a(t)$, we say that the players in $\mathcal{A}$ can *guarantee* themselves payoffs of at least π by playing s.

We use Shubik's alpha theory to define our characteristic sets [21, pp. 134-136]. Thus for a set of players $\mathcal{A} \subseteq \mathcal{P}$, we define the characteristic set, $V(\mathcal{A})$, to be the set of all payoff vectors π such that there is a strategy vector $s \in \mathcal{S}_\mathcal{A}$, possibly dependent on π, with which the players in $\mathcal{A}$ can guarantee themselves payoffs of at least π.

We say that a payoff vector π dominates a payoff vector ν through a set $\mathcal{A} \subseteq \mathcal{P}$ if $\pi_a > \nu_a$ for all $a \in \mathcal{A}$. In other words, π dominates ν through $\mathcal{A}$ when each player in $\mathcal{A}$ does better in π than in ν.

For a set of players $\mathcal{A} \subseteq \mathcal{P}$, we define $D(\mathcal{A})$ as the set of all payoff vectors which are dominated through $\mathcal{A}$ by a payoff vector in $V(\mathcal{A})$. Interpreting, the players in $\mathcal{A}$ would never settle for a payoff vector $\pi \in D(\mathcal{A})$ since they can guarantee themselves higher payoffs than those offered in π.

The *core* of a game consists of all $\pi \in V(\mathcal{P})$ such that $\pi \notin D(\mathcal{A})$ for all $\mathcal{A} \subseteq \mathcal{P}$.

3 Characterization of the Core

In this section we show a characterization of the core of buyer-supplier games in the transferable utility and non-transferable utility settings. In Section 3.1, we give some preliminary lemmas for the games NOSP and SP. In Section 3.2, we show that the core of SP is the same as the core of NOSP. In Section 3.3, we give a characterization of the core of NOSP.

3.1 Lemmas True of Both NOSP and SP

This section contains some lemmas which are useful in characterizing the core of NOSP and SP.

Lemma 3.1. *Let $s \in \mathcal{S}_{\mathcal{A}\cup\mu}$ be such that $s_\mu = (\mathcal{A}, p_\mu)$. If s guarantees the players in $\mathcal{A} \cup \mu$ payoffs of at least $\pi \in \Re^{|\mathcal{A}\cup\mu|}$, then there is a $t \in \mathcal{S}_{\mathcal{A}\cup\mu}$ such that*

- $t_a = s_a$ *for all* $a \in C - \mathcal{A}$
- $t_\mu = (\mathcal{A}, 0)$
- *All side payments from players in $\mathcal{A} \cup \mu$ to players in $\mathcal{A} \cup \mu$ fixed to zero*
- *t also guarantees payoffs of at least π.*

Proof. We show how to sequentially remove the specified side payments while maintaining the payoff guarantee.

Let a and b be suppliers in $\mathcal{A}$. Let $s_a = (\beta(a), p_a)$ and $s_b = (\beta(b), p_b)$.

First, consider a supplier to supplier payment. Suppose that $p_a(b) = \lambda$, that is, supplier a pays λ to supplier b. Because both a and b are in $\mathcal{A}$, we can achieve the same utility transfer as the side payment by setting the side payment to zero and changing $\beta(a)$ to $\beta(a) - \lambda$ and $\beta(b)$ to $\beta(b) + \lambda$. Thus, we can zero out the side payment from a to b.

Now, consider a supplier to buyer payment. Suppose that $p_a(\mu) = \lambda$. In other words supplier a pays λ to the buyer. We can achieve the same utility transfer as the side payment by setting the side payment to zero and changing $\beta(a)$ to $\beta(a) - \lambda$. Thus, we can zero out the side payment from a to μ.

A similar change works for a payment from the buyer to a supplier.

□

Lemma 3.2. *Let strategy vector $s \in \mathcal{S}_{\mathcal{P}}$ realize payoff vector π. If $s_\mu = (\mathcal{A}, p_\mu)$ and there exists $a \in C - \mathcal{A}$ such that $\pi_a > 0$, then $\pi \in D(\mathcal{A} \cup \mu)$ in both* SP *and* NOSP.

Proof. Since all side payments are zero in NOSP, it is impossible for π_a to be greater than zero. Thus, the statement is trivial for NOSP.

Think of SP as a two stage distribution of wealth, where the strategy $s \in \mathcal{S}_{\mathcal{P}}$ determines the utility transfers. Initially, the buyer has M utility and all suppliers have zero utility. In the first stage, the buyer gives $\beta(b)$ to each supplier $b \in \mathcal{A}$ and loses an extra Bcost($\mathcal{A}$) from the initial M utility. Also in the first stage, each supplier $b \in \mathcal{A}$ loses $\tau(b)$ of utility. In the second stage, side payments are distributed.

Since there exists $a \in C - \mathcal{A}$ such that $\pi_a > 0$, in s there is a net flow of side payments from $\mathcal{A} \cup \mu$ to $C - \mathcal{A}$ in stage two of SP. Instead of following strategy s, the players in $\mathcal{A} \cup \mu$ can set to zero all side payments going from $\mathcal{A} \cup \mu$ to $C - \mathcal{A}$. With this action, at least π_a more utility stays in $\mathcal{A} \cup \mu$ at the end of stage two. The players in $\mathcal{A} \cup \mu$ can use side payments amongst themselves so that each player gets an extra $\pi_a/(|\mathcal{A}| + 1)$ utility at the end of stage two than what the player received from following strategy s. Moreover, since the players in $C - \mathcal{A}$ only have control over the nonnegative side payments flowing from $C - \mathcal{A}$ to $\mathcal{A} \cup \mu$, we have shown that the players in $\mathcal{A} \cup \mu$ can guarantee themselves payoffs greater than the payoffs that they received from following s. Thus, $\pi \in D(\mathcal{A} \cup \mu)$ in SP.

□

Lemma 3.3. *If π is in the core of* SP *or* NOSP, *then $\pi_a \geq 0$ for all $a \in \mathcal{P}$.*

Proof. We prove the statement by contradiction. Suppose that $\pi_a < 0$ for some player a.

If a is a supplier, a can guarantee at least 0 utility with strategy $(\tau(a), 0) \in \mathcal{S}_a$. If a is the buyer, a can guarantee 0 utility with strategy $(\emptyset, 0)$. Thus, $\pi \in D(\{a\})$ in both SP and NOSP. Thus, π is not in the core of either game.

□

Lemma 3.4. *Let π be a payoff vector, and let s be a strategy vector in $\mathcal{S}_{\mathcal{P}}$. If the players in $\mathcal{P}$ can guarantee themselves payoffs of at least π by playing s, but s does not realize π, then π is not in the core of either* SP *or* NOSP.

Proof. Let $s_\mu = (\mathcal{A}, p_\mu)$.

We use a proof by contradiction. Assume π is in the core of either SP or NOSP. By Lemma 3.3, we know that $\pi_a \geq 0$ for all $a \in \mathcal{P}$. By Lemma 3.2, we know that $\pi_a = 0$ for all $a \in C - \mathcal{A}$.

First, we derive a contradiction with the assumption that π is in the core of SP.

Think of SP as a two stage distribution of wealth, where the strategy $s \in \mathcal{S}_{\mathcal{P}}$ determines the utility transfers. Initially, the buyer has M utility and all suppliers have zero utility. In the first stage, the buyer gives $\beta(a)$ to each supplier $a \in \mathcal{A}$ and loses an extra Bcost($\mathcal{A}$) from the initial M utility. Also in the first stage, each supplier $a \in \mathcal{A}$ loses $\tau(a)$ of utility. In the second stage, side payments are distributed.

Since s guarantees payoffs of at least π but s does not realize π, we know that $\pi_a \leq u_a(s)$ for all $a \in \mathcal{P}$ and there is some $a \in \mathcal{P}$ such that $\pi_a < u_a(s)$. Thus by following s, the total wealth held by $\mathcal{A} \cup \mu$ in SP at the end of stage one is greater than $\sum_{a \in \mathcal{P}} \pi_a$. In turn, we have $\sum_{a \in \mathcal{P}} \pi_a \geq \sum_{a \in \mathcal{A} \cup \mu} \pi_a$. Let λ be the difference between the total wealth held by $\mathcal{A} \cup \mu$ at the end of stage one and $\sum_{a \in \mathcal{A} \cup \mu} \pi_a$.

Instead of following s, the players in $\mathcal{A} \cup \mu$ can set to zero all side payments from $\mathcal{A} \cup \mu$ to $C - \mathcal{A}$. The players in $\mathcal{A} \cup \mu$ can use side payments amongst themselves so that each player gets an extra $\lambda/(|\mathcal{A}| + 1)$ utility at the end of stage two than what the player received in π. Moreover, since the players in $C - \mathcal{A}$ only have control over the nonnegative side payments flowing from $C - \mathcal{A}$ to $\mathcal{A} \cup \mu$, we have shown that the players in $\mathcal{A} \cup \mu$ can guarantee themselves payoffs greater than the payoffs that they received in π. Thus, we have constructed a new strategy $t \in \mathcal{S}_{\mathcal{A} \cup \mu}$ in SP for the players in $\mathcal{A} \cup \mu$ which guarantees payoffs greater than π for each player in $\mathcal{A} \cup \mu$. Thus, $\pi \in D(\mathcal{A} \cup \mu)$ in SP. This contradicts the assumption that π is in the core of SP. Thus, π must be in the core of NOSP.

We now derive a contradiction with the assumption that π is in the core of NOSP. By Lemma 3.1 and the fact that t guarantees payoffs greater than π for each player in $\mathcal{A} \cup \mu$, we also have $\pi \in D(\mathcal{A} \cup \mu)$ in NOSP. This contradicts the assumption that π is in the core of NOSP.

□

3.2 Core Equivalence between SP and NOSP

In this section, we prove that the core of NOSP is the same as the core of SP.

Lemma 3.5. *Let π be a payoff vector. If $\pi \in D(\mathcal{A})$ in* NOSP*, then $\pi \in D(\mathcal{A})$ in* SP.

Proof. The players in $\mathcal{A}$ can follow exactly the same strategy in SP as they would in NOSP to guarantee payoffs greater than the payoffs in π; they simply fix all their side payments to zero.

□

Lemma 3.6. *Let π be a payoff vector. If $\pi \in V(\mathcal{P})$ in* SP *and π is in the core of* SP*, then $\pi \in V(\mathcal{P})$ in* NOSP.

Proof. By Lemma 3.4, π is realized by some strategy vector $s \in \mathcal{S}_{\mathcal{P}}$ in SP. Let $s_\mu = (\mathcal{A}, p_\mu)$.

Think of SP as a two stage distribution of wealth, where the strategy s determines the utility transfers. Initially, the buyer has M utility and all suppliers have zero utility. In the first stage, the buyer gives $\beta(b)$ to each supplier $b \in \mathcal{A}$ and loses an extra Bcost($\mathcal{A}$) from the initial M utility. Also in the first stage, each supplier $b \in \mathcal{A}$ loses $\tau(b)$ of utility. In the second stage, side payments are distributed.

By Lemma 3.2, we have $\pi_a = 0$ for all $a \in C - \mathcal{A}$. Thus, at the end of stage two of SP, there is no utility in $C - \mathcal{A}$. Thus, the players can achieve the same utility distribution by setting to zero all side payments except side payments from $\mathcal{A} \cup \mu$ to $\mathcal{A} \cup \mu$. Thus π is realized by a strategy vector with all zero side payments except the side payments from $\mathcal{A} \cup \mu$ to $\mathcal{A} \cup \mu$.

By Lemma 3.1, there is a strategy vector with all side payments fixed to zero which guarantees payoffs of at least π for all the players in $\mathcal{P}$. Thus $\pi \in V(\mathcal{P})$.

□

Lemma 3.7. *If payoff vector π is in the core of* SP*, then π is in the core of* NOSP.

Proof. The statement follows from Lemmas 3.6 and 3.5.

□

Lemma 3.8. *Let π be a payoff vector. If $\pi \in V(\mathcal{P})$ in* NOSP*, then $\pi \in V(\mathcal{P})$ in* SP.

Proof. By the definition of $V(\mathcal{P})$, the players in $\mathcal{P}$ can guarantee themselves payoffs of at least π by playing some strategy $s \in \mathcal{S}_{\mathcal{P}}$ in NOSP. The players in $\mathcal{P}$ can follow exactly the same strategy in SP to guarantee payoffs of at least π.

□

Lemma 3.9. *If payoff vector π is in the core of* NOSP*, then for all $\mathcal{A} \subseteq \mathcal{P}$ we have $\pi \notin D(\mathcal{A})$ in* SP.

Proof. We use a proof by contradiction. Suppose that $\pi \in D(\mathcal{A})$ in SP for some $\mathcal{A} \subseteq \mathcal{P}$. Thus, there is a strategy vector $s \in \mathcal{S}_{\mathcal{A}}$ in SP which guarantees each player in $\mathcal{A}$ a greater payoff than the payoff given in π.

Since π is in the core of NOSP, by Lemma 3.3 we know $\pi_a \geq 0$ for all $a \in \mathcal{A}$.

We split the proof into two cases. In the first case, suppose that $\mu \notin \mathcal{A}$. It is impossible for s to guarantee a payoff greater than 0 for any player in $\mathcal{A}$ since the buyer can always play $\emptyset$. Thus, we get a contradiction with the assumption that $\pi \in D(\mathcal{A})$ in SP.

For the second case, suppose $\mu \in \mathcal{A}$. Let $s_\mu = (\mathcal{B}, p_\mu)$. There can not be some supplier in $\mathcal{B}$ but not in $\mathcal{A}$ since that supplier can always play the strategy $(\lambda, 0)$ where $\lambda > M$ to give the buyer a negative payoff. Since $\pi_\mu \geq 0$, the existence of a supplier in $\mathcal{B} - \mathcal{A}$ contradicts the assumption that $\pi \in D(\mathcal{A})$ in SP.

Thus, we have $\mathcal{B} \subseteq \mathcal{A} - \{\mu\}$.

If there is some supplier a in $\mathcal{A}$ but not in $\mathcal{B}$, since $\pi_a \geq 0$, we know that s must guarantee a payoff greater than 0 for a. Let v be the payoff vector realized when the players in $\mathcal{A}$ follow s and each of the players in $\mathcal{P} - \mathcal{A}$ follow the strategy $(0, 0)$. Thus, we have $v_a > 0$ and $v_b > \pi_b$ for all $b \in \mathcal{A}$. By Lemma 3.2, we have $v \in D(\mathcal{B} \cup \mu)$ in SP. Since $v \in D(\mathcal{B} \cup \mu)$ in SP, $v_b > \pi_b$ for all $b \in \mathcal{A}$, and $\mathcal{B} \subseteq \mathcal{A} - \{\mu\}$, we have $\pi \in D(\mathcal{B} \cup \mu)$ in SP.

Thus, we have $s_\mu = (\mathcal{B}, p_\mu)$ and $\pi \in D(\mathcal{B} \cup \mu)$ in SP. By Lemma 3.1, we also have $\pi \in D(\mathcal{B} \cup \mu)$ in NOSP, which contradicts the fact that π is in the core of NOSP.

□

Lemma 3.10. *If payoff vector π is in the core of* NOSP*, then π is in the core of* SP.

Proof. The statement follows from Lemmas 3.8 and 3.9.

□

Theorem 3.11. *The core of* NOSP *is equal to the core of* SP.

Proof. Follows from Lemmas 3.7 and 3.10.

□

3.3 The Core of NOSP

In this section, we prove the the following theorem, which gives a characterization of the core of NOSP. In the theorem, the parameter τ is made explicit, though its value is clear from the definition of the buyer-supplier game, NOSP.

Theorem 3.12. *A payoff vector π is in the core of* NOSP *iff it satisfies*

$$\pi_a \geq 0 \qquad \textit{for all } a \in \mathcal{P} \tag{1}$$

$$\sum_{a \in \mathcal{A}} \pi_a \leq \text{MinEval}(C - \mathcal{A}, \tau) - \text{MinEval}(C, \tau) \qquad \textit{for all } \mathcal{A} \subseteq \ cit \tag{2}$$

$$\pi_\mu = M - \text{MinEval}(C, \tau) - \sum_{a \in C} \pi_a \tag{3}$$

Proof. The statement follows from Lemmas 3.17 and 3.21.

□

Lemma 3.13. *If a payoff vector π is in the core of* NOSP*, then $\pi_a \geq 0$ for all $a \in \mathcal{P}$.*

Proof. The lemma is a restatement of Lemma 3.3.

□

Lemma 3.14. *If a payoff vector π is realized by some strategy vector $s \in \mathcal{S}_\mathcal{P}$ where $s_\mu = (\mathcal{A}, 0)$, then $\pi_\mu = M - \text{Eval}(\mathcal{A}) - \sum_{a \in C} \pi_a$.*

Proof. For any $a \in C$ let $s_a = (\beta(a), p_a)$. Since π is realized by s and all side payments are zero, we have $\pi_a = \beta(a) - \tau(a)$ for all $a \in \mathcal{A}$. We also have that $\pi_\mu = M - [\text{Bcost}(\mathcal{A}) + \sum_{a\in\mathcal{A}} \beta(a)]$. Substituting for $\beta(a)$, we have $\pi_\mu = M - [\text{Bcost}(\mathcal{A}) + \sum_{a\in\mathcal{A}} \tau(a) + \sum_{a\in\mathcal{A}} \pi_a]$. By the definition of Eval, we have $\pi_\mu = M - \text{Eval}(\mathcal{A}) - \sum_{a\in\mathcal{A}} \pi_a$.

Since all side payments are fixed to zero, we have $\pi_a = 0$ for all $a \in C - \mathcal{A}$. Thus we can write $\pi_\mu = M - \text{Eval}(\mathcal{A}) - \sum_{a\in C} \pi_a$.

□

Lemma 3.15. *If a payoff vector π is in the core of* NOSP*, then $\pi_\mu = M - \text{MinEval}(C) - \sum_{a\in C} \pi_a$.*

Proof. Let $\mathcal{F} \subseteq C$ be such that $\text{Eval}(\mathcal{F}) = \text{MinEval}(C)$. We first show that any v realized by a strategy vector s with $s_\mu = (\mathcal{A}, 0)$ and $\text{Eval}(\mathcal{A}) \neq \text{Eval}(\mathcal{F})$ is not in the core.

Let $\lambda = (\text{Eval}(\mathcal{A}) - \text{Eval}(\mathcal{F}))/(|\mathcal{F}| + 1)$. Since $\text{Eval}(\mathcal{A}) \neq \text{Eval}(\mathcal{F})$ and by the definition of $\mathcal{F}$, we have $\lambda > 0$. Construct a strategy vector $t \in \mathcal{S}_{\mathcal{F}\cup\mu}$ where

$$\begin{aligned} t_\mu &= (\mathcal{F}, 0) \\ t_a &= (v_a + \tau(a) + \lambda, 0) \qquad\qquad \text{for all } a \in \mathcal{F}. \end{aligned}$$

Since side payments are fixed to zero, the suppliers in $C - \mathcal{F}$ have no strategies which can affect the payoffs of the players in $\mathcal{F} \cup \mu$ given that the players in $\mathcal{F} \cup \mu$ follow t. Let $u \in \mathcal{S}_{\mathcal{P}}$ be any strategy vector with projection onto $\mathcal{F} \cup \mu$ equal to t.

Straight forward calculations with the game's utility functions show that $u_a(u) - v_a = u_a(u) - u_a(s) = \lambda$ for each supplier $a \in \mathcal{F}$.

Consider

$$\begin{aligned} u_\mu(u) - v_\mu &= u_\mu(u) - u_\mu(s) \\ &= [M - \text{Bcost}(\mathcal{F}) - \sum_{a\in\mathcal{F}} (v_a + \tau(a) + \lambda)] - [M - \text{Bcost}(\mathcal{A}) - \sum_{a\in\mathcal{A}} (v_a + \tau(a))] \\ &= [-\text{Eval}(\mathcal{F}) - \sum_{a\in\mathcal{F}} (v_a + \lambda)] - [-\text{Eval}(\mathcal{A}) - \sum_{a\in\mathcal{A}} v_a] \\ &= \text{Eval}(\mathcal{A}) - \text{Eval}(\mathcal{F}) - \sum_{a\in\mathcal{F}} \lambda + [\sum_{a\in\mathcal{A}} v_a - \sum_{a\in\mathcal{F}} v_a]. \end{aligned}$$

By the utility functions of NOSP and the definition of $\mathcal{A}$ and v, we have $v_a = 0$ for all $a \in C - \mathcal{A}$. Thus, the bracketed quantity in the above expression is at least zero. Thus, we have

$$u_\mu(u) - v_\mu \geq \text{Eval}(\mathcal{A}) - \text{Eval}(\mathcal{F}) - \sum_{a\in\mathcal{F}} \lambda = \lambda$$

where the equality comes from the definition of λ.

Thus, we have $v \in D(\mathcal{F} \cup \mu)$.

We have shown that any vector in the core is realized by a strategy vector s with $s_\mu = (\mathcal{A}, 0)$ where $\text{Eval}(\mathcal{A}) = \text{Eval}(\mathcal{F})$. The lemma statement follows from Lemma 3.14 and the definition of $\mathcal{F}$.

□

Lemma 3.16. *If payoff vector π is in the core of* NOSP*, then $\sum_{a\in\mathcal{A}} \pi_a \leq \text{MinEval}(C - \mathcal{A}) - \text{MinEval}(C)$ for all $\mathcal{A} \subseteq C$.*

Proof. We use a proof by contradiction. Assume π is in the core and $\sum_{a\in\mathcal{A}} \pi_a > \text{MinEval}(C - \mathcal{A}) - \text{MinEval}(C)$ for some $\mathcal{A} \subseteq C$. We show that $\pi \in D(\mathcal{F} \cup \mu)$ where $\mathcal{F} \subseteq C - \mathcal{A}$ is such that $\text{Eval}(\mathcal{F}) = \text{MinEval}(C - \mathcal{A})$.

Since π is in the core, by Lemma 3.4 it is realized by some strategy vector $s \in \mathcal{S}_{\mathcal{P}}$

Let $\lambda = (\sum_{a\in\mathcal{A}} \pi_a - \text{Eval}(\mathcal{F}) + \text{MinEval}(C))/(|\mathcal{F}| + 1)$. Since $\sum_{a\in\mathcal{A}} \pi_a > \text{MinEval}(C - \mathcal{A}) - \text{MinEval}(C)$, we have $\lambda > 0$. Construct a strategy vector $t \in \mathcal{S}_{\mathcal{F}\cup\mu}$ where

$$\begin{aligned} t_\mu &= (\mathcal{F}, 0) \\ t_a &= (\pi_a + \tau(a) + \lambda, 0) && \text{for all } a \in \mathcal{F}. \end{aligned}$$

Since side payments are fixed to zero, the suppliers in $C - \mathcal{F}$ have no strategies which can affect the payoffs of the players in $\mathcal{F} \cup \mu$ given that the players in $\mathcal{F} \cup \mu$ follow t. Let $u \in \mathcal{S}_{\mathcal{P}}$ be any strategy vector with projection onto $\mathcal{F} \cup \mu$ equal to t.

Straight forward calculations with the game's utility functions show that $u_a(u) - \pi_a = u_a(u) - u_a(s) = \lambda$ for each supplier $a \in \mathcal{F}$.

Let $s_\mu = (\mathcal{B}, 0)$ and consider

$$\begin{aligned} u_\mu(u) - \pi_\mu &= u_\mu(u) - u_\mu(s) \\ &= [M - \text{Bcost}(\mathcal{F}) - \sum_{a\in\mathcal{F}} (\pi_a + \tau(a) + \lambda)] - [M - \text{Bcost}(\mathcal{B}) - \sum_{a\in\mathcal{B}} (\pi_a + \tau(a))] \\ &= [-\text{Eval}(\mathcal{F}) - \sum_{a\in\mathcal{F}} (\pi_a + \lambda)] - [\text{Eval}(\mathcal{B}) - \sum_{a\in\mathcal{B}} \pi_a] \\ &= \text{Eval}(\mathcal{B}) - \text{Eval}(\mathcal{F}) - \sum_{a\in\mathcal{F}} \lambda + [\sum_{a\in\mathcal{B}} \pi_a - \sum_{a\in\mathcal{F}} \pi_a] \end{aligned}$$

By the utility functions of NOSP and the definitions of $\mathcal{B}$ and π, we have $\pi_a = 0$ for all $a \in C - \mathcal{B}$. Thus, $\sum_{a\in\mathcal{B}} \pi_a = \sum_{a\in C} \pi_a$. Thus, we can write

$$\begin{aligned} u_\mu(u) - u_\mu(s) &= \text{Eval}(\mathcal{B}) - \text{Eval}(\mathcal{F}) - \sum_{a\in\mathcal{F}} \lambda + [\sum_{a\in C} \pi_a - \sum_{a\in\mathcal{F}} \pi_a] \\ &= \text{Eval}(\mathcal{B}) - \text{Eval}(\mathcal{F}) - \sum_{a\in\mathcal{F}} \lambda + \sum_{a\in\mathcal{A}} \pi_a + [\sum_{a\in C-\mathcal{A}} \pi_a - \sum_{a\in\mathcal{F}} \pi_a] \end{aligned}$$

Since $\mathcal{F} \subseteq C - \mathcal{A}$, we know that the bracketed quantity in the above expression is at least zero. Thus, we have

$$u_\mu(u) - u_\mu(s) \geq \text{Eval}(\mathcal{B}) - \text{Eval}(\mathcal{F}) + \sum_{a\in\mathcal{A}} \pi_a - \sum_{a\in\mathcal{F}} \lambda = \lambda$$

where the equality comes from the definition on λ.

Thus, we have $\pi \in D(\mathcal{F} \cup \mu)$, which contradicts the fact that π is in the core of NOSP.

□

Lemma 3.17. *Payoff vectors in the core of* NOSP *satisfy Equations (1), (2), and (3).*

Proof. The statement follows from Lemmas 3.13, 3.16, and 3.15.

□

Lemma 3.18. *If payoff vector π satisfies Equations (1), (2), and (3) then $\pi \notin D(\mathcal{A})$ for $\mathcal{A} \subseteq \mathcal{P}$ such that $\mu \notin \mathcal{A}$.*

Proof. We use a proof by contradiction. Suppose $\pi \in D(\mathcal{A})$. In other words, the players in $\mathcal{A}$ can guarantee payoffs greater than the payoffs given in π. But, we know that $\pi_a \geq 0$ for all $a \in \mathcal{A}$ and the players in $\mathcal{A}$ can only guarantee 0 payoffs because the buyer can always play $(\emptyset, 0)$. Thus, we have a contradiction with $\pi \in D(\mathcal{A})$.

□

Lemma 3.19. *If payoff vector π satisfies Equations (1), (2), and (3) then $\pi \notin D(\mathcal{A})$ for $\mathcal{A} \subseteq \mathcal{P}$ such that $\mu \in \mathcal{A}$.*

Proof. We use a proof by contradiction. Suppose $\pi \in D(\mathcal{A})$ for $\mathcal{A} \subseteq \mathcal{P}$ such that $\mu \in \mathcal{A}$. Thus, the players in $\mathcal{A}$ can follow a strategy $s \in \mathcal{S}_{\mathcal{A}}$ that guarantees payoffs greater than the payoffs they are given in π.

Let $t \in \mathcal{S}_{\mathcal{P}}$ be any strategy vector with projection onto $\mathcal{A}$ equal to s. Let $t_\mu = (\mathcal{B}, 0)$. Let the payoff vector realized by t be ν.

Since π satisfies Equation (1), we have $\pi_\mu \geq 0$. Suppose the players in $\mathcal{A}$ follow strategy s. If there is some $a \in \mathcal{B} - \mathcal{A}$, then the bid of supplier a must be bounded so as to guarantee the buyer a payoff greater than 0 when the players in $\mathcal{A}$ follow s. This contradicts contradicts the fact that $\pi \in D(\mathcal{A})$. Thus, we have that there is no such a and $\mathcal{B} \subseteq \mathcal{A}$.

Since π satisfies Equation (3) we have $\pi_\mu = M - \text{MinEval}(C) - \sum_{a \in C} \pi_a$. By Lemma 3.14, we have $\nu_\mu = M - \text{Eval}(\mathcal{B}) - \sum_{a \in C} \nu_a$.

Since following s guarantees a payoff greater than the payoff given in π for every player in $\mathcal{A}$, we have $\pi_\mu < \nu_\mu$. Thus, we have

$$\begin{aligned}0 &< \nu_\mu - \pi_\mu \\ &= M - \text{Eval}(\mathcal{B}) - \sum_{a \in C} \nu_a - [M - \text{MinEval}(C) - \sum_{a \in C} \pi_a] \\ &= \text{MinEval}(C) - \text{Eval}(\mathcal{B}) + \sum_{a \in C} \pi_a - \sum_{a \in C} \nu_a\end{aligned}$$

Let $\mathcal{F} \subseteq C$ be such that $\text{Eval}(\mathcal{F}) = \text{MinEval}(C)$. From Equation (2) with the singleton sets, we have that $\pi_a = 0$ for all $a \notin \mathcal{F}$. From the definition of ν, we have that $\nu_a = 0$ for all $a \notin \mathcal{B}$. Let $\mathcal{U} = \mathcal{A} - \{\mu\}$. Thus, we have

$$\begin{aligned}0 &< \text{MinEval}(C) - \text{Eval}(\mathcal{B}) + \sum_{a \in \mathcal{F}} \pi_a - \sum_{a \in \mathcal{B}} \nu_a \\ &= \text{MinEval}(C) - \text{Eval}(\mathcal{B}) + \sum_{a \in \mathcal{F} - \mathcal{U}} \pi_a + (\sum_{a \in \mathcal{F} \cap \mathcal{U}} \pi_a - \sum_{a \in \mathcal{F} \cap \mathcal{B}} \nu_a) - \sum_{a \in \mathcal{B} - \mathcal{F}} \nu_a\end{aligned}$$

By the definition of ν and the utility functions in NOSP, we have that ν_a for all $a \in C - \mathcal{B}$. By Equation (1), we have $\pi_a \geq 0$ for all $a \in C$. By the definition of ν and the fact that $\nu \subseteq \pi$, we also have $\nu_a > \pi_a \geq 0$ for all $a \in \mathcal{B}$. Thus, the last term in the above expression at least zero. Thus, we have

$$0 < \text{MinEval}(C) - \text{Eval}(\mathcal{B}) + \sum_{a \in \mathcal{F} - \mathcal{U}} \pi_a + (\sum_{a \in \mathcal{F} \cap \mathcal{U}} \pi_a - \sum_{a \in \mathcal{F} \cap \mathcal{U}} \nu_a)$$

By the definitions of ν and $\mathcal{U}$, we also have that $\nu_a > \pi_a$ for all $a \in \mathcal{U}$. Thus, the parenthesized term in the above expression is at least zero. Thus, we have

$$0 < \text{MinEval}(C) - \text{Eval}(\mathcal{B}) + \sum_{a \in \mathcal{F} - \mathcal{A}} \pi_a$$

$$\text{Eval}(\mathcal{B}) - \text{MinEval}(C) < \sum_{a \in \mathcal{F} - \mathcal{U}} \pi_a$$

Let $\mathcal{K} = \mathcal{F} - \mathcal{U}$. Since $\mathcal{B} \subseteq \mathcal{A}$ and $\mathcal{B} \subseteq C$, we have $\mathcal{B} \subseteq \mathcal{U}$. Thus, we have $\mathcal{B} \subseteq C - \mathcal{K}$. By the definition of MinEval, we have $\text{MinEval}(C - \mathcal{K}) \leq \text{Eval}(\mathcal{B})$. Thus, we have

$$\text{MinEval}(C - \mathcal{K}) - \text{MinEval}(C) \leq \text{Eval}(\mathcal{B}) - \text{MinEval}(C) < \sum_{a \in \mathcal{K}} \pi_a.$$

This statement contradicts the fact that π satisfies Equation (2) for $\mathcal{K}$.

□

Lemma 3.20. *If payoff vector π satisfies Equations (1), (2), and (3), then $\pi \in V(\mathcal{P})$ in* NOSP.

Proof. Let $\mathcal{F} \subseteq C$ be such that $\text{Eval}(\mathcal{F}) = \text{MinEval}(C)$. Define $s \in \mathcal{S}_{\mathcal{P}}$ such that

$$\begin{aligned} s_\mu &= (\mathcal{F}, 0) && \\ s_a &= (\pi_a + \tau(a), 0) && \text{for all } a \in \mathcal{F} \\ s_a &= (0, 0) && \text{for all } a \in C - \mathcal{F} \end{aligned}$$

Straight forward calculations with the game's utility functions show that $u_a(s) = \pi_a$ for each supplier $a \in \mathcal{F}$.
Consider

$$\begin{aligned} u_\mu(s) &= M - [\text{Bcost}(\mathcal{F}) + \sum_{a \in \mathcal{F}} (\pi_a + \tau(a))] \\ &= M - [\text{Eval}(\mathcal{F}) + \sum_{a \in \mathcal{F}} \pi_a] \\ &= M - \text{Eval}(\mathcal{F}) - \sum_{a \in \mathcal{F}} \pi_a \end{aligned}$$

Since π satisfies Equation (2), we have $\pi_a = 0$ for all $a \in C - \mathcal{F}$. Thus, we have

$$\begin{aligned} u_\mu(s) &= M - \text{Eval}(\mathcal{F}) - \sum_{a \in C} \pi_a \\ &= M - \text{MinEval}(C) - \sum_{a \in C} \pi_a \\ &= \pi_\mu \end{aligned}$$

where the second equality comes from the definition of $\mathcal{F}$ and the last equality comes from the fact that π satisfies Equation (3).

Finally, we have $u_a(s) = \pi_a$ for each supplier $a \in C - \mathcal{F}$, since $\pi_a = 0$ for such a.

Thus, s realizes π and $\pi \in V(\mathcal{P})$.

□

Lemma 3.21. *If payoff vector π satisfies Equations (1), (2), and (3) then π is in the core of* NOSP.

Proof. The statement follows from Lemmas 3.18 and 3.19.

□

Lemma 3.22. *Let* $\text{Bcost}^*(\mathcal{A}) = \sum_{a \in \mathcal{A}} \tau(a) + \text{Bcost}(\mathcal{A})$. *The core of the buyer-supplier games defined by* (C, τ, Bcost) *and* $(C, 0, \text{Bcost}^*)$ *is the same.*

Proof. By the definition of Eval and Bcost*, we have $\text{Eval}(\mathcal{B}, \tau, \text{Bcost}) = \text{Eval}(\mathcal{B}, 0, \text{Bcost}^*)$ for all $\mathcal{B} \subseteq C$. Thus, we also have $\text{MinEval}(\mathcal{A}, \tau, \text{Bcost}) = \text{MinEval}(\mathcal{A}, 0, \text{Bcost}^*)$ for all $\mathcal{A} \subseteq C$. The result follows from Theorem 3.12.

□

4 Polynomial Time Optimization Over the Core Vectors

We define the separation problem on a set of linear inequalities $\mathcal{A}$ as follows. Given a vector π, if π satisfies all of the inequalities in $\mathcal{A}$, then do nothing; otherwise, output a violated inequality $a \in \mathcal{A}$. It is well known that the separation problem is polynomial time equivalent to linear function optimization over the same set of inequalities [11, p. 161].

Let (C, τ, Bcost) define a buyer-supplier game. In this section, to simplify the notation, we will omit the parameter Bcost from Eval and MinEval since it is fixed by the buyer-supplier game.

In this section, we will analyze an algorithm to solve the separation problem for Equations (1), (2), and (3). We now give the algorithm, which we call the separation algorithm. Given the payoff vector π as input,

1 Iterate over Equations (1) and (3) to check that they hold. If some equation does not hold, output that equation and halt.
2 Compute $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \tau) = \text{MinEval}(C, \tau)$. If there is some $a \in C - \mathcal{F}$ with $\pi_a > 0$, output the inequality from Equation (2) corresponding to $\{a\}$ and halt.
3 Define $\hat{\tau}(a) = \tau(a) + \pi_a$ for $a \in C$. Now, compute $\hat{\mathcal{F}} \subseteq C$ such that $\text{Eval}(\hat{\mathcal{F}}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$. If $\text{Eval}(\hat{\mathcal{F}}, \hat{\tau}) < \text{Eval}(\mathcal{F}, \hat{\tau})$, output the inequality from Equation (2) corresponding to $\mathcal{F} - \hat{\mathcal{F}}$. Otherwise, halt.

Theorem 4.1. *If given an input $\hat{\tau} : C \to \Re_+$ it is possible to compute both* $\text{Eval}(\mathcal{A}, \hat{\tau})$ *for any $\mathcal{A} \subseteq C$ and $\mathcal{F} \subseteq C$ such that* $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ *in polynomial time, then the separation problem for Equations (1), (2), and (3) is solvable in polynomial time. By the equivalence of separation and optimization, optimizing any linear function of π over Equations (1), (2), and (3) is also possible in polynomial time.*

Proof. The statement follows from Lemmas 4.2, 4.3, and 4.4. □

Lemma 4.2. *If on input $\hat{\tau} : C \to \Re_+$ it is possible to compute both* $\text{Eval}(\mathcal{A}, \hat{\tau})$ *for any $\mathcal{A} \subseteq C$ and $\mathcal{F} \subseteq C$ such that* $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ *in polynomial time, the separation algorithm runs in polynomial time.*

Proof. Iterating over Equations (1) and (3) takes polynomial time since there are $O(|C|)$ equations to check. Thus, step 1 of the algorithm completes in polynomial time.

By the lemma assumption, computing $\mathcal{F}$ takes polynomial time. Checking that for each $a \in C - \mathcal{F}$, π_a at most zero takes polynomial time since there are at most $|C|$ such checks. Thus, step 2 of the algorithm completes in polynomial time.

Defining $\hat{\tau}()$ takes polynomial time since there are $O(|C|)$ possible inputs to $\hat{\tau}$. By the lemma assumption, computing $\hat{\mathcal{F}}$ takes polynomial time. By the lemma assumption, computing $\text{Eval}(\hat{\mathcal{F}}, \hat{\tau})$ and $\text{Eval}(\mathcal{F}, \hat{\tau})$ takes polynomial time. Finally, computing $\mathcal{F} - \hat{\mathcal{F}}$ takes polynomial time since each set has at most $|C|$ elements. Thus both the run time of step 3 and the overall run time is polynomial. □

Lemma 4.3. *If the separation algorithm returns an inequality on input π, then π violates the returned inequality.*

Proof. If the algorithm returns an inequality in step 1, then the inequality is violated since the algorithm performed a direct check.

If the algorithm returns an inequality in step 2, then the inequality is violated since $\pi_a > 0$, but $\text{MinEval}(C - a, \tau) = \text{MinEval}(C, \tau) = \text{Eval}(\mathcal{F}, \tau)$.

Suppose the algorithm returns an inequality in step 3. Thus,

$$\begin{aligned}\mathrm{Eval}(\hat{\mathcal{F}}, \hat{\tau}) &< \mathrm{Eval}(\mathcal{F}, \hat{\tau}) \\ \sum_{a\in\hat{\mathcal{F}}} \hat{\tau}(a) + \mathrm{Bcost}(\hat{\mathcal{F}}) &< \sum_{a\in\mathcal{F}} \hat{\tau}(a) + \mathrm{Bcost}(\mathcal{F}) \\ \sum_{a\in\hat{\mathcal{F}}} \pi_a + \sum_{a\in\hat{\mathcal{F}}} \tau(a) + \mathrm{Bcost}(\hat{\mathcal{F}}) &< \sum_{a\in\mathcal{F}} \pi_a + \sum_{a\in\mathcal{F}} \tau(a) + \mathrm{Bcost}(\mathcal{F}) \\ \sum_{a\in\hat{\mathcal{F}}} \pi_a + \mathrm{Eval}(\hat{\mathcal{F}}, \tau) &< \sum_{a\in\mathcal{F}} \pi_a + \mathrm{Eval}(\mathcal{F}, \tau)\end{aligned}$$

Since the algorithm reaches step 3, we know that $\pi_a = 0$ for all $a \in C - \mathcal{F}$. Thus,

$$\begin{aligned}\sum_{a\in\hat{\mathcal{F}}\cap\mathcal{F}} \pi_a + \mathrm{Eval}(\hat{\mathcal{F}}, \tau) &< \sum_{a\in\mathcal{F}} \pi_a + \mathrm{Eval}(\mathcal{F}, \tau) \\ \mathrm{Eval}(\hat{\mathcal{F}}, \tau) - \mathrm{Eval}(\mathcal{F}, \tau) &< \sum_{a\in\mathcal{F}-\hat{\mathcal{F}}} \pi_a\end{aligned}$$

Let $\mathcal{A} = \mathcal{F} - \hat{\mathcal{F}}$. From the algorithm, we know that the set $\mathcal{F}$ satisfies $\mathrm{Eval}(\mathcal{F}, \tau) = \mathrm{MinEval}(C, \tau)$. Since $\hat{\mathcal{F}} \subseteq C - \mathcal{A}$, by definition of MinEval, we know that $\mathrm{MinEval}(C - \mathcal{A}, \tau) \leq \mathrm{Eval}(\hat{\mathcal{F}}, \tau)$. Thus, we have

$$\mathrm{MinEval}(C - \mathcal{A}, \tau) - \mathrm{MinEval}(C, \tau) \leq \mathrm{Eval}(\hat{\mathcal{F}}, \tau) - \mathrm{Eval}(\mathcal{F}, \tau) < \sum_{a\in\mathcal{A}} \pi_a$$

Thus, we have shown that inequality output by the algorithm is violated.

□

Lemma 4.4. *If π violates some inequality in Equations (1), (2), and (3), then the separation algorithm run on input π returns an inequality.*

Proof. If the violation is in Equations (1) or (3), the violated inequality will be output by the direct check in step 1.

If some inequality is output by step 2, we are done. Otherwise, since steps 1 and 2 output no inequality, we know that $\pi_a = 0$ for all $a \in C - \mathcal{F}$, where $\mathcal{F}$ is as computed in the algorithm.

Now, suppose the inequality from Equation (2) for set $\mathcal{A} \subseteq C$ is violated. In other words, we have, $\sum_{a\in\mathcal{A}} \pi_a > \mathrm{MinEval}(C - \mathcal{A}, \tau) - \mathrm{MinEval}(C, \tau)$. Let $\mathcal{B}$ be such that $\mathrm{Eval}(\mathcal{B}, \tau) = \mathrm{MinEval}(C - \mathcal{A}, \tau)$.

Thus, we have

$$\sum_{a\in\mathcal{A}} \pi_a > \mathrm{MinEval}(C - \mathcal{A}, \tau) - \mathrm{MinEval}(C, \tau) = \mathrm{Eval}(\mathcal{B}, \tau) - \mathrm{Eval}(\mathcal{F}, \tau)$$

Since $\pi_a = 0$ for all $a \in C - \mathcal{F}$.

$$\mathrm{Eval}(\mathcal{F}, \tau) + \sum_{a\in\mathcal{F}\cap\mathcal{A}} \pi_a > \mathrm{Eval}(\mathcal{B}, \tau)$$

Adding $\sum_{a\in\mathcal{F}-\mathcal{A}} \pi_a$ to both sides, we have

$$\begin{aligned}\mathrm{Eval}(\mathcal{F}, \tau) + \sum_{a\in\mathcal{F}} \pi_a &> \mathrm{Eval}(\mathcal{B}, \tau) + \sum_{a\in\mathcal{F}-\mathcal{A}} \pi_a \\ \mathrm{Bcost}(\mathcal{F}) + \sum_{a\in\mathcal{F}} \tau(a) + \sum_{a\in\mathcal{F}} \pi_a &> \mathrm{Bcost}(\mathcal{B}) + \sum_{a\in\mathcal{B}} \tau(a) + \sum_{a\in\mathcal{F}-\mathcal{A}} \pi_a\end{aligned}$$

Since $\pi_a = 0$ for all $a \in C - \mathcal{F}$ and $\mathcal{B} \subseteq C - \mathcal{A}$, we have

$$\text{Bcost}(\mathcal{F}) + \sum_{a \in \mathcal{F}} \tau(a) + \sum_{a \in \mathcal{F}} \pi_a > \text{Bcost}(\mathcal{B}) + \sum_{a \in \mathcal{B}} \tau(a) + \sum_{a \in \mathcal{B}} \pi_a + \sum_{a \in \mathcal{F} - \mathcal{A} - \mathcal{B}} \pi_a$$

By the definition of $\hat{\tau}$, we have

$$\begin{aligned}\text{Bcost}(\mathcal{F}) + \sum_{a \in \mathcal{F}} \hat{\tau}(a) &> \text{Bcost}(\mathcal{B}) + \sum_{a \in \mathcal{B}} \hat{\tau}(a) + \sum_{a \in \mathcal{F} - \mathcal{A} - \mathcal{B}} \pi_a \\ \text{Eval}(\mathcal{F}, \hat{\tau}) &> \text{Eval}(\mathcal{B}, \hat{\tau}) + \sum_{a \in \mathcal{F} - \mathcal{A} - \mathcal{B}} \pi_a\end{aligned}$$

We know that $\pi_a \geq 0$ for all $a \in \mathcal{P}$ since the algorithm does not output anything in step 1. Thus,

$$\text{Eval}(\mathcal{F}, \hat{\tau}) > \text{Eval}(\mathcal{B}, \hat{\tau})$$

Finally, we have

$$\text{Eval}(\mathcal{F}, \hat{\tau}) > \text{Eval}(\mathcal{B}, \hat{\tau}) \geq \text{MinEval}(C, \hat{\tau}) = \text{Eval}(\hat{\mathcal{F}}, \hat{\tau})$$

where $\hat{\mathcal{F}}$ is as computed in the algorithm. Thus, step 3 will output an inequality.

□

5 Inapproximability of Optimization Over Core Solutions

Consider a buyer-supplier game defined by (C, τ, Bcost). We introduced the concept of the focus point price in the introduction. The concept leads us to ask the natural question: What is the difference between the best and worst core outcome for the buyer? In other words, the value of interest is the solution to the linear program

$$\begin{aligned}&\max \sum_{a \in C} \pi_a \\ &\text{s.t.} \sum_{a \in \mathcal{A}} \pi_a \leq \text{MinEval}(C - \mathcal{A}, \tau) - \text{MinEval}(C, \tau) && \text{for all } \mathcal{A} \subseteq C \\ &\qquad \pi_\mu = M - \text{MinEval}(C, \tau) - \sum_{a \in C} \pi_a \\ &\qquad \pi_a \geq 0 && \text{for all } a \in \mathcal{P}.\end{aligned}$$

This natural question leads us to define the focus point price (FFP) problem as follows: on input (C, τ, Bcost), output the optimal value of the afore mentioned linear program.

Define the Necessary Element (NEL) problem as follows. On input (C, τ, Bcost) return TRUE if there exist an element $a \in C$ such that for all $\mathcal{F} \subseteq C$ satisfying $\text{Eval}(\mathcal{F}, \tau, \text{Bcost}) = \text{MinEval}(C, \tau, \text{Bcost})$ we have $a \in \mathcal{F}$. Otherwise, return FALSE.

Define the OPT-SET problem as follows. On input (C, τ, Bcost), return $\mathcal{F}$ such that $\text{Eval}(\mathcal{F}, \tau, \text{Bcost}) = \text{MinEval}(C, \tau, \text{Bcost})$.

We will prove a polynomial time equivalence between the NEL problem, the OPT-SET problem, and the separation problem over Equations (1), (2), and (3). In Section 5.1, we show how to solve the OPT-SET problem in polynomial time if the NEL problem is solvable in polynomial time. In Section 5.2, we show the polynomial time equivalence of NEL, OPT-SET, and separation over Equations (1), (2), and (3).

5.1 Polynomial Time Reduction from OPT-SET to NEL

In this section, we show that given a polynomial time algorithm to solve the NEL problem, we can solve the OPT-SET problem in polynomial time.

For a fixed tuple (C, τ, Bcost) we say we extend the tuple to contain a *shadow element* for an element $a \subseteq C$ by creating the extended tuple $(\hat{C}, \hat{\tau}, \text{Bcost}^*)$, where $\hat{C} = C \cup b$ with $b \notin C$; $\hat{\tau}$ is the same as τ with the addition that $\hat{\tau}(b) = \tau(a)$; and for $\mathcal{A} \subseteq \hat{C}$, if $b \notin \mathcal{A}$, then $\text{Bcost}^*(\mathcal{A}) = \text{Bcost}(\mathcal{A})$, otherwise $\text{Bcost}^*(\mathcal{A}) = \text{Bcost}((\mathcal{A} - \{b\}) \cup \{a\})$. We call b the *shadow element* corresponding to a.

The *full shadow extension* of (C, τ, Bcost) is the tuple $(\hat{C}, \hat{\tau}, \text{Bcost}^*)$ resulting from extending (C, τ, Bcost) to contain a shadow element for each element in C.

First, we reduce OPT-SET to NEL. To show the result, we analyze the following algorithm, which we call the shadow algorithm.

On input (C, τ, Bcost),

1 Let $(\hat{C}^*, \hat{\tau}, \text{Bcost}^*)$ be the full shadow extension of (C, τ, Bcost). Let the program variable $\hat{C}$ equal $\hat{C}^*$.

2 For each $a \in C$

- Remove a's corresponding shadow element from $\hat{C}$.
- Run NEL on $(\hat{C}, \hat{\tau}, \text{Bcost}^*)$.
- If the return value is TRUE, then add the shadow element back to $\hat{C}$.
- If the return value is FALSE, then remove a from $\hat{C}$.

3 Return $\hat{C} \cap C$. In other words, we return all elements from C remaining in $\hat{C}$, disregarding any shadow elements.

Lemma 5.1. *Let* (C, τ, Bcost) *be the input to the shadow algorithm. Let* $(\hat{C}^*, \hat{\tau}, \text{Bcost}^*)$ *be the full shadow extension of* (C, τ, Bcost). *If for all* $\mathcal{A} \subseteq \hat{C}^*$ *the NEL problem on input* $(\mathcal{A}, \hat{\tau}, \text{Bcost}^*)$ *is solvable in polynomial time, then the shadow algorithm runs in polynomial time.*

Proof. Constructing $\hat{C}^*$ takes polynomial time since there are $O(|C|)$ elements. Defining $\hat{\tau}$ takes polynomial time since there are $O(|C|)$ inputs. Queries to Bcost^* can be implemented with polynomial overhead on top of queries to Bcost. Thus, the initialization step of the algorithm takes polynomial time.

Consider a single loop iteration. The first, third and forth lines of the loop each take $O(|C|)$ time. The second step takes polynomial time by the lemma assumption. Thus, a single loop iteration takes polynomial time.

There are $|C|$ loop iterations and computing the intersection in the algorithm's final step takes $O(|C|)$ time. Thus the algorithm runs in polynomial time.

□

Lemma 5.2. *The shadow algorithm maintains the invariant* $\text{MinEval}(C, \tau, \text{Bcost}) = \text{MinEval}(\hat{C}, \hat{\tau}, \text{Bcost}^*)$.

Proof. Initially, $\text{MinEval}(C, \tau, \text{Bcost}) = \text{MinEval}(\hat{C}, \hat{\tau}, \text{Bcost}^*)$ by the definitions of $\hat{C}$, $\hat{\tau}$, and Bcost^*.

Consider the iteration of the loop associated with $a \in C$. Let the corresponding shadow element be b. When we remove or add b to $\hat{C}$, we have $\text{MinEval}(C, \tau, \text{Bcost}) = \text{MinEval}(\hat{C}, \hat{\tau}, \text{Bcost}^*)$ by the definitions of $\hat{\tau}$, and Bcost^* and the fact that a is still in $\hat{C}$.

We only remove both a and b if NEL returned FALSE before the removal of a. Let $\hat{C}_a$ and $\hat{C}'_a$ be the value of the variable $\hat{C}$ before and after the removal of a, respectively. Since NEL returned FALSE on $(\hat{C}_a, \hat{\tau}, \text{Bcost}^*)$, there exists some $\mathcal{F} \subseteq \hat{C}_a$ such that $a \notin \mathcal{F}$ and $\text{Eval}(\mathcal{F}, \hat{\tau}, \text{Bcost}^*) = \text{MinEval}(\hat{C}_a, \hat{\tau}, \text{Bcost}^*)$. Thus, $\mathcal{F} \subseteq \hat{C}'_a$ and $\text{MinEval}(\hat{C}_a, \hat{\tau}, \text{Bcost}^*) = \text{MinEval}(\hat{C}'_a, \hat{\tau}, \text{Bcost}^*)$.

Thus, throughout the algorithm the value of $\text{MinEval}(\hat{C}, \hat{\tau}, \text{Bcost}^*)$ does not change, which concludes the proof.

□

Lemma 5.3. *Let $\hat{C}_a$ be the value of the variable $\hat{C}$ at the end of iteration corresponding to $a \in C$. If $a \in \hat{C}_a$, then a is in all OPT-SET solutions on input $(\mathcal{A}, \hat{\tau}, \text{Bcost}^*)$ where $\mathcal{A} = \hat{C}_a \cap C$.*

Proof. Let the arguments of the NEL problem which is solved during the iteration corresponding to a be $(\mathcal{B}, \hat{\tau}, \text{Bcost}^*)$.

Since $a \in \hat{C}_a$, NEL returns TRUE during the iteration corresponding to a.

Suppose there is a solution $\mathcal{F} \subseteq C$ to the OPT-SET problem on input $(\mathcal{A}, \hat{\tau}, \text{Bcost}^*)$ which does not contain a. Consider $\mathcal{F}$ and $\hat{\mathcal{F}}$ where $\hat{\mathcal{F}}$ contains all the shadow elements of the elements of $\mathcal{F}$. The sets $\mathcal{F}$ and $\hat{\mathcal{F}}$ are disjoint and both subsets of $\mathcal{B}$. Also, by the definition of $\mathcal{F}$, $\hat{\tau}$ and Bcost^*, $\text{Eval}(\mathcal{F}, \hat{\tau}, \text{Bcost}^*) = \text{Eval}(\hat{\mathcal{F}}, \hat{\tau}, \text{Bcost}^*) = \text{MinEval}(\mathcal{B}, \hat{\tau}, \text{Bcost}^*)$. Thus, the NEL problem run during the iteration corresponding to a should return FALSE, which is a contradiction.

□

Lemma 5.4. *Let $\hat{C}_a$ be the value of the variable $\hat{C}$ at the end of iteration corresponding to $a \in C$. We have* $\text{MinEval}(C, \tau, \text{Bcost}) = \text{MinEval}(\hat{C}_a \cap C, \tau, \text{Bcost})$.

Proof. Let $\mathcal{F}$ be such that $\text{Eval}(\mathcal{F}, \hat{\tau}, \text{Bcost}^*) = \text{MinEval}(\hat{C}_a, \hat{\tau}, \text{Bcost}^*)$. If $\text{Eval}(\mathcal{F}, \hat{\tau}, \text{Bcost}^*) = M$, then let $\hat{\mathcal{F}} = \emptyset$; otherwise, let $\hat{\mathcal{F}}$ be $\mathcal{F}$ with each shadow element replaced by the corresponding element in C. By the definitions of $\hat{\tau}$ and Bcost^*, we have $\text{Eval}(\mathcal{F}, \hat{\tau}, \text{Bcost}^*) = \text{Eval}(\hat{\mathcal{F}}, \hat{\tau}, \text{Bcost}^*)$. But, by the construction of $\hat{\mathcal{F}}$, we have $\hat{\mathcal{F}} \subseteq \hat{C}_a \cap C$.

Thus, $\text{MinEval}(\hat{C}_a, \hat{\tau}, \text{Bcost}^*) = \text{Eval}(\mathcal{F}, \hat{\tau}, \text{Bcost}^*) = \text{Eval}(\hat{\mathcal{F}}, \hat{\tau}, \text{Bcost}^*) \geq \text{MinEval}(\hat{C}_a \cap C, \hat{\tau}, \text{Bcost}^*)$. Also, by the definition of MinEval, we have $\text{MinEval}(\hat{C}_a, \hat{\tau}, \text{Bcost}^*) \leq \text{MinEval}(\hat{C}_a \cap C, \hat{\tau}, \text{Bcost}^*)$. So, we have $\text{MinEval}(\hat{C}_a, \hat{\tau}, \text{Bcost}^*) = \text{MinEval}(\hat{C}_a \cap C, \hat{\tau}, \text{Bcost}^*)$.

Combining Lemma 5.2 with the result from the last paragraph, we have

$$\text{MinEval}(C, \tau, \text{Bcost}) = \text{MinEval}(\hat{C}_a, \hat{\tau}, \text{Bcost}^*) = \text{MinEval}(\hat{C}_a \cap C, \hat{\tau}, \text{Bcost}^*).$$

Finally, by the definition of $\hat{\tau}$ and Bcost^*, we have $\text{MinEval}(\hat{C}_a \cap C, \hat{\tau}, \text{Bcost}^*) = \text{MinEval}(\hat{C}_a \cap C, \tau, \text{Bcost})$.

□

Lemma 5.5. *Let (C, τ, Bcost) be the input to the shadow algorithm. Let $(\hat{C}^*, \hat{\tau}, \text{Bcost}^*)$ be the full shadow extension of (C, τ, Bcost). If for all $\mathcal{A} \subseteq \hat{C}^*$ the NEL problem on input $(\mathcal{A}, \hat{\tau}, \text{Bcost}^*)$ is solvable in polynomial time, then the OPT-SET problem on input (C, τ, Bcost) is solvable in polynomial time.*

Proof. By Lemma 5.1, the shadow algorithm runs in polynomial time.

Let $\hat{C}'$ be the value of the variable $\hat{C}$ at the end of the algorithm. By Lemma 5.4, $\hat{C}' \cap C$ is a superset of a solution to the OPT-SET problem. By Lemma 5.3, $\hat{C}' \cap C$ is a subset of a solution to the OPT-SET problem. Thus, value returned by the shadow algorithm, $\hat{C}' \cap C$, is a solution to the OPT-SET problem.

□

5.2 Polynomial Time Equivalence of NEL, OPT-SET, and Separation

In this section we show a polynomial time equivalence between the NEL problem, the OPT-SET problem and the separation problem over Equations (1), (2), and (3).

Lemma 5.6. *The solution to the focus point price problem on input (C, τ, Bcost) is 0 iff the solution to the NEL problem on input (C, τ, Bcost) is FALSE.*

Proof. First, we prove that if the solution to the NEL problem is FALSE, then the solution to the FPP problem is zero. Consider all of the inequality pairs $\pi_a \leq \text{MinEval}(C - \{a\}, \tau, \text{Bcost}) - \text{MinEval}(C, \tau, \text{Bcost})$ and $\pi_a \geq 0$. Since the solution to the NEL problem is FALSE, for each $a \in C$ there is a $\mathcal{F}_a \subseteq C$ such that

Eval($C, \mathcal{F}_a$, Bcost) = MinEval(C, τ, Bcost) and $a \notin \mathcal{F}_a$. Thus the first inequality in the pair reduces to $\pi_a \leq 0$, and the pair of inequalities imply $\pi_a = 0$. This is true for all $a \in C$. Thus, optimal value of the FPP linear program is zero.

Second, we prove that if the solution to the NEL problem is TRUE, then the solution to the FPP problem is greater than zero. If the solution to NEL is TRUE, then there is some $a \in C$ such that if Eval($\mathcal{F}, \tau$, Bcost) = MinEval(C, τ, Bcost) then $a \in \mathcal{F}$. In other words, a is in all solutions to the OPT-SET problem on input (C, τ, Bcost).

Thus, for all $\mathcal{A} \subseteq C$ with $a \in \mathcal{A}$, we have MinEval($C - \mathcal{A}, \tau$, Bcost) $-$ MinEval(C, τ, Bcost) > 0. Let $\lambda = \min_{\substack{\mathcal{A} \subseteq C \\ a \in \mathcal{A}}}$[MinEval($C - \mathcal{A}, \tau$, Bcost) $-$ MinEval(C, τ, Bcost)]. Consider the vector π with $\pi_b = 0$ for all $b \in C - \{a\}$ and $\pi_a = \lambda$ and $\pi_\mu = M -$ MinEval(C, τ, Bcost) $- \lambda$. Since $\lambda \leq M -$ MinEval(C, τ, Bcost), this vector is feasible in the focus point price linear program and achieves a objective function value greater than zero.

□

Lemma 5.7. *If it is possible to approximate the solution to the FPP problem on input* (C, τ, Bcost) *within any multiplicative factor, then the NEL problem on input* (C, τ, Bcost) *is solvable in polynomial time.*

Proof. Follows from Lemma 5.6.

□

A set of (C, τ, Bcost) instances is *proper* if the following conditions hold:

- Given that (C, τ, Bcost) is in the set, then so is ($C, \hat{\tau}$, Bcost), where $\hat{\tau}(a) = \tau(a) + \pi_a$ for a vector $\pi \in \mathfrak{R}_+^{|C|}$.
- Given that (C, τ, Bcost) is in the set, then so is ($\mathcal{A}, \hat{\tau}$, Bcost*), where ($\hat{C}, \hat{\tau}$, Bcost*) is the full shadow extension of (C, τ, Bcost) and $\mathcal{A}$ is a subset of $\hat{C}$.

The definition of proper has a natural interpretation when applied to the transformations of combinatorial minimization problems to buyer-supplier games. For example, for the shortest path problem, the first condition implies that the set of instances is closed with respect to lengthening the edges of the graph. On the other hand, the second condition implies that the set of instances is closed with respect to adding parallel edges or removing a subset of the edges.

Theorem 5.8. *On a proper set of instances, the separation problem over Equations (1), (2), and (3), the NEL problem and the OPT-SET problem are polynomial time equivalent.*

Proof. If we can solve the NEL problem on a proper set of instances in polynomial time, then, by Lemma 5.5, we can solve the OPT-SET problem in polynomial time.

If we can solve the OPT-SET problem on a proper set of instances in polynomial time, then, by Theorem 4.1, we can solve the separation problem over Equations (1), (2), and (3) in polynomial time.

If we can solve the separation problem over Equations (1), (2), and (3) on a proper set of instances in polynomial time, then, by the polynomial time equivalence of separation and optimization, we can optimize linear objective functions over Equations (1), (2), and (3) in polynomial time. If we can optimize linear objective functions in polynomial time, by Lemma 5.7 we can solve the NEL problem in polynomial time.

□

Lemma 5.9. *On a proper set of instances, if it is not possible to solve the OPT-SET problem in polynomial time, it is not possible to approximate the solution to the FPP problem to within any multiplicative factor in polynomial time.*

Proof. Follows from Theorem 5.8 and Lemma 5.7.

□

6 A Simplification of the FPP Problem

In this section, we give a simplified linear program that may be used to solve the FPP problem.

For this section, fix a buyer-supplier game defined by (C, τ, Bcost). Let the buyer-supplier game be derived from the combinatorial minimization problem MinProb as described in Section 1. We omit the parameters τ and Bcost from MinEval since they are fixed by the game.

Lemma 6.1. *For all $\mathcal{A} \subseteq C$, we have* $\text{MinEval}(\mathcal{A}) = \min(M, \text{MinProb}(\mathcal{A}))$.

Proof. By the definition of MinEval and Eval, we have

$$\begin{aligned}\text{MinEval}(\mathcal{A}) &= \min_{\mathcal{B}\subseteq\mathcal{A}}[\text{Eval}(\mathcal{B})]\\ &= \min_{\mathcal{B}\subseteq\mathcal{A}}[\text{Bcost}(\mathcal{B}) + \sum_{a\in\mathcal{B}} \tau(a)]\end{aligned}$$

We explicitly instantiate the case when $\mathcal{B} = \emptyset$. Since $\text{Bcost}(\emptyset) = M$, we have

$$\text{MinEval}(\mathcal{A}) = \min(M, \min_{\mathcal{B}\subseteq\mathcal{A}} [\text{Bcost}(\mathcal{B}) + \sum_{a\in\mathcal{B}} \tau(a)])$$

Since $P(\mathcal{B}) = 0$ implies $\text{Bcost}(\mathcal{B}) = M$ and since $\tau(a) \geq 0$ for all $a \in C$, we have

$$\text{MinEval}(\mathcal{A}) = \min(M, \min_{\substack{\mathcal{B}\subseteq\mathcal{A}\\ P(\mathcal{B})=1}} [\text{Bcost}(\mathcal{B}) + \sum_{a\in\mathcal{B}} \tau(a)])$$

By the definition of Bcost, we have

$$\begin{aligned}\text{MinEval}(\mathcal{A}) &= \min(M, \min_{\substack{\mathcal{B}\subseteq\mathcal{A}\\ P(\mathcal{B})=1}} [\text{Bcost}^*(\mathcal{B}) + \sum_{a\in\mathcal{B}} \tau(a)])\\ &= \min(M, \text{MinProb}(\mathcal{A}))\end{aligned}$$

□

Consider the linear program from the FPP problem for the given game. In particular, consider the variable π_μ. The variable can be viewed as a slack variable for the constraint arising from Equation (3). In specific, we can write $0 \leq \pi_\mu = M - \text{MinEval}(C) - \sum_{b\in C} \pi_b$, where the inequality comes from the constraint $\pi_\mu \geq 0$ and the equality comes from the constraint arising from Equation (3). Thus, the following linear program is equivalent to the linear program from the FPP problem

$$\begin{aligned}\max \sum_{b\in C} \pi_b & \\ \text{s.t.} \sum_{b\in\mathcal{A}} \pi_b \leq \text{MinEval}(C - \mathcal{A}) - \text{MinEval}(C) & \qquad \text{for all } \mathcal{A} \subseteq C\\ \sum_{b\in C} \pi_b \leq M - \text{MinEval}(C) & \\ \pi_b \geq 0 & \qquad \text{for all } b \in C.\end{aligned}$$

Which, in turn, by Lemma 6.1 is equivalent to

$$\begin{aligned}
\max \sum_{b\in C} \pi_b & \\
\text{s.t.} \sum_{b\in\mathcal{A}} \pi_b &\leq \min(M, \text{MinProb}(C-\mathcal{A})) - \min(M, \text{MinProb}(C)) && \text{for all } \mathcal{A}\subseteq C \\
\sum_{b\in C} \pi_b &\leq M - \min(M, \text{MinProb}(C)) \\
\pi_b &\geq 0 && \text{for all } b\in C.
\end{aligned}$$

Call the above linear program LP1 and let its optimal value be O_1.

Consider the following linear program

$$\begin{aligned}
\max \sum_{b\in C} \pi_b & \\
\text{s.t.} \sum_{b\in\mathcal{A}} \pi_b &\leq \text{MinProb}(C-\mathcal{A}) - \text{MinProb}(C) && \text{for all } \mathcal{A}\subseteq C \\
\pi_b &\geq 0 && \text{for all } b\in C.
\end{aligned}$$

Call the above linear program LP2 and let its optimal value be O_2.

Lemma 6.2. *If* MinProb$(C) \geq M$*, then* $O_1 = 0$.

Proof. Consider the inequality $\sum_{b\in C} \pi_b \leq \min(M, \text{MinProb}(C-C)) - \min(M, \text{MinProb}(C))$, which must be satisfied by all vectors feasible in LP1. The right hand side of this inequality is equal to 0, since $M \leq \text{MinProb}(C)$ and $\text{MinProb}(\emptyset) = \infty$. Thus, we know that $O_1 \leq 0$. We also have $O_1 \geq 0$ since the all zero vector is feasible for LP1.

□

Lemma 6.3. *If* MinProb$(C) < M$ *and* $O_2 \leq M - \text{MinProb}(C)$*, then* $O_1 = O_2$.

Proof. Since MinProb$(C) < M$, for any $\mathcal{A}\subseteq C$ we have

$$\begin{aligned}
\min(M, \text{MinProb}(C-\mathcal{A})) - \min(M, \text{MinProb}(C)) &= \min(M, \text{MinProb}(C-\mathcal{A})) - \text{MinProb}(C) \\
&\leq \text{MinProb}(C-\mathcal{A}) - \text{MinProb}(C)
\end{aligned}$$

Thus, if a vector π is feasible in LP1, then π is also feasible in LP2. Thus, we have $O_1 \leq O_2$.

Let π^* be an optimal vector for LP2. Let $\mathcal{A}$ be any subset of C. Since π^* is feasible in LP2, we have

$$\sum_{b\in\mathcal{A}} \pi^*_b \leq \text{MinProb}(C-\mathcal{A}) - \text{MinProb}(C).$$

Since $O_2 \leq M - \text{MinProb}(C)$ we also have

$$\sum_{b\in\mathcal{A}} \pi^*_b \leq \sum_{b\in C} \pi^*_b = O_2 \leq M - \text{MinProb}(C).$$

Thus, we have

$$\sum_{b\in\mathcal{A}} \pi^*_b \leq \min(M, \text{MinProb}(C-\mathcal{A})) - \text{MinProb}(C) = \min(M, \text{MinProb}(C-\mathcal{A})) - \min(M, \text{MinProb}(C))$$

Thus, π^* is feasible in LP1. Thus, we have $O_1 \geq O_2$.

Thus, $O_1 = O_2$.

□

Lemma 6.4. *If* MinProb(C) $< M$ *and* $O_2 > M -$ MinProb(C)*, then* $O_1 = M -$ MinProb(C).

Proof. Let π^* be an optimal vector for LP2. Thus, we have

$$\sum_{a\in C} \pi^*_a = O_2 > M - \text{MinProb}(C)$$

Let $v^* \in \Re^{|C|}$ be any vector such that

$$\begin{aligned} &0 \le v^*_a \le \pi^*_a && \text{for all } a \in C \\ &\sum_{a\in C} v^*_a = M - \text{MinProb}(C) \end{aligned}$$

We know that such a v^* exists since we can decrease each of the coordinates of π^* in turn, not decreasing any coordinate past zero, until we have $\sum_{a\in C} v^*_a = M - \text{MinProb}(C)$.

Thus, we have that v^* is feasible in the constraint $\sum_{a\in C} v^* \le M - \min(M, \text{MinProb}(C))$ of LP1.

Let $\mathcal{A}$ be any subset of C. We have

$$\sum_{a\in\mathcal{A}} v^*_a \le \sum_{a\in C} v^*_a = M - \text{MinProb}(C).$$

Since π^* is feasible in LP2, we also have

$$\sum_{a\in\mathcal{A}} v^*_a \le \sum_{a\in\mathcal{A}} \pi^*_a \le \text{MinProb}(C - \mathcal{A}) - \text{MinProb}(C).$$

Thus,

$$\sum_{a\in\mathcal{A}} v^*_a \le \min(M, \text{MinProb}(C - \mathcal{A})) - \text{MinProb}(C) = \min(M, \text{MinProb}(C - \mathcal{A})) - \min(M, \text{MinProb}(C))$$

and v^* is feasible in LP1.

The value of LP1 for vector v^* is $M-$MinProb(C). But, the constraint $\sum_{a\in C} v \le M - \min(M, \text{MinProb}(C))$ of LP1, which must be satisfied by all feasible vectors v, tells us that the value of LP1 can be at most $M -$ MinProb(C). Thus, we have $O_1 = M -$ MinProb(C).

□

By Lemmas 6.2, 6.3, and 6.4 finding the solution to LP2 is sufficient to solve the FPP problem for given buyer-supplier game.

7 A Complementary Combinatorial Algorithm

In this section, we present an efficient combinatorial algorithm for solving the FPP problem for the buyer-supplier minimum spanning tree (MST) game. The combinatorial algorithm complements the theoretically efficient algorithm based on the ellipsoid method arising from Theorem 4.1.

Let a graph $G = (\mathcal{V}, \mathcal{E})$ and edge weights $w : \mathcal{E} \to \Re_+$ be given. Let MSTVal : $2^{\mathcal{E}} \to \Re_+$ be a function that takes as input a set of the edges $\mathcal{A} \subseteq \mathcal{E}$ and returns the weight of the minimum spanning tree of the graph induced by the edges of $\mathcal{A}$. If no spanning tree exists, MSTVal returns ∞.

We use the transformation in Section 1 on the minimum spanning tree problem. By Lemma 3.22, any internal cost decomposition is equivalent in terms of core solutions. Thus, we have the definition of the buyer-supplier minimum spanning tree game. It is a buyer-supplier game where $C = \mathcal{E}$, $\tau(a) = w(a)$, and Bcost($\mathcal{A}$) $= M$ if $\mathcal{A}$ does not connect all nodes in $\mathcal{V}$, or 0 otherwise.

Lemma 7.1. *Let $H = (\mathcal{V}_1, \mathcal{E}_1)$ be any graph. Let $T = (\mathcal{V}_1, \mathcal{E}_1')$ be a minimum spanning tree of the graph H. For $e \in \mathcal{E}_1'$ let the cut created in T by the removal of e be $(\mathcal{A}_e, \mathcal{B}_e)$ for some $\mathcal{A}_e \subseteq \mathcal{V}_1$ and $\mathcal{B}_e \subseteq \mathcal{V}_1$.*

- *The edge e is a minimum weight edge spanning the cut $(\mathcal{A}_e, \mathcal{B}_e)$.*
- *The tree T restricted to vertices in $\mathcal{A}_e$ is a minimum spanning tree of the graph induced by the vertices $\mathcal{A}_e$. A symmetric statement is true for $\mathcal{B}_e$*
- *Let $\mathcal{U}$ be the set of edges spanning the cut and let $a = arg\ min_{b \in \mathcal{U}-\{e\}} w(b)$. We have $w(a) - w(e) = \text{MSTVal}(\mathcal{E}_1 - \{e\}) - \text{MSTVal}(\mathcal{E}_1)$.*

Proof. We use a proof by contradiction. Suppose there is some other edge e' which spans the cut such that $w(e') < w(e)$. Then, we could get a new spanning tree T' of the graph H by replacing the edge e in $\mathcal{E}_1'$ with the edge e'. The tree T' has a strictly smaller weight than the tree T since $w(e') < w(e)$, contradicting the fact that T is a minimum spanning tree of H.

Similarly, if T restricted to the vertices in $\mathcal{A}_e$ is not a minimum spanning tree of the graph induced by the vertices in $\mathcal{A}_e$, we can construct a spanning tree of H with weight strictly smaller than T by connecting e, the tree T restricted to $\mathcal{B}_e$, the minimum spanning tree induced by the vertices in $\mathcal{A}$.

By first second result of this lemma, a minimum spanning tree of the graph induced by $\mathcal{E}_1 - \{e\}$ can be obtained by keeping the portions of T which lie wholly in exactly one of $\mathcal{A}_e$ or $\mathcal{B}_e$ and attaching the smallest weight edge which spans the cut. Thus, we get $w(a) - w(e) = \text{MSTVal}(\mathcal{E}_1 - \{e\}) - \text{MSTVal}(\mathcal{E}_1)$.

□

Lemma 7.2. *For $\mathcal{A} \subseteq \mathcal{E}$ and $e \in \mathcal{E} - \mathcal{A}$, if the removal of the edges in $\mathcal{A}$ does not disconnect the graph G, we have $\text{MSTVal}(\mathcal{E} - \mathcal{A} - \{e\}) - \text{MSTVal}(\mathcal{E} - \mathcal{A}) \geq \text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})$.*

Proof. Let $T = (\mathcal{V}, \mathcal{E}')$ be a minimum spanning tree of the graph G.

If $e \notin \mathcal{E}'$, we have $0 = \text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E}) \leq \text{MSTVal}(\mathcal{E} - \mathcal{A} - \{e\}) - \text{MSTVal}(\mathcal{E} - \mathcal{A})$, since $\text{MSTVal}(\mathcal{B}) \geq \text{MSTVal}(\mathcal{U})$ for any $\mathcal{B} \subseteq \mathcal{U}$.

If $e \in \mathcal{E}'$, let the cut created in T by the removal of e be $(\mathcal{K}_e, \mathcal{B}_e)$ for some $\mathcal{K} \subseteq \mathcal{V}$ and $\mathcal{B} \subseteq \mathcal{V}$. Let $\mathcal{U}$ be the set of edges spanning the cut. Let a be arg $\min_{b \in \mathcal{U}-\{e\}} w(b)$. By Lemma 7.1, we have $w(a) - w(e) = \text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})$.

An application of *Lemma* 7.1 on the graph induced by the edges $\mathcal{E} - \mathcal{A}$ yields the result that $w(a') - w(e) = \text{MSTVal}(\mathcal{E} - \mathcal{A} - \{e\}) - \text{MSTVal}(\mathcal{E} - \mathcal{A})$ where $a' = \text{arg} \min_{b \in \mathcal{U}-\mathcal{A}-\{e\}} w(b)$.

By the definitions of a and a' and the fact that $\mathcal{U} - \mathcal{A} - \{\mathcal{E}\} \subseteq \mathcal{U} - \{\mathcal{E}\}$, we have

$$\text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E}) = w(a) - w(e) \leq w(a') - w(e) = \text{MSTVal}(\mathcal{E} - \mathcal{A} - \{e\}) - \text{MSTVal}(\mathcal{E} - \mathcal{A}).$$

□

Lemma 7.3. *Suppose the graph G is connected and thus* $\text{MSTVal}(\mathcal{E})$ *is finite. For $\mathcal{A} \subseteq \mathcal{E}$, we have* $\text{MSTVal}(\mathcal{E} - \mathcal{A}) - \text{MSTVal}(\mathcal{E}) \geq \sum_{e \in \mathcal{A}} [\text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})]$

Proof. We prove the statement by induction on the size of $\mathcal{A}$.

For $\mathcal{A} = \emptyset$, both the left hand side and the right hand side of the statement are equal to zero.

Assume the statement holds for $\mathcal{B}$. We show that it holds for $\mathcal{B} \cup \{e\}$ where $e \in \mathcal{E} - \mathcal{B}$. We must show

$$\begin{aligned} &\text{MSTVal}(\mathcal{E} - \mathcal{B} - \{e\}) - \text{MSTVal}(\mathcal{E}) \\ &\qquad \geq [\text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})] + \sum_{a \in \mathcal{B}} [\text{MSTVal}(\mathcal{E} - \{a\}) - \text{MSTVal}(\mathcal{E})] \end{aligned} \quad (4)$$

If removing the edges $\mathcal{B} \cup \{e\}$ disconnects the graph, the left hand side of the statement is infinite, and thus the statement holds.

If removing $\mathcal{B} \cup \{e\}$ does not disconnect the graph, we have

$$\begin{aligned}
&\text{MSTVal}(\mathcal{E} - \mathcal{B} - \{e\}) - \text{MSTVal}(\mathcal{E}) \\
&\qquad = [\text{MSTVal}(\mathcal{E} - \mathcal{B} - \{e\}) - \text{MSTVal}(\mathcal{E} - \mathcal{B})] + [\text{MSTVal}(\mathcal{E} - \mathcal{B}) - \text{MSTVal}(\mathcal{E})] \\
&\qquad \leq [\text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})] + [\text{MSTVal}(\mathcal{E} - \mathcal{B}) - \text{MSTVal}(\mathcal{E})] \\
&\qquad \leq [\text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})] + \sum_{a \in \mathcal{B}} [\text{MSTVal}(\mathcal{E} - \{a\}) - \text{MSTVal}(\mathcal{E})]
\end{aligned}$$

where the first inequality comes from Lemma 7.2 and the second inequality comes from the induction hypothesis.

□

Let LP2 be as in Section 6 with MinProb equal to MSTVal.

Lemma 7.4. *The inequalities corresponding to singleton sets $\{e\}$ for $e \in \mathcal{E}$ form an optimal basis for* LP2. *In particular, setting π_e =* MSTVal$(\mathcal{E} - \{e\})$ − MSTVal$(\mathcal{E})$ *for all $e \in \mathcal{E}$ gives an optimal vector for* LP2.

Proof. For $\mathcal{A} \subseteq \mathcal{E}$, by Lemma 7.3, we have

$$\sum_{e \in \mathcal{A}} \pi_e = \sum_{e \in \mathcal{A}} [\text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})] \leq \text{MSTVal}(\mathcal{E} - \mathcal{A}) - \text{MSTVal}(\mathcal{E}).$$

Thus, π is feasible in LP2. Since each inequality of the form $\pi_e \leq \text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})$ is tight, it is not possible to increase any coordinate of π. Thus π is an optimal vector and the singleton inequalities form an optimal basis.

□

We give a modified Kruskal Algorithm which can be used to compute the optimal value of LP2.

Run Kruskal Algorithm with the following modifications.

- Throughout the algorithm's execution we will keep an auxiliary set of edges, $\mathcal{A}$, which is initially empty.
- When edge e is added to the minimum spanning forest, also add e to the set $\mathcal{A}$.
- Suppose edge e is rejected from addition to the minimum spanning Forest because it creates a cycle. Let the cycle created be $H = (\mathcal{V}', \mathcal{E}')$. For each edge $a \in \mathcal{E}' - \{e\}$, if $a \in \mathcal{A}$, label a with $w(e) - w(a)$ and remove a from $\mathcal{A}$.

Lemma 7.5. *Let G be connected and $T = (\mathcal{V}_1, \mathcal{E}_1)$ be the minimum spanning tree computed by the modified Kruskal Algorithm when it is run on $G = (\mathcal{V}, \mathcal{E})$. If $e \in \mathcal{E}_1$ has been labeled, the label is equal to* MSTVal$(\mathcal{E} - \{e\})$ − MSTVal$(\mathcal{E})$. *Otherwise,* MSTVal$(\mathcal{E} - \{e\})$ − MSTVal$(\mathcal{E}) = \infty$.

Proof. Let $\mathcal{U}$ be the set of edges spanning the cut created in T by the removal of e. Also, let a' be any $\arg\min_{b \in \mathcal{U} - \{e\}} w(b)$. By Lemma 7.1 we have $w(a') - w(e) = \text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E})$.

Let e be labeled by the modified Kruskal Algorithm when the edge a creates a cycle. For the first part of the lemma, all we must show is $a = a'$.

Let $\mathcal{K}$ be the set of edges which create a cycle involving e during the algorithm. If $b \in \mathcal{E} - \mathcal{U}$, then $\mathcal{B}$ does not span the cut created in T by the removal of e. Thus, both vertices of b lie on the same side of the cut. Thus, b cannot create a cycle involving e, since all edges in the cycle except b must be in the tree T. Also, the edge e cannot form a cycle with itself. Thus, we have $\mathcal{K} \subseteq \mathcal{U} - \{e\}$.

Consider any $b \in \mathcal{U} - \{e\}$. The edge b must be rejected by the modified Kruskal algorithm, otherwise the edge would be in T and wouldn't span the cut created in T by the removal of e. Thus, b must create a cycle during the algorithm. Since all edges except b which make up the cycle must be in T and b spans the cut created by e, the cycle created by b includes e. Thus, we have $\mathcal{U} - \{e\} \subseteq \mathcal{K}$, and $\mathcal{U} - \{e\} = \mathcal{K}$.

Since the modified Kruskal Algorithm process edges of G in ascending order of weight and e is labeled by the element from $\mathcal{K}$ which is processed first, we know $a = \arg\min_{b \in \mathcal{K}} w(b)$. Since $\mathcal{K} = \mathcal{U}$, we also have $a = a'$ and we have shown the first part of the lemma.

If e is not labeled, then $\mathcal{K}$ must be empty. Since $\mathcal{K} = \mathcal{U}$, the set $\mathcal{U}$ must also be empty. And thus, the removal of e must disconnect the graph G. Thus we have, $\text{MSTVal}(\mathcal{E} - \{e\}) - \text{MSTVal}(\mathcal{E}) = \infty$.

□

By Lemmas 7.4 and 7.5, we can find the optimal value to LP2 using the modified Kruskal algorithm. By Lemmas 6.2, 6.3, and 6.4, the minimum spanning tree weight of G and the optimal value of LP2 give us the optimal value of LP1 which is the focus point price of the buyer-supplier minimum spanning tree game on graph G.

8 Applications

In this section we present a sampling of the natural games which fit the buyer-supplier game paradigm. We use these games to illustrate the power of the results in this paper. We place each game into one of two categories.

Category A games are those for which on input $\hat{\tau} : C \to \Re_+$ it is possible to compute both $\text{Eval}(\mathcal{A}, \hat{\tau})$ for any $\mathcal{A} \subseteq C$ and an $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ in polynomial time. For such games, by Theorems 3.12 and 4.1, one can optimize any linear function over the core vectors of the game. For example, for any given player it is possible to determine the difference between the best and worst possible core outcome for the player.

Category B games those for which on input $\hat{\tau} : C \to \Re_+$ it is impossible, in polynomial time, to compute $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$. In other words, the corresponding OPT-SET problem is not solvable in polynomial time on proper sets of instances. By Theorem 5.8, in general, given some payoff vector π it is not possible to determine in polynomial time whether π is in the core. Moreover, by Lemma 5.9 it is impossible to find the difference between the best and worst possible core outcome for the buyer in polynomial time.

Facility Location In the buyer-supplier facility location game a company has customers on some of the nodes of a graph. The company must build facilities on the graph to supply its customers with a product. The company pays to both build the facilities, and run trucks from each customer to the customer's assigned facility. Each possible facility location is owned by a unique supplier. The supplier has some internal cost associated with hosting a facility. Specifically, given a graph $G = (\mathcal{V}, \mathcal{E})$, a set of customers $\mathcal{B} \subseteq \mathcal{V}$, a set of possible facility locations $\mathcal{U} \subseteq \mathcal{V}$, an opening cost function $f : \mathcal{U} \to \Re_+$, and a service function $h : 2^{\mathcal{U}} \to \Re_+$ which captures the cost of providing the product to the customers given facilities on $\mathcal{U}$, let

$$C = \mathcal{U}$$
$$\tau(a) = f(a)$$
$$\text{Bcost}(\mathcal{A}) = \begin{cases} M & \text{if } \mathcal{A} = \emptyset \\ h(\mathcal{A}) & \text{otherwise.} \end{cases}$$

For this game, on input $\hat{\tau}$, computing $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ is the facility location problem. Unless NP = P, computing the optimal facility locations is not possible in polynomial time since the facility location problem is NP-complete. Thus, unless NP = P, the buyer-supplier facility location game is in category B.

Minimum Cut In the buyer-supplier minimum cut game a company wishes to control all the paths between two nodes of a graph. Each edge of the graph is initially owned by a unique supplier. The company wishes to purchase the edges of a cut which separates the two specified nodes. Each supplier has a cost associated with the company's usage of the supplier's edge. Specifically, given a graph $G = (\mathcal{V}, \mathcal{E})$, edge weights $w : \mathcal{E} \to \Re_+$, and two nodes $s \in \mathcal{V}$ and $t \in V$, let

$$C = \mathcal{E}$$
$$\tau(a) = w(a)$$
$$\text{Bcost}(\mathcal{A}) = \begin{cases} M & \text{if } \mathcal{A} \text{ is not a cut separating } s \text{ and } t \\ 0 & \text{otherwise.} \end{cases}$$

For this game, on input $\hat{\tau}$, computing $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ is simply the mininum cut problem. Computing $\text{Eval}(\mathcal{A}, \hat{\tau})$ for $\mathcal{A} \subseteq C$ can be done using depth first search and a simple summation. Thus, the buyer-supplier minimum cut game is in category A.

Mininum Set Cover In the buyer-supplier minimum set cover game a company has a set of services it wishes to outsource to a set of subcontractors. Each subcontractor provides only some subset of the company's required services. Each subcontractor as a cost associated with providing services to the company. Specifically, given a set of services $\mathcal{B}$, a set of subcontractors $\mathcal{U}$, the services provided by each subcontractor $h : \mathcal{U} \to 2^{\mathcal{B}}$, and a cost function for each subcontractor $f : \mathcal{U} \to \Re_+$, let

$$C = \mathcal{U}$$
$$\tau(a) = f(a)$$
$$\text{Bcost}(\mathcal{A}) = \begin{cases} M & \text{if } \cup_{a \in \mathcal{A}} h(a) \neq \mathcal{B} \\ 0 & \text{otherwise.} \end{cases}$$

For this game, even if we restrict $\hat{\tau}$ to be identically zero, computing $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ is the minimum set cover problem. Unless NP = P, computing the minimum set cover is not possible in polynomial time [5, p. 222]. Thus, unless NP = P, the buyer-supplier minimum set cover game is in category B.

Minimum Spanning Tree We have already introduced this game. The buyer supplier minimum spanning tree game is in category A.

Shortest Path In the buyer-supplier shortest path game a company wishes to transport goods between two nodes on a graph. The company wishes to purchase a path connecting the two nodes for the transportation use. Each edge of the graph is owned by a unique supplier. Each supplier has a cost associated with the company's traffic passing through the supplier's edge. Specifically, given a graph $G = (\mathcal{V}, \mathcal{E})$, edge weights $w : \mathcal{E} \to \Re_+$, and two nodes $s \in \mathcal{V}$ and $t \in V$, let

$$C = \mathcal{E}$$
$$\tau(a) = w(a)$$
$$\text{Bcost}(\mathcal{A}) = \begin{cases} M & \text{if } \mathcal{A} \text{ does not connect } s \text{ and } t \\ 0 & \text{otherwise.} \end{cases}$$

For this game, on input $\hat{\tau}$, computing $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ is simply the shortest path problem. Computing $\text{Eval}(\mathcal{A}, \hat{\tau})$ for $\mathcal{A} \subseteq C$ can be done using depth first search and a simple summation. Thus, the buyer-supplier shortest path game is in category A.

Single Commodity Flow In the buyer-supplier single commodity flow game a company wishes to move a commodity from from a set of source vertices of a graph to a set of destination vertices of the graph. Each edge is owned by a unique supplier, and each supplier has a cost associated with the company's usage of the supplier's edge. Moreover, each edge has an associated capacity constraint, limiting the company from over-burdening the edge. Time is a factor in the company's movement of the commodity. The latency of using a particular edge increases linearly with the amount of commodity flowing through the edge. The company wishes to buy edges to use in moving the commodity, while at the same time minimizing the latency of delivery. Specifically, given a graph $G = (\mathcal{V}, \mathcal{E})$, edge weights representing each supplier's costs $w : \mathcal{E} \rightarrow \Re_+$, a set of source vertices $\mathcal{B} \subseteq \mathcal{E}$, a set of destination vertices $\mathcal{U} \subseteq \mathcal{E}$, fixed amount of commodity $\lambda \in \Re_+$ and a function $h : 2^{\mathcal{E}} \rightarrow \Re_+$ which captures the latency of routing λ flow of the commodity from $\mathcal{B}$ to $\mathcal{U}$ subject to the capacity constraints of the chosen edges, let

$$\begin{aligned} C &= \mathcal{E} \\ \tau(a) &= w(a) \\ \text{Bcost}(\mathcal{A}) &= \begin{cases} M & \text{if } \mathcal{A} = \emptyset \\ h(\mathcal{A}) & \text{otherwise.} \end{cases} \end{aligned}$$

For this game, on input $\hat{\tau}$, computing $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ is the fixed charge network flow problem.. Unless NP = P, solving the fixed charge network flow problem is not possible in polynomial time [9]. Thus, unless NP = P, the buyer-supplier single commodity flow game is in category B.

Steiner Tree This game is very similar to the buyer-supplier minimum spanning tree game. However, in the buyer-supplier Steiner tree game, the company owns factories on a subset of the graph's nodes. Specifically, given a graph $G = (\mathcal{V}, \mathcal{E})$, edge weights $w : \mathcal{E} \rightarrow \Re_+$, and $\mathcal{B} \subseteq \mathcal{V}$ let

$$\begin{aligned} C &= \mathcal{E} \\ \tau(a) &= w(a) \\ \text{Bcost}(\mathcal{A}) &= \begin{cases} M & \text{if } \mathcal{A} \text{ does not connect all nodes in } \mathcal{B} \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

For this game, on input $\hat{\tau}$, computing $\mathcal{F} \subseteq C$ such that $\text{Eval}(\mathcal{F}, \hat{\tau}) = \text{MinEval}(C, \hat{\tau})$ is minimum cost Steiner tree problem. Unless NP = P, computing the minimum cost Steiner tree is not possible in polynomial time [5, p. 208]. Thus, unless NP = P, the buyer-supplier Steiner tree game is in category B.

References

[1] A. Archer and E. Tardos. Truthful mechanisms for one-parameter agents. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 482–491, Oct. 2001.

[2] R. J. Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1:67–96, 1974.

[3] F. Y. Edgeworth. *Mathematical psychics, an essay on the application of mathematics to the moral sciences*. A. M. Kelley, New York, NY, 1961.

[4] J. Feigenbaum, C. H. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 218–227, May 2000.

[5] M. Garey and D. Johnson. *Computers and Intractability: A guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, NY, 1979.

[6] D. B. Gillies. *Some Theorems on n-Person Games*. PhD thesis, Princeton University, 1953.

[7] M. X. Goemans and M. Skutella. Cooperative facility location games. *Journal of Algorithms*, 50:194–214, 2004.

[8] J. C. Harsanyi. Games with incomplete information played by "Bayesian" players. *Management Science*, 14:159–182,320–334,486–502, 1968.

[9] D. S. Hochbaum and A. Segev. Analysis of a flow problem with fixed charges. *Networks*, 19:291–312, 1989.

[10] J. F. Nash. Equilibrium points in n-person games. In *Proceedings of the National Academy of Sciences*, pages 48–49, 1950.

[11] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley and Sons, New York, NY, 1988.

[12] N. Nisan and A. Ronen. Algorithmic mechanism design. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 129–140, May 1999.

[13] M. Pál and E. Tardos. Group strategy proof mechanisms via primal-dual algorithms. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 584–593, Oct. 2003.

[14] C. H. Papadimitriou. Algorithms, games, and the internet. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 749–753, July 2001.

[15] C. H. Papadimitriou and T. Roughgarden. Computing equilibria in multi-player games. In *Proceedings of the 16th Annual ACM-SIAM symposium on Discrete algorithms*, pages 82–91, January 2005.

[16] K. Roberts. The characterization of implementable choice rules. In *Aggregation and Revelation of Preferences*, pages 321–348. North-Holland, Amsterdam, 1979.

[17] T. Roughgarden. *Selfish Routing and the Price of Anarchy*. MIT Press, Cambridge, MA, 2005.

[18] T. Roughgarden and E. Tardos. How bad is selfish routing? *J. ACM*, 49:236–259, 2002.

[19] T. Roughgarden and E. Tardos. Bounding the inefficiency of equilibria in nonatomic congestion games. *Games and Economic Behaviour*, 47:389–403, 2004.

[20] L. S. Shapley. Notes on the n-person game III: Some variants of the von Neumann-Morgenstern definition of solution. Research memorandum, RAND Corporation, Santa Monica, CA, 1952.

[21] M. Shubik. *Game Theory In The Social Sciences*. MIT Press, Cambridge, Massachussetts, 1984.

[22] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ, 1953.

[23] R. Weber. Multiple-object auctions. In *Auctions, bidding, and contracting: Uses and theory*, pages 165–191. New York University Press, New York, NY, 1983.