

Nash Bargaining via Flexible Budget Markets

*Vijay V. Vazirani**

Abstract

We initiate a study of Nash bargaining games [Nas50] via combinatorial, polynomial time algorithms and we carry this program over to solving nonsymmetric bargaining games of Kalai [Kal77] as well. For each game studied, this involves obtaining a combinatorial algorithm for solving the corresponding convex program. We reduce the game to computing an equilibrium in a new market model called *flexible budget market*, and we find an equilibrium using the primal-dual paradigm. Our main result pertains to a natural Nash bargaining game called **ADNB** which is derived from the linear case of Arrow-Debreu markets.

We discuss the importance of seeking combinatorial algorithms for Nash bargaining games, even though they admit convex programs. We give an application of our algorithm for **ADNB** to a “fair” throughput allocation problem on a wireless channel. Additionally, the structural insights gained from the combinatorial nature of our algorithms has led to novel insights into game-theoretic properties of the solution concepts of Nash and nonsymmetric bargaining games, see [CGV⁺09].

*College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280, E-mail: vazirani@cc.gatech.edu.

1 Introduction

In his seminal 1950 paper, John Nash defined the bargaining game [Nas50]. The ensuing theory of bargaining (e.g., see [Kal85, TL89, OR94]) lies today at the heart of game theory. In this paper, we initiate a study of Nash bargaining games via combinatorial¹ polynomial time algorithms. We also carry this program over to solving nonsymmetric bargaining games of Kalai [Kal77].

Game theory develops solution concepts for negotiating in situations of conflict of interest and bargaining is perhaps the oldest such situation known to mankind. Presently, at least in the West, the scope of bargaining has been narrowed down to very high end items, such as the sale of houses; however, one can see its return via portals² on the Internet. The latter is made possible by the massive computational power available on the Internet which allows companies to negotiate directly with individual customers. Such applications point to the importance of studying bargaining in the “traditional” style of algorithmic game theory [NRTV07], i.e., taking into consideration both strategic issues and algorithmic efficiency.

1.1 Flexible budget markets and the game **ADNB**

The solution to a Nash or a nonsymmetric bargaining game is obtained by maximizing a concave function over a convex set, i.e., it is the solution to a convex program. Hence a combinatorial algorithm is possible only if the convex program is *rational*, i.e., always has a rational solution if all parameters are rational. The quintessential rational nonlinear program, which is also responsible for starting this line of work, is the Eisenberg-Gale program [EG59]. This program captures, as its optimal solution, equilibrium for the linear case of a classic market model given by Irwing Fisher [BS00]; a combinatorial algorithm for it was given in [DPSV08].

In this paper, we define the class LNB of *linear Nash and nonsymmetric bargaining games* – games in which the convex set is a polytope and hence the constraints of the corresponding convex program are linear. We give combinatorial, polynomial time algorithms for a number of games in this class. All of our algorithms are obtained by first reducing the bargaining game to a new market model, which we call a *flexible budget market*, and finding an equilibrium in such a market; see Section 4 for the latter notion.

Our main result is an algorithm for a natural Nash bargaining game derived from the linear case of the Arrow-Debreu market model (see Section 4). We call this game **ADNB**. Extending this algorithm to the nonsymmetric version of this game is not straightforward and is left as an open problem. For the remaining Nash bargaining games studied in this paper, we give extensions to their nonsymmetric versions as well. Our polynomial time algorithm, Algorithm 18, for **ADNB** builds on [DPSV08]. In Section 10 we state the new difficulties that arise beyond [DPSV08] and high level ideas on how we overcome them.

The notion of balanced flows used in the DPSV algorithm and the potential function used by

¹For a discussion of this notion, see Sections 1.3 and 17.1.

²E.g., priceline.com now allows customers to name their price for airline tickets, hotel reservations and car rentals. Even more importantly, iOffer.com operates like eBay, except that it uses bargaining instead of auctions as its basic mechanism for arriving at prices at which sales happen.

them for establishing a bound on the running time uses l_2 norm, which makes the arguments difficult. [DPSV08] observe that l_2 norm can be dispensed with for stating the algorithm and ask, “Can a polynomial running time be established for the algorithm using the alternative definition, thereby dispensing with l_2 norm altogether? At present we see no way of doing this” We answer this question in the negative. In the full paper we will provide an infinite family of examples in which the l_1 norm-based potential function makes inverse exponentially small progress in one phase of the DPSV algorithm. An upshot of this is that the l_2 norm-based argument for our algorithm also appears to be essential.

Our algorithm for **ADNB** is highly involved; however, we would like to argue that it is not needlessly complicated. Operating the primal-dual paradigm in the enhanced setting of convex programs and KKT conditions, instead of the usual setting of LP-duality theory, leads to some inherent difficulties; see Section 10. The reader can see that the algorithm exploits a surprisingly rich and clean structure which is, in some ways, reminiscent of the majestic structure of matching. In our experience, such structure does not occur in isolation and in Section 17 we argue that what we see so far is the tip of an iceberg, and we give some ways of making progress to get at the complete picture.

1.2 Subclasses of LNB

In the second part of this paper, building on ideas from [JV08], we define two subclasses of LNB, UNB (and SNB), for *uniform (submodular) utility Nash and nonsymmetric bargaining games*. These classes figure prominently in [CGV⁺09], which studies several fundamental game-theoretic properties of Nash bargaining solutions. We show that all games in SNB are rational and solvable by combinatorial polynomial time algorithms.

In game theory, 2-player games occupy a special place – not only because numerous applications involve 2 players but also because they often have remarkable properties that are not possessed by extensions to more players. In [Vaz09] we explore this theme for Nash and nonsymmetric bargaining games; we show that any game in LNB2, the subclass of 2-person LNB games, is rational and is solvable in polynomial time.

1.3 Why seek combinatorial algorithms?

The synergy between combinatorial and “continuous” algorithms has had a far reaching impact on the fields of computer science, operations research and mathematical programming; moreover, each has its own forte and advantages over the other. The major advantage of combinatorial algorithms is that they have led to deep insights into the combinatorial structure of problems studied. In turn, these insights have yielded algorithms for generalizations and variants of the basic problem. Combinatorial algorithms also tend to be the most efficient algorithms known for many fundamental problems and tend not to have issue of instability due to arithmetic overflow.

On the other hand, general purpose “continuous” methods, such as ellipsoid method or interior point method, provide a “black box” approach to algorithmically solving problems in that they provide a solution to the given *instance*, but no insight into the problem as a whole.

Furthermore, as long as the problem is captured via an LP or a convex program, the same big hammer applies – the instance may come from as easy a problem as shortest path or as difficult a problem as market equilibrium.

For problems whose solutions are captured by linear programs, such as matching, max-flow, shortest path, minimum spanning tree, etc., the community has embraced combinatorial algorithms as the method of choice because of all the advantages listed above, e.g., see the recent treatise by Schrijver [Sch03]. Our attempts at building an analogous theory for problems whose solutions are captured via convex programs are motivated by the same advantages; we examples of the latter below.

We start with market equilibrium algorithms before moving on to Nash bargaining algorithms. The combinatorial algorithm of [DPSV08] led to an understanding of continuity properties of equilibria for several market models [MV07, VW09]. It also gave the linear program described in [VY10] that captures equilibria for the case of piecewise-linear, concave utility functions, given a suitable guess. This led to a proof of rationality and moreover the LP was used crucially in showing membership of this case in PPA. The algorithm of [DPSV08] was extended in [Vaz06] to tackle a generalization of Fisher’s linear case which has applications in Google’s Adwords market. Interestingly enough, a convex program is not known for this generalization, hence the continuous approaches were not applicable – this and the possibility of heuristic adaptations of combinatorial algorithms illustrates a key strength. [DPSV08] was also extended in [GV09] to study equilibria in a market model that captures price discrimination. Combinatorial insights gained in the current paper have led to novel insights into game-theoretic properties of the solution concepts of Nash and nonsymmetric bargaining games, see [CGV⁺09].

In terms of applications, Nisan [Nis09] mentions that his algorithm for Google’s auction of TV ads was inspired by combinatorial market equilibrium algorithms. We believe that Nash bargaining, especially via combinatorial algorithms, will find applications in many “fair” allocation settings, e.g., [AQS05] consider the following throughput allocation problem on a wireless channel. There are n users $1, 2, \dots, n$ and the wireless router can be in any of m different states $1, 2, \dots, m$ whose probabilities, $\pi(j)$ are known. Each user i derives utility at rate u_{ij} if it is connected to the router while the router is in state j ; the u_{ij} ’s are known. No matter what state the router is in, only one user can be connected to it. If user i is given connection for $x_{ij} \leq \pi(j)$ of the time the router is in state j , for $1 \leq j \leq m$, then the total utility derived by i is $v_i = \sum_{j=1}^m u_{ij}x_{ij}$. Clearly, we must ensure that $\sum_{i=1}^n x_{ij} \leq \pi(j)$. The question is to find a “fair” way of dividing the $\pi(j)$ ’s among the users.

Following the work of Kelly [Kel97], [AQS05] consider a proportional fair scheme which essentially reduces to solving the Eisenberg-Gale convex program; observe that the above setting can be viewed as a linear Fisher market with n users and m divisible goods. [AQS05] give a gradient descent method for solving this program and actually implement it in their router at Lucent. However, they report that their algorithm suffers from instability due to rounding errors [And09]. In addition, they were not satisfied with this solution because it may allocate unacceptably low utility to certain users. To rectify this, they assume they are given numbers c_i specifying a lower bound on the utility that user i should get. Their extended solution is to simply add the constraints $v_i \geq c_i$ to the convex program.

However, this will distribute the surplus utility among the users in an arbitrary manner. A

better solution is to view this problem as an instance of **ADNB**, with c_i 's providing the disagreement utilities. Furthermore, implementing our combinatorial algorithm, or a heuristic simplification of it, would be a way of avoiding the instability issues.

2 The Nash Bargaining Game

An n -person Nash bargaining game consists of a pair $(\mathcal{N}, \mathbf{c})$, where $\mathcal{N} \subseteq \mathbf{R}_+^n$ is a compact, convex set and $\mathbf{c} \in \mathcal{N}$. Set \mathcal{N} is the *feasible set* and its elements give utilities that the n players can simultaneously accrue. Point \mathbf{c} is the *disagreement point* – it gives the utilities that the n players obtain if they decide not to cooperate. The set of n agents will be denoted by B and the agents will be numbered $1, 2, \dots, n$. Game $(\mathcal{N}, \mathbf{c})$ is said to be *feasible* if there is a point $\mathbf{v} \in \mathcal{N}$ such that $\forall i \in B, v_i > c_i$, and *infeasible* otherwise.

The solution to a feasible game is the point $\mathbf{v} \in \mathcal{N}$ that satisfies the following four axioms:

1. **Pareto optimality:** No point in \mathcal{N} can weakly dominate \mathbf{v} .
2. **Invariance under affine transformations of utilities:** If the utilities of any player are redefined by multiplying by a scalar and adding a constant, then the solution to the transformed game is obtained by applying these operations to the particular coordinate of \mathbf{v} .
3. **Symmetry:** If the players are renumbered, then it suffices to renumber the coordinates of \mathbf{v} accordingly.
4. **Independence of irrelevant alternatives:** If \mathbf{v} is the solution for $(\mathcal{N}, \mathbf{c})$, and $\mathcal{S} \subseteq \mathbf{R}_+^n$ is a compact, convex set satisfying $\mathbf{c} \in \mathcal{S}$ and $\mathbf{v} \in \mathcal{S} \subseteq \mathcal{N}$, then \mathbf{v} is also the solution for $(\mathcal{S}, \mathbf{c})$.

Via an elegant proof, Nash proved:

Theorem 1 Nash [Nas50] *If game $(\mathcal{N}, \mathbf{c})$ is feasible then there is a unique point in \mathcal{N} satisfying the axioms stated above. This is also the unique point that maximizes $\prod_{i \in B} (v_i - c_i)$, over all $\mathbf{v} \in \mathcal{N}$.*

Most papers in game theory assume that the given Nash bargaining game $(\mathcal{N}, \mathbf{c})$ is feasible. However, in this paper, it will be more natural to not make this assumption and to algorithmically determine this fact. Thus, we can have one of 2 outcomes:

Thus Nash's solution to his bargaining game involves maximizing a concave function over a convex domain, and is therefore the optimal solution to the following convex program.

$$\begin{aligned} & \text{maximize} && \sum_{i \in B} \log(v_i - c_i) && (1) \\ & \text{subject to} && \mathbf{v} \in \mathcal{N} \end{aligned}$$

As a consequence, if for a specific game, a separation oracle can be implemented in polynomial time, then using the ellipsoid algorithm one can get as good an approximation to the solution of this convex program as desired in time polynomial in the number of bits of accuracy needed [GLS88].

3 Nonsymmetric Bargaining Games

Kalai [Kal77] generalized Nash’s bargaining game by removing the axiom of symmetry and showed that any solution to the resulting game is the unique point that maximizes $\prod_{i \in B} (v_i - c_i)^{p_i}$, over all $\mathbf{v} \in \mathcal{N}$, for some choice of positive numbers p_i , for $i \in B$, such that $\sum_{i \in B} p_i = 1$.

Thus, any particular nonsymmetric bargaining solution is specified by giving the p_i ’s satisfying the 2 conditions given above. For the purposes of computability, we will restrict to rational p_i ’s. Equivalently, let us define the *n-person nonsymmetric bargaining game* as follows. Assume that $B, \mathcal{N}, \mathbf{c}$ are as defined in Section 2. In addition, we are given the *clout* of each player: a positive integer w_i for each player i . Assuming the game is feasible, the solution to this nonsymmetric bargaining game is the unique point that maximizes $\prod_{i \in B} (v_i - c_i)^{w_i}$, over all $\mathbf{v} \in \mathcal{N}$. Thus an instance of an *n-person nonsymmetric bargaining game* is specified by the triple $(\mathcal{N}, \mathbf{c}, \mathbf{w})$.

The choice of the term “clout of a player” is justified by a key theorem³ of Kalai [Kal77], stating that the solution to the game defined above corresponds precisely to the solution of a k -person game, with $k = \sum_{i \in B} w_i$, which is obtained by taking w_i copies of player i , for $1 \leq i \leq n$. In the solution to the latter game, the utility of all copies of player i will be identical. The correspondence in the forward direction is given by replicating the solution for player i w_i times, for $1 \leq i \leq n$ and that in the reverse direction is given by collapsing the w_i copies into one.

Once again, this nonsymmetric solution can be captured as the optimal solution to a convex program that generalizes program (1):

$$\begin{aligned} & \text{maximize} && \sum_{i \in B} w_i \log(v_i - c_i) && (2) \\ & \text{subject to} && \mathbf{v} \in \mathcal{N} \end{aligned}$$

4 The Class LNB and the Game ADN

The solution to a Nash or a nonsymmetric bargaining game is obtained by maximizing a concave function over a convex set, i.e., it is the solution to a convex program. We first define the subclass NB of these games which can be solved using the ellipsoid algorithm. Let \mathcal{G} be an *n-person Nash or nonsymmetric bargaining game* whose solution is given by the optimal solution to the following convex program, where \mathbf{x} are auxiliary variables, the functions f_i are convex and the functions h_i are affine. (Clearly, \mathcal{G} is a Nash bargaining game if each $w_i = 1$.)

³The theorem in [Kal77] does more, since it also deals with the case that p_i ’s are irrational; however, we will not deal with that generality.

$$\begin{aligned}
& \text{maximize} && \sum_{i \in B} w_i \log(v_i - c_i) && (3) \\
& \text{subject to} && \text{for } i = 1 \dots k : && f_i(\mathbf{v}, \mathbf{x}) \leq 0 \\
& && \text{for } i = 1 \dots l : && h_i(\mathbf{v}, \mathbf{x}) = 0
\end{aligned}$$

The game \mathcal{G} is said to be in the class NB if each of the $k + l$ constraints of program (3) can be checked in polynomial time at any given point (\mathbf{v}, \mathbf{x}) . This gives a separation oracle for the program and therefore, using the ellipsoid algorithm, the Nash or nonsymmetric bargaining solution to the game \mathcal{G} can be obtained to any desired accuracy, assuming the game is feasible. Furthermore, \mathcal{G} is feasible iff the optimal solution to the following convex program is > 0 , which can also be checked in polynomial time.

$$\begin{aligned}
& \text{maximize} && t && (4) \\
& \text{subject to} && \text{for } i = 1 \dots n : && v_i \geq c_i + t \\
& && \text{for } i = 1 \dots k : && f_i(\mathbf{v}, \mathbf{x}) \leq 0 \\
& && \text{for } i = 1 \dots l : && h_i(\mathbf{v}, \mathbf{x}) = 0
\end{aligned}$$

We will say that a Nash or nonsymmetric bargaining game is *linear* if its feasible set \mathcal{N} is a polytope defined via a finite number of linear constraints. The class of these games will be called *linear Nash and nonsymmetric bargaining games*, and abbreviated to LNB.

The game **ADNB**, short for *Arrow-Debreu Nash Bargaining game*, which will be studied extensively, is derived from the linear case of the Arrow-Debreu model. We state the latter first. Let $B = \{1, 2, \dots, n\}$ be a set of agents and $G = \{1, 2, \dots, g\}$ be a set of divisible goods. We will assume w.l.o.g. that there is a unit amount of each good. Let u_{ij} be the utility derived by agent i on receiving one unit of good j ; w.l.o.g., we will assume that u_{ij} is integral. If x_{ij} is the amount of good j that agent i gets, for $1 \leq j \leq g$, then the total utility derived by her is

$$v_i(x) = \sum_{j \in G} u_{ij} x_{ij}.$$

Finally, we assume that each agent has an initial endowment of these goods; the total amount of each good possessed by the agents is 1 unit.

W.l.o.g. we may assume that each good is desired by at least one agent and each agent desires at least one good, i.e.,

$$\forall j \in G, \exists i \in B : u_{ij} > 0 \quad \text{and} \quad \forall i \in B, \exists j \in G : u_{ij} > 0.$$

If not, we can remove the good or the agent from consideration.

The question is to find prices for these goods so that if each agent sells her entire initial endowment at these prices and uses the money to buy an optimal bundle of goods, the market clears exactly, i.e., there is no deficiency or surplus of any good. Such prices are called *equilibrium prices*.

The Arrow-Debreu market model gives one mechanism by which the agents can redistribute

goods to achieve higher utilities. Another mechanism is to view this setup as a Nash bargaining game as follows. For each $i \in B$, let c_i denote the utility derived by agent i from her initial endowment. Regard this as agent i 's disagreement utility and redistribute the goods in accordance with the Nash bargaining solution.

We will define game **ADNB** in a slightly more general manner: we will assume that the disagreement utilities, c_i 's, are arbitrary numbers specified in the particular instance. (As stated in the Introduction, we will not deal with the nonsymmetric extension of **ADNB** in this paper.) Clearly, the Nash bargaining solution is the optimal solution to the following convex program:

$$\begin{aligned}
& \text{maximize} && \sum_{i \in B} \log(v_i - c_i) && (5) \\
& \text{subject to} && \forall i \in B : v_i = \sum_{j \in G} u_{ij} x_{ij} \\
& && \forall j \in G : \sum_{i \in B} x_{ij} \leq 1 \\
& && \forall i \in B, \forall j \in G : x_{ij} \geq 0
\end{aligned}$$

The KKT conditions for this program are:

- (1) $\forall j \in G : p_j \geq 0$.
- (2) $\forall j \in G : p_j > 0 \Rightarrow \sum_{i \in B} x_{ij} = 1$.
- (3) $\forall i \in B, \forall j \in G : p_j \geq \frac{u_{ij}}{v_i - c_i}$.
- (4) $\forall i \in B, \forall j \in G : x_{ij} > 0 \Rightarrow p_j = \frac{u_{ij}}{v_i - c_i}$.

Theorem 2 *If all utilities u_{ij} 's for an **ADNB** instance are rational and the instance is feasible, then the Nash bargaining solution and the duals (prices) are also rational. Moreover, they can be written using polynomially many bits in the length of the instance.*

Proof : We will show that the Nash bargaining solution and the duals (prices) are solutions to a system of linear equations. For each good j , let $q_j = 1/p_j$ be a new variable and let k be the number of nonzero x_{ij} 's in a solution to convex program (5). Now, the system will consist of $g + k$ equations over $g + k$ unknowns. The latter are the g q_j 's and the k nonzero x_{ij} 's. For each good $j \in G$, there is one equation corresponding to KKT condition (2), and for each nonzero x_{ij} there is one equation corresponding to KKT condition (4). \square

5 Resource Allocation Nash Bargaining Games

In this section, we define a subclass of LNB, which we call *resource allocation Nash and nonsymmetric bargaining games* and denote by RNB. Our definition of this class is motivated

by the notion of resource allocation markets given by Kelly [Kel97] and a slight extension given in [KV] (in which agents may derive different utilities from different objects). The special feature of this subclass of LNB is that, as shown in Section 7, finding a solution to a game in RNB reduces to finding an equilibrium in a suitably defined flexible budget market. The game **ADNB** lies in this class.

RNB consists of games of the following form. Let B be a set of agents, $|B| = n$, and G a set of divisible goods, $|G| = g$. Assume that the amount of good $j \in G$ available is b_j . Each agent can “make” *objects* using the goods; let T_i denote the set of objects that agent i can make. Assume that in order to make one unit of object $k \in T_i$, i must use a_{ijk} amount of good j , for each $j \in G$, and she derives utility u_{ik} from one unit of this object. Note that i is allowed to make fractional units of any of the objects. For each $k \in T_i$, if y_{ik} denotes the total amount of object k that i makes, then the total utility derived by i is $\sum_{k \in T_i} u_{ik} y_{ik}$. The assignments to y_{ik} 's will be called *allocations*. We assume that all these parameters, i.e., a_{ijk}, b_j, u_{ik} are integral.

Consider all possible ways of distributing the goods among the agents, and for each, the utilities derived by all the agents. Let \mathcal{N} denote the set of all utility vectors obtainable in this manner and let $c \in \mathcal{N}$ denote the disagreement point. The problem is to find the Nash bargaining solution. For the nonsymmetric extension, we are also given the clout of each player via w . In some games, T_i may be exponentially large in n and g , though it may be succinctly specified, as in game **RNB2** below.

Besides **ADNB**, RNB contains the following natural games. The games **RNB1** and **RNB2** come directly from Kelly's work [Kel97] on obtaining a mathematical theory of TCP congestion control; a special case of **RNB1** is studied in Section 16 and a special case of **RNB2** is studied in [Vaz09].

1. **Game RNB1:** We are given a directed graph $G = (V, E)$, with capacities of edges specified. Agents are source-sink pairs of nodes, $(s_1, t_1), \dots, (s_k, t_k)$. An object for agent (s_i, t_i) is an $s_i - t_i$ path of unit capacity.
2. **Game RNB2:** Same as above, except the graph is undirected.
3. **Game RNB3:** We are given a directed graph $G = (V, E)$, with capacities of edges specified. Given a directed graph $G = (V, E)$, E is the set of resources, with capacities specified. Agents are $A \subset V$. For $s \in A$ objects are branchings rooted at s and spanning all V .

The nonsymmetric bargaining solution to a generic RNB game is the optimal solution to the following convex program.

$$\begin{aligned}
& \text{maximize} && \sum_{i \in B} w_i \log(v_i - c_i) && (6) \\
& \text{subject to} && \forall i \in B : v_i = \sum_{k \in T_i} u_{ik} y_{ik} \\
& && \forall j \in G : \sum_{i \in B} \sum_{k \in T_i} a_{ijk} y_{ik} \leq b_j
\end{aligned}$$

$$\forall i \in B, \forall k \in T_i : y_{ik} \geq 0$$

We will say that this program is *feasible* iff the underlying Nash bargaining game is feasible. Let p_j , $j \in G$, be the Lagrange variables corresponding to the second set of constraints; we will interpret these as prices of the goods. Observe that these constraints are packing constraints, i.e., of the type \leq , with all coefficients and the rhs nonnegative and the variables being constrained to be nonnegative. Denote the cost of making one unit of object k by agent i at prices \mathbf{p} by $P(i, k, \mathbf{p})$. Clearly,

$$P(i, k, \mathbf{p}) = \sum_{j \in G} a_{ijk} p_j.$$

By the KKT conditions, optimal solutions to y_{ik} 's and p_j 's must satisfy:

- (1) $\forall j \in G : p_j \geq 0$.
- (2) $\forall j \in G : p_j > 0 \Rightarrow \sum_{i \in B} \sum_{k \in T_i} a_{ijk} y_{ik} = b_j$.
- (3) $\forall i \in B, \forall j \in G : P(i, k, \mathbf{p}) \geq \frac{w_i \cdot u_{ik}}{v_i - c_i}$.
- (4) $\forall i \in B, \forall j \in G : y_{ik} > 0 \Rightarrow P(i, k, \mathbf{p}) = \frac{w_i \cdot u_{ik}}{v_i - c_i}$.

Strict concavity of the objective function of program (6) and the fourth KKT condition imply the following in a straightforward manner (e.g., see Theorem 1 in [Vaz07]).

Proposition 3 *If convex program (6) is feasible, it has a unique optimal solution $\mathbf{v} \in \mathcal{N}$. Also, $\forall i \in B, \forall k \in T_i : P(i, k, \mathbf{p})$ is unique.*

6 Fisher's Model and its Extension via Flexible Budgets

First we state Fisher's market model for the case of linear utilities [BS00]. Consider a market consisting of a set of n buyers $B = \{1, 2, \dots, n\}$, and a set of g divisible goods, $G = \{1, 2, \dots, g\}$; we may assume w.l.o.g. that there is a unit amount of each good. Let m_i be the money possessed by buyer i , $i \in B$. Let u_{ij} be the utility derived by buyer i on receiving one unit of good j . Thus, if x_{ij} is the amount of good j that buyer i gets, for $1 \leq j \leq g$, then the total utility derived by i is

$$v_i(x) = \sum_{j=1}^g u_{ij} x_{ij}.$$

The problem is to find prices $\mathbf{p} = \{p_1, p_2, \dots, p_g\}$ for the goods so that when each buyer is given her utility maximizing bundle of goods, the market clears, i.e., each good having a positive price is exactly sold, without there being any deficiency or surplus. Such prices are called *market clearing prices* or *equilibrium prices*.

The following is the Eisenberg-Gale convex program. Using KKT conditions, one can show that its optimal solution is an equilibrium allocation for Fisher's linear market and the Lagrange variables corresponding to the inequalities give equilibrium prices of goods (e.g., see Theorem 5.1 in [Vaz07]).

$$\begin{aligned}
& \text{maximize} && \sum_{i \in B} m_i \log v_i && (7) \\
& \text{subject to} && \forall i \in B : v_i = \sum_{j \in G} u_{ij} x_{ij} \\
& && \forall j \in G : \sum_{i \in B} x_{ij} \leq 1 \\
& && \forall i \in B, \forall j \in G : x_{ij} \geq 0
\end{aligned}$$

Below we will define a flexible budget market corresponding to each game in RNB. However, first we introduce such a market by slightly modifying Fisher's linear case; this market will be used for solving **ADNB**.

The utility functions of buyers are as before. The two main differences are that each buyer i now has a parameter c_i giving a strict lower bound on the amount of utility she wants to derive and buyers do not come to the market with a fixed amount of money, but instead the money they spend is a function of prices of goods in the following manner. Given prices \mathbf{p} for the goods, define the *maximum bang-per-buck* of buyer i to be

$$\gamma_i = \max_j \left\{ \frac{u_{ij}}{p_j} \right\}.$$

Now, buyer i 's money is defined to be $m_i = 1 + \frac{c_i}{\gamma_i}$.

Let us say that set $S_i = \operatorname{argmax}_j \left\{ \frac{u_{ij}}{p_j} \right\}$ constitutes i 's *maximum bang-per-buck goods*. Clearly, at prices \mathbf{p} , any utility maximizing bundle of goods for i will consist of goods from S_i worth m_i . The problem again is to find market clearing or equilibrium prices. Observe that in an equilibrium, if it exists, each buyer i will derive utility at least c_i .

Next, we define a flexible budget market corresponding to an arbitrary game in RNB. As in the definition of an RNB game, let B be a set of buyers and G a set of goods. The parameters w_i, a_{ijk}, u_{ik} and sets T_i will also have the same meaning as in the definition of an RNB game. The only difference is that instead of representing the disagreement utility of buyer i , parameter c_i will give a strict lower bound on the utility that buyer i wants to derive.

Once again, the money spent by buyers in the market is a function of prices of goods. Relative to prices \mathbf{p} , define the *maximum bang-per-buck* of buyer i to be

$$\gamma_i = \max_{k \in T_i} \left\{ \frac{u_{ik}}{P(i, k, \mathbf{p})} \right\}.$$

Now, the money of agent i is defined to be $m_i = w_i + \frac{c_i}{\gamma_i}$. We will denote by y_{ik} the allocation of object k to buyer i , for $k \in T_i$ and by $S_i \subseteq T_i$ the set of objects which provide i maximum bang-per-buck at prices \mathbf{p} .

At any given prices, each agent is interested in maximizing the total utility accrued by her. Clearly, she will spend all her money to buy maximum bang-per-buck objects. The problem again is to find market clearing prices – prices such that if each buyer is sold a bundle of objects that maximizes her utility, all goods having positive price are sold exactly, with there being no deficiency or surplus.

7 The Reduction

In this section we show how to transform an instance I of RNB to a flexible budget market \mathcal{M} so that I is feasible iff \mathcal{M} is and the equilibrium allocation and prices for \mathcal{M} give us the solution to I . Assume that in this instance $B, G, c_i, T_i, a_{ijk}, u_{ik}$ are as defined in Section 5. These sets and parameters will carry over to specify the flexible budget market \mathcal{M} , as defined in Section 6.

Theorem 4 *Instance I is feasible iff \mathcal{M} is. Moreover, if I and \mathcal{M} are both feasible, then allocations \mathbf{y} and dual \mathbf{p} are optimal for RNB game I iff they are equilibrium allocations and prices for the flexible budget market \mathcal{M} .*

Proof : (\Rightarrow) First assume that I is feasible and that allocations \mathbf{y} and dual \mathbf{p} are optimal for RNB game I . The latter must satisfy KKT conditions for convex program (6).

By the second KKT condition, each good having a positive price is fully sold. Assume that $y_{ik} > 0$. Then, by the definition of γ_i and the fourth KKT condition,

$$\gamma_i = \frac{u_{ik}}{P(i, k, \mathbf{p})} = \frac{v_i - c_i}{w_i}.$$

The money of buyer i at prices \mathbf{p} in market \mathcal{M} is defined to be $m_i = w_i + c_i/\gamma_i$. The money spent by i in market \mathcal{M} is:

$$\begin{aligned} \sum_{k \in T_i} y_{ik} P(i, k, \mathbf{p}) &= \sum_{k \in T_i} \frac{y_{ik} u_{ik}}{\gamma_i} \\ &= \frac{w_i}{v_i - c_i} \sum_{k \in T_i} y_{ik} u_{ik} = \frac{w_i v_i}{v_i - c_i} = w_i + \frac{w_i c_i}{v_i - c_i} = w_i + \frac{c_i}{\gamma_i} = m_i. \end{aligned}$$

Furthermore, by the third and fourth KKT conditions, i buys only her maximum bang-per-buck objects. Hence, she makes an optimal bundle. This proves that \mathbf{y} and \mathbf{p} constitute equilibrium allocations and prices for market \mathcal{M} .

(\Leftarrow) Next, assume that \mathcal{M} is feasible and that \mathbf{y} and \mathbf{p} are equilibrium allocations and prices for market \mathcal{M} . Now, \mathbf{y} is clearly feasible for program (6); we will show that \mathbf{y} and \mathbf{p} satisfy all KKT conditions for this program. The first two conditions are obvious.

Since i makes an optimal bundle of objects at prices \mathbf{p} ,

$$y_{ik} > 0 \Rightarrow \frac{u_{ik}}{P(i, k, \mathbf{p})} = \gamma_i.$$

Since i spends all her money,

$$m_i = w_i + \frac{c_i}{\gamma_i} = \sum_{k \in T_i} y_{ik} P(i, k, \mathbf{p}) = \sum_{k \in T_i} y_{ik} \frac{u_{ik}}{\gamma_i} = \frac{v_i}{\gamma_i}.$$

Therefore, $\gamma_i = \frac{v_i - c_i}{w_i}$. This gives the last two conditions as well. \square

8 A Test for Equilibrium Prices

In the next 5 sections, we will present an efficient algorithm for solving an instance of the game **ADNB** by first reducing it to a flexible budget market, say \mathcal{M} .

In this section, we give an efficient algorithm for the following simpler question: Given prices $\mathbf{p} = \{p_1, \dots, p_g\}$ for the goods in \mathcal{M} , determine if these are equilibrium prices, and if so, find an equilibrium allocation.

First construct a directed network $N(\mathbf{p})$ as follows. $N(\mathbf{p})$ has a source s , a sink t and it has vertex subsets B and G corresponding to the buyers and goods, respectively. For each good $j \in G$, there is an edge (s, j) of capacity p_j , and for each buyer $i \in B$, there is an edge (i, t) of capacity m_i , where $m_i = 1 + c_i/\gamma_i$ is i 's money in \mathcal{M} . For each $i \in B$ and $j \in S_i$, we will say that edge (j, i) is *active*. The network $N(\mathbf{p})$ will contain all active edges, each with infinite capacity.

Lemma 5 *Prices \mathbf{p} are equilibrium prices for \mathcal{M} iff the two cuts $(s, B \cup G \cup t)$ and $(s \cup B \cup G, t)$ are min-cuts in network $N(\mathbf{p})$. Moreover, if \mathbf{p} are equilibrium prices, then the set of equilibrium allocations corresponds exactly to max-flows in $N(\mathbf{p})$.*

The proof of this lemma is straightforward using the transformation between a max-flow f in $N(\mathbf{p})$ and an allocation x in \mathcal{M} given by $x_{ij} = f(j, i)/p_j$. Now, the condition that $(s, B \cup G \cup t)$ and $(s \cup B \cup G, t)$ are min-cuts in network $N(\mathbf{p})$, and hence saturated by f , corresponds to all goods being sold and all buyers' money being spent. The fact that (j, i) is an edge in $N(\mathbf{p})$ iff $j \in S_i$ ensures that buyers get only their maximum bang-per-buck goods.

The next lemma gives the combinatorial object that yields equilibrium prices. Assume that \mathbf{p}^* are equilibrium prices, i.e., $N(\mathbf{p}^*)$ satisfies the condition in Lemma 5. Let H be the uncapacitated directed subgraph of $N(\mathbf{p}^*)$ induced on $B \cup G$.

Lemma 6 *Given H , we can find \mathbf{p}^* in strongly polynomial time.*

Proof : Consider the connected components of H after ignoring directions on its edges. In each component, pick a good and assign it price p , say. The prices of the rest of the goods in this component can be obtained in terms of p . The bang-per-buck, and hence the money, of each buyer in this component can also be obtained in terms of p . Finally, by equating the money of all buyers in this component with the total value of all goods in this component, we can compute p . \square

The following characterization will be useful in Algorithm 8.

Lemma 7 *Let \mathbf{p} be equilibrium prices for market \mathcal{M} and let \mathbf{q} be arbitrary prices. If $(s, B \cup G \cup t)$ is a min-cut in network $N(\mathbf{q})$, then for each good j , $q_j \leq p_j$.*

Proof : Let us assume that there are goods j such that $q_j > p_j$ and yet $(s, B \cup G \cup t)$ is a min-cut in $N(\mathbf{q})$. Let

$$\theta = \max_{j \in G} \left\{ \frac{q_j}{p_j} \right\} \quad \text{and} \quad S = \{j \in G \mid q_j = \theta p_j\}.$$

Clearly, $\theta > 1$.

Let T_p and T_q be the set of buyers who are interested in goods in S at prices p and q , respectively. Since S represents the set of goods whose prices increase by the largest factor in going from prices \mathbf{p} to \mathbf{q} , $T_q \subseteq T_p$. We claim that a buyer $i \in T_q$ is not interested in any goods in $G - S$ at prices \mathbf{p} (because otherwise at prices \mathbf{q} , i will not be interested in any goods in S , since their prices increased the most). Therefore, in any max-flow in $N(\mathbf{p})$, all flow going through nodes in T_q must also have used nodes in S . Therefore,

$$\sum_{j \in S} p_j \geq \sum_{i \in T_q} \left(1 + \frac{c_i}{\gamma_i} \right),$$

where γ_i is the maximum bang-per-buck of buyer i w.r.t. prices \mathbf{p} . Multiplying this inequality by θ and using the fact that $\theta > 1$ we get,

$$\theta \sum_{j \in S} p_j \geq \sum_{i \in T_q} \left(\theta + \frac{c_i \theta}{\gamma_i} \right) > \sum_{i \in T_q} \left(1 + \frac{c_i \theta}{\gamma_i} \right),$$

Observe that the maximum bang-per-buck of buyer $i \in T_q$ w.r.t. prices \mathbf{q} is γ_i/θ . Therefore, the last inequality implies that w.r.t. prices \mathbf{q} , the total value of goods in S is strictly more than the total value of money possessed by buyers in T_q . On the other hand, since in $N(\mathbf{q})$ all flow using nodes of T_q goes through S , we get that $(s, B \cup G \cup t)$ is not a min-cut in network $N(\mathbf{q})$, leading to a contradiction. \square

9 Solution to ADNB in the Limit

Assume that we are given an instance of game **ADNB** that is feasible and let \mathcal{M} be the flexible budget market obtained from it. In this section, we present an algorithm that converges to the equilibrium of \mathcal{M} in the limit. The polynomial time algorithm for finding the equilibrium is quite involved and we hope the present section will serve as a suitable warm-up.

Algorithm 8 will use the DPSV algorithm as a subroutine of [DPSV08]. When this subroutine is called, we assume that the money of agents is fixed and is specified in the the vector \mathbf{m} .

Let $N'(\mathbf{p})$ denote the network for the case that the money of agents is fixed and specified by vector \mathbf{m} ; this network differs from network $N(\mathbf{p})$ only in that the capacities of edges going from buyers to t are specified by \mathbf{m} , rather than being defined as a function of the prices.

Algorithm 8 (Solution to ADNB in the Limit)

1. Initialization: $\forall i \in B : m_i \leftarrow 1$.
2. Compute equilibrium prices, \mathbf{p} , for market (\mathbf{u}, \mathbf{m}) using the DPSV algorithm.
3. For each $i \in B$, compute γ_i w.r.t. prices \mathbf{p} , and set $m'_i \leftarrow 1 + \frac{c_i}{\gamma_i}$.
4. If $\mathbf{m}' = \mathbf{m}$ then output equilibrium allocations and **HALT**.
Else, update \mathbf{m} to \mathbf{m}' and go to Step 2.

Let \mathbf{p}^* and \mathbf{m}^* be the equilibrium prices and moneys for the flexible budget market \mathcal{M} , and let $\mathbf{p}^{(k)}$ and $\mathbf{m}^{(k)}$ denote the prices and moneys computed by the algorithm in the k -th iteration, $k \geq 1$.

Lemma 9 $\mathbf{p}^{(k)}$ and $\mathbf{m}^{(k)}$ are monotone increasing and bounded, in each coordinate, by \mathbf{p}^* and \mathbf{m}^* , respectively.

Proof : We will use the following 2 facts. First, the DPSV algorithm maintains the following throughout:

Invariant: W.r.t. current prices, \mathbf{p} , $(s, B \cup G \cup t)$ is a min-cut in network $N'(\mathbf{p})$.

Second, if \mathbf{p} are equilibrium prices for money \mathbf{m} and if \mathbf{m}' is at least as large as \mathbf{m} in each component, then the equilibrium prices for money \mathbf{m}' cannot be smaller than \mathbf{p} in any component.

Consider the following induction hypothesis:

- the algorithm given above maintains the Invariant throughout
- $\mathbf{p}^{(k)}$ is monotone increasing (hence, for each agent i , γ_i is monotonically decreasing).
- $\mathbf{m}^{(k)}$ is monotone increasing.

It is easy to carry out this induction simultaneously for all 3 assertions.

Using the first assertion and Lemma 7, $\mathbf{p}^{(k)}$ is bounded in each coordinate by \mathbf{p}^* . Now, using the formula for money in flexible budget markets, it is easy to see that $\mathbf{m}^{(k)}$ is bounded in each coordinate by \mathbf{m}^* . \square

Theorem 10 *Algorithm 8 converges to the equilibrium prices and moneys of market \mathcal{M} in the limit.*

Proof : We will use the following fact: for the linear case of Fisher’s model, the analog of Lemma 5 holds, i.e., if \mathbf{p} are equilibrium prices for money \mathbf{m} , then in network $N'(\mathbf{p})$, $(s, B \cup G \cup y)$ and $(s \cup B \cup G, t)$ must both be min-cuts (for a proof, see Lemma 5.2 in [Vaz07]). Since $\mathbf{p}^{(k)}$ and $\mathbf{m}^{(k)}$ are monotone increasing and bounded, they must converge. Let $\mathbf{p}^{(0)}$ and $\mathbf{m}^{(0)}$ be their limit points. W.r.t. these prices and moneys, it must be the case that for each $i \in B$, $m_i = 1 + c_i/\gamma_i$ and $(s, B \cup G \cup t)$ and $(s \cup B \cup G, t)$ must both be min-cuts in the corresponding network (by the fact stated above). Using lemma 5 we get that $\mathbf{p}^{(0)}$ and $\mathbf{m}^{(0)}$ are equilibrium prices and moneys for market \mathcal{M} . \square

Finally, by Theorem 4 we get.

Corollary 11 *Algorithm 8 converges to the Nash bargaining solution for ADNB.*

10 High-Level Idea of the Algorithm for ADNB

We start with some preliminaries. At any point in the algorithm, we will denote by \mathbf{p} the current prices of goods and by $N(\mathbf{p})$ the corresponding network as defined in Section 8. For agent i , we will denote c_i/γ_i by α_i ; therefore, $m_i = 1 + \alpha_i$. For $J \subseteq G$, define $p(J) = \sum_{j \in J} p_j$ and $\Gamma(J) = \{i \in B \mid \exists j \in G \text{ s.t. } (j, i) \in N(\mathbf{p})\}$. Similarly, for $I \subseteq B$, define $\alpha(I) = \sum_{i \in I} \alpha_i$, $m(I) = \sum_{i \in I} m_i$ and $\Gamma(I) = \{j \in G \mid \exists i \in B \text{ s.t. } (j, i) \in N(\mathbf{p})\}$.

We will impose the following condition throughout; by Lemma 7, it will ensure that if the given game is feasible, the equilibrium price of no good is ever exceeded.

Invariant: W.r.t. current prices, \mathbf{p} , $(s, B \cup G \cup t)$ is a min-cut in network $N(\mathbf{p})$.

Using the DPSV algorithm, our algorithm first finds prices for which the Invariant is guaranteed to hold; at these prices, buyers will have surplus money left over. It then raises prices in such a way that the surplus vanishes and equilibrium is found. This overall scheme suffers from 2 major difficulties, neither of which arises in [DPSV08]:

1. As we raise the prices of goods, the money of buyers also changes, and in particular, the surplus of some buyers may actually increase.
2. We need to determine if the given instance is feasible.

Let us look closely at the first difficulty, assuming the following way of raising prices which is very basic to our algorithm. Multiply the price of each good by a variable x , initialize x to 1, and raise x (we will also need to consider decreasing x). How does the surplus of each buyer change as a function of x ? Clearly, this depends on the particular max-flows chosen for various values of x . Let f be any max-flow chosen in $N(\mathbf{p})$, i.e., for $x = 1$. Define $x \cdot f$ to be the flow obtained by multiplying by x the flow on each edge w.r.t. f . The next lemma shows that this is a max-flow in $N(x\mathbf{p})$; we will choose it as the max-flow in $N(x\mathbf{p})$.

At prices \mathbf{p} , let the money of buyer i be $m_i = 1 + \alpha_i$. Then, i 's surplus will be $m_i - f(i, t) = 1 + \alpha_i - f(i, t)$, where $f(i, t)$ is the flow on edge (i, t) . We next introduce a crucial notion. Define i 's *1-surplus* to be 1 less than i 's surplus, i.e., $\beta_i = m_i - 1 - f(i, t) = \alpha_i - f(i, t)$.

For the purposes of the next lemma (and with a slight abuse of notation), let $\beta_i(x)$ denote i 's 1-surplus w.r.t. flow $x \cdot \mathbf{p}$. Define

$$b = \min_{i \in B} \left\{ \frac{-1}{\beta_i} \right\}.$$

Clearly, $b > 1$.

Lemma 12 *Flow $x \cdot f$ is a max-flow in $N(x\mathbf{p})$ for $0 < x \leq b$. Furthermore, for each $i \in B$, $\beta_i(x) = x\beta_i$.*

Proof : Since the Invariant holds and f is a max-flow in $N(\mathbf{p})$, the cut $(s, J \cup I \cup t)$ is saturated by f . Now, it suffices to show that $x \cdot f$ is a feasible flow in $N(x\mathbf{p})$, since it must also saturate this cut. For this, we need to show that for each buyer $i \in B$, edge (i, t) is not over saturated.

Now, $\beta_i = \alpha_i - f(i, t)$. Therefore, the surplus on edge (i, t) w.r.t. flow $x \cdot f$ is $1 + x(\alpha_i - f(i, t)) = 1 + x\beta_i \geq 0$ for $0 < x \leq b$. Hence, edge (i, t) is not over saturated. Furthermore, $\beta_i(x) = x\beta_i$. \square

By Lemma 12, i 's surplus decreases with increasing x iff $\beta_i < 0$. If so, let us say that buyer i is *good* w.r.t. prices \mathbf{p} . Thus, the first difficulty can be overcome if we can find prices for which all buyers are good. Do such prices always exist? This brings us to the resolution of the second difficulty.

In Section 12 we give an LP whose objective function value is negative iff the given game is feasible. Via this LP, we show that if we can find prices such that

$$\sum_{i \in B} \alpha_i \geq \sum_{j \in G} p_j,$$

then the given game is infeasible (see Lemma 20).

Our algorithm consist of 2 stages; Stage I solves the decision problem and, if the market is found to be feasible, Stage II finds an equilibrium for it. Stage I terminates either with all buyers rendered good or with the above-stated condition holding. In the former case, by raising prices appropriately, the surplus of buyers can be decreased until equilibrium prices are found – this is done in Stage II.

10.1 The primal-dual paradigm in the enhanced setting and a high-level description of Stage II

Almost all primal-dual algorithms, in exact as well as approximation algorithms, operate by raising dual variables via a greedy process. The only exception is Edmonds' algorithm for

maximum weight matching in general graphs [Edm65], which finds an optimal dual by a process that increases and decreases duals.

A greedy dual growth process is too restrictive – the fact that a raised dual is “bad”, in the sense that it “obstructs” other duals, which could have led to a larger overall dual solution, may become clear only later in the run of the algorithm. In view of this, the issue of using more sophisticated dual growth processes received a lot of attention, especially in the area of approximation algorithms, since obtaining a better dual would lead to a better approximation factor.

The problem with such a process is that it will make primal objects go tight and loose and all such events will have to be accounted for in the running time. The perfect combinatorial structure of matching supports such an accounting; however, thus far, all attempts at making such a scheme work out for other problems have failed.

The fundamental difference between complementary slackness conditions for linear programs and KKT conditions for nonlinear convex programs is that whereas the former do not involve both primal and dual variables simultaneously in an equality constraint (obtained by assuming that one of the variables takes a non-zero value), the latter do.

Once we determine that the given instance is feasible, our dual growth process in Stage II is also greedy and the prices of goods are never decreased. Yet, because of the more complex nature of KKT conditions, edges in network $N(\mathbf{p})$ appear and disappear as the prices are raised. Hence, we are forced to carry out the difficult accounting alluded to above for bounding the running time.

We next point out which KKT conditions our algorithm enforces and which ones it relaxes, as well as the exact mechanism by which it satisfies the latter. Throughout our algorithm, we enforce the first two conditions listed in Section 4 and we relax the third and fourth KKT conditions to the following:

- $\forall i \in B, \forall j \in G : \frac{p_j}{-\beta_i} \geq \frac{u_{ij}}{v_i - c_i}$.
- $\forall i \in B, \forall j \in G : x_{ij} > 0 \Rightarrow \frac{p_j}{-\beta_i} = \frac{u_{ij}}{v_i - c_i}$.

Recall that at the start of Stage II, for each $i, \beta_i < 0$. Since in this stage, in each iteration we only raise x , by Lemma 12 this condition is maintained throughout the stage.

Lemma 13 *Enforcing the relaxed KKT conditions stated above for convex program (5) amounts to finding an equilibrium for a flexible budget market in which the money of buyer i is redefined to be*

$$e_i = -\beta_i + \frac{c_i}{\gamma_i}.$$

Proof : The money spent by buyer i assuming relaxed KKT conditions is

$$e_i = \sum_j p_j x_{ij} = \sum_j \frac{(-\beta_i) u_{ij} x_{ij}}{v_i - c_i} = (-\beta_i) \frac{c_i}{v_i - c_i} = (-\beta_i) \left(1 + \frac{v_i}{v_i - c_i} \right) = -\beta_i + \frac{c_i}{\gamma_i},$$

where we have used the fact that

$$\gamma_i = \frac{u_{ij}}{p_j} = \frac{v_i - c_i}{(-\beta_i)}.$$

□

Thus, at a high level, the job of Stage II is to iteratively decrease all β_i 's until they drop to -1 . At that point, the surplus money of each buyer vanishes, the condition in Lemma 5 starts holding, and equilibrium has been attained. The obvious potential function for measuring progress would have been $\sum_{i=1}^n (1 + \beta_i)$. However, we do not know of a way of ensuring that the total number of events, alluded to above, can be polynomially bounded with this potential function.

Instead, we use the following potential function

$$\Phi = \sum_{i=1}^n (1 + \beta_i)^2,$$

and we give a process by which Φ decreases by an inverse polynomial fraction in each phase, until it drops to zero. (The potential function Φ given in Section 13.2 is equivalent to the one given above.) This involves using the notion of balanced flows described in Section 11.1.

Primal-dual algorithms in the two previous settings of exact and approximation algorithms satisfy complementary conditions in *discrete steps*, i.e., in each iteration, the algorithm satisfies at least one new condition. Each iteration is implemented in strongly polynomial time thereby leading to a strongly polynomial algorithm. On the other hand, our algorithm satisfies KKT conditions *gradually*: The appropriate conditions for each buyer are relaxed and as the algorithm proceeds, it satisfies the conditions to a greater extent. This is the second point of departure from previous primal-dual algorithms. It has repercussions to the running time of the algorithm: The algorithm terminates when the potential is inverse polynomial in the *length of the input* and not just n . Therefore, even though each phase is implemented in strongly polynomial time, the algorithm is polynomial time but not strongly polynomial. Obtaining a strongly polynomial algorithm remains an important open problem.

10.2 High-level description of Stage I

Stage I first finds equilibrium prices under the assumption that the money of each buyer is fixed at 1 – this is a linear Fisher market and the DPSV algorithm yields the equilibrium. With these prices, the Invariant is guaranteed to hold for the flexible budget market. Therefore, if this market is feasible, we must raise prices to arrive at the equilibrium. Interestingly enough, Stage I needs to *lower* prices of goods to accomplish its job; indeed, this is perhaps the most counter-intuitive aspect of Stage I.

The job of Stage I is not to find the equilibrium; instead, it is to either find prices that render all buyers good or that ensure

$$\sum_{i \in B} \beta_i = \sum_{i \in B} \alpha_i - \sum_{j \in G} p_j \geq 0,$$

thereby proving infeasibility. Recall that buyer i is defined to be *good* if $\beta_i < 0$.

Another intriguing aspect of Stage I is that although its goals are very different from that of Stage II, and it is not driven by a desire to satisfy KKT conditions, the kinds of techniques used in it are the same as in Stage II. This includes the use of balanced flows, the use of l_2 norm in the potential function, and a similar process that decreases the latter by an inverse polynomial fraction in each phase.

The buyers are partitioned into 2 sets: the good buyers and the rest. One way to view the operation of Stage I is as a tug-of-war between these two sets of buyers. The algorithm decreases the prices of certain goods desired by good buyers. This helps towards reaching the infeasibility condition stated above. However, as new edges enter the network and a balanced flow is recomputed, buyers may move between the 2 sets. Stage I terminates when either all buyers are rendered good or the infeasibility condition starts holding. Let us argue that one of these outcomes must happen. Suppose in a run of Stage I, all buyers never get rendered good. Then it must be the case that eventually the prices of goods desired by good buyers must become zero and their β_i 's become 0, i.e., the infeasibility condition must start holding.

11 Details of the Algorithm for ADN

11.1 Balanced flow

The polynomial time implementation of the ideas given in Section 10 requires the notion of *balanced flow* from [DPSV08]. We will follow the exposition in [Vaz07] and refer the reader to this chapter for all facts stated below without proof.

For simplicity, denote the current network, $N(\mathbf{p})$, by simply N . Given a feasible flow f in N , let $R(f)$ denote the residual graph w.r.t. f . Define the *surplus* of buyer i w.r.t. flow f in network N , $\theta_i(N, f)$, to be the residual capacity of the edge (i, t) with respect to flow f in network N , i.e., m_i minus the flow sent through the edge (i, t) . The *surplus vector w.r.t. flow f* is defined to be $\theta(N, f) := (\theta_1(N, f), \theta_2(N, f), \dots, \theta_n(N, f))$. Let $\|v\|$ denote the l_2 norm of vector v . A *balanced flow* in network N is a flow that minimizes $\|\theta(N, f)\|$. A balanced flow must be a max-flow in N because augmenting a given flow can only lead to a decrease in the l_2 norm of the surplus vector.

A balanced flow can be computed in N using at most n max-flow computations. It is easy to see that all balanced flows in N have the same surplus vector. The key property of a balanced flow that our algorithm will rely on is that a maximum flow f in N is balanced iff it satisfies Property 1:

Property 1: For any two buyers i and j , if $\theta_i(N, f) < \theta_j(N, f)$ then there is no path from node i to node j in $R(f) - \{s, t\}$.

Balanced flows play a crucial role in both stages of our algorithm; moreover in several ways. In Section 11.4, after stating the full algorithm, we state the various ways this notion is used.

11.2 Details of Stage I

A run of Stage I is partitioned into *phases*, which are further partitioned into *iterations*. In Stage I, an iteration ends when a new edge becomes active. A phase ends either when the condition of Step 5 holds or if for some $i \in I, \beta_i \geq 0$. In each iteration, the algorithm computes a balanced flow in the current network, $N(\mathbf{p})$.

The subroutines used in Stage I are:

- **Find sets(I):** Sets $I \subseteq B$ and $J \subseteq G$ are initialized as follows.

$$I \leftarrow \arg \min_{i \in B} \{\beta_i\} \quad \text{and} \quad J \leftarrow (\Gamma(I) - \Gamma(B - I)).$$

All edges are removed from goods in $G - J$ to buyers in I ; this is justified in Lemma 16 below.

- **Update sets(I):** Find the set, I' , of all buyers in $B - I$ such that there is a residual path from a buyer in I to a buyer in I' . Update

$$I \leftarrow (I \cup I') \quad \text{and} \quad J \leftarrow (\Gamma(I) - \Gamma(B - I)).$$

All edges are removed from goods in $G - J$ to buyers in I . Once again, this is justified in Lemma 16.

The choice of set J above ensures that if the prices of goods in J are reduced by an infinitesimally small amount (by decreasing x as stated in Algorithm 18), there is no change in the maximum bang-per-buck goods of buyers in B .

Lemma 14 *In Stage I, in each iteration, for each buyer $i \in I$ there is a good $j \in J$ such that edge (j, i) is in the network.*

Proof : Since $\beta_i < 0$, the balanced flow must be sending flow on some edge (j, i) . If an edge (j, i') to some buyer $i' \in (B - I)$ is also present in the network, then there will be a residual path from i to i' , violating Property 1. Therefore, there is no such edge and $j \in (\Gamma(I) - \Gamma(B - I))$, proving the lemma. \square

In each iteration, the algorithm needs to compute the smallest value of x at which a new edge becomes active. For any one edge this is straightforward; taking the minimum over all relevant edges gives the required value.

At the start of each phase in Stage I, the algorithm tests for the 2 conditions that indicate feasibility and infeasibility, respectively. For this purpose, the test is conducted on buyers remaining in B only. This suffices because the buyers in B' can be made to be consistent with the outcome of the remaining buyers. Thus, if $\forall i \in B, \beta_i < 0$, by restoring the frozen prices of goods in G' , we can ensure that $\forall i \in B', \beta_i < 0$. And if $\sum_{i \in B} \beta_i \geq 0$, then by setting prices of all goods in G' to be zero (which can be done without introducing any new active edges), we can ensure that $\forall i \in B', \beta_i = 0$, thereby ensuring that these buyers don't affect the sum.

11.3 Details of Stage II

A run of Stage II is also partitioned into *phases*, which are further partitioned into *iterations*. An iteration ends when a new edge becomes active and a phase ends when a new set goes tight. We will say that $S \subseteq G$ is a *tight set* if the total price of goods in S exactly equals the money possessed by buyers who are interested in goods in S , i.e., $p(S) = m(\Gamma(S))$. Clearly, if S is tight, buyers in $\Gamma(S)$ must have zero surplus and hence have $\beta_i = -1$. In each iteration, the algorithm computes a balanced flow in the current network, $N(\mathbf{p})$.

The subroutines used in Stage II are:

- **Find sets(II):** Sets $I \subseteq B$ and $J \subseteq G$ are initialized as follows.

$$I \leftarrow \arg \max_{i \in B} \{\theta_i\} \quad \text{and} \quad J \leftarrow \Gamma(I).$$

All edges are removed from goods in J to buyers in $B - I$; this is justified in Lemma 16 below.

- **Update sets(II):** Find the set, I' , of all buyers in $B - I$ that have residual paths to buyers in I . Update

$$I \leftarrow (I \cup I') \quad \text{and} \quad J \leftarrow \Gamma(I).$$

All edges are removed from goods in J to buyers in $B - I$. Once again, this is justified in Lemma 16.

Observe that if (j, i) is the newly added edge, then good j must move from $G - J$ to J , whether or not $I' = \emptyset$. The choice of set J above ensures that if the prices of goods in J are increased by an infinitesimally small amount (by increasing x as stated in Algorithm 18), there is no change in the maximum bang-per-buck goods of buyers in B .

Lemma 15 *In Stage II, in each iteration, for each buyer $i \in (B - I)$ there is a good $j \in (G - J)$ such that edge (j, i) is in the network.*

Proof : Since $\theta_i < 1$, the balanced flow must be sending flow on some edge (j, i) . If $j \in J$, then there will be a residual path from i to a buyer in I , violating Property 1. Therefore, $j \in (G - J)$. \square

Lemma 16 *In Stage I (Stage II), removing all edges from goods in $G - J$ (J) to buyers in I ($B - I$) does not violate the Invariant, and is required for updating the maximum bang-per-buck goods of buyers in I ($B - I$).*

Proof : We will prove the lemma for the statement of Stage II; the proof for Stage I is analogous (it uses Lemma 14 instead of Lemma 15).

By Lemma 15, for each buyer $i \in (B - I)$ there is a good $j \in (G - J)$ such that edge (j, i) is in the network. As soon as the prices of goods in J are raised, i will prefer goods in $G - J$ to those in J , hence the latter edges need to be removed in order to update the maximum bang-per-buck goods of i .

After the operations **Find sets(II)** and *Update sets(II)* have been performed, by Property 1, there are no residual path from buyers in $B - I$ to those in I . Therefore, any edges from J to $B - I$ do not carry any flow in the balanced flow. Hence, removal of these edges will not violate the Invariant. \square

In each iteration, we also need to compute the smallest value of x at which a new edge becomes active or a new set goes tight. The former computation is the same as in Stage I. Next, we show how to find the smallest value of x , say x^* , at which a new set goes tight.

We first recall the following definition from Section 10.

$$b = \min_{i \in I} \left\{ -\frac{1}{\beta_i} \right\}.$$

Clearly, $b > 1$. We will critically use Lemma 12, proved in Section 10, for x in the range $1 \leq x \leq b$.

Lemma 17 $x^* = b$.

Proof : It is easy to see that for $1 \leq x < b$, each edge (i, t) will have positive surplus, implying that there are no tight sets.

Next, assume that $x = b$. Let

$$T = \left\{ i \in I \mid -\frac{1}{\beta_i} = b \right\} \quad \text{and} \quad S = \{j \in J \mid f(j, i) > 0, \text{ for some } i \in T\}.$$

Since f is a balanced flow in $N(\mathbf{p})$, there cannot be an edge (j, i) for $j \in S$ and $i \in (I - T)$ in $N(\mathbf{p})$. This is so because otherwise there would be a path from T to i in the residual graph, contradicting Property 1 (observe that $\beta_i < -1/b$). Therefore, $\Gamma(S) = T$. Moreover, for $i \in T$, the surplus on edge (i, t) w.r.t. flow $x \cdot f$ in $N(x\mathbf{p})$ is $1 + x(-1/b) = 0$. Hence S is a tight set in network $N(x\mathbf{p})$ for $x = b$. \square

11.4 The role of balanced flow

Balanced flow plays the following three, rather diverse, crucial roles in both stages of our algorithm.

1. Ensure that edges, that need to be removed as prices of goods in J are raised, did not carry any flow and hence their removal would not violate the Invariant; this is argued in Lemma 16

2. Ensure that in each iteration, buyers entering I in Stage I (Stage II) have sufficiently large $|\beta_i|$ ($|\theta_i|$); this is established in Lemma 26 (Lemma 36).
3. Prove that sufficient progress is made in an iteration and hence in a phase. This is established in Lemma 27 for Stage I and Lemma 37 for Stage II.

As stated in the Introduction, balanced flow could have been defined without resorting to l_2 norm – as a max-flow that makes the surplus vector lexicographically smallest, after its components are sorted in decreasing order (and hence making the components as balanced as possible). It is easy to prove Property 1 with this definition as well. The first two roles listed above make use of Property 1 only. On the other hand, the third role uses the definition of balanced flow via l_2 norm and as argued in Section 14, the use of l_2 norm seems indispensable.

Algorithm 18 (Initialization and Stage I of the Algorithm for ADNB)

1. Initialization:

- (i) $\forall i \in B : m_i \leftarrow 1.$
- (ii) Use the DPSV algorithm to compute equilibrium prices, $\mathbf{p}.$
- (iii) $\forall i \in B : m_i \leftarrow 1 + \frac{c_i}{\gamma_i}.$
- (iv) $B' \leftarrow \emptyset; \quad G' \leftarrow \emptyset.$

Stage I

2. **(New Phase)** Compute a balanced flow in $N(\mathbf{p}).$

If $\sum_{i \in B} \beta_i \geq 0,$ go to step 7.

3. If $\forall i \in B, \beta_i < 0,$ then:

Restore frozen prices of goods in G' and $G \leftarrow G \cup G'.$

$B \leftarrow B \cup B'.$

Go to Step 1 in Stage II.

4. **Find sets(I).**

5. **(New Iteration)** If $\forall i \in (B - I), \forall j \in J : u_{ij} = 0,$ then:

Freeze prices of goods in $J, \quad G' \leftarrow (G' \cup J)$ and $G \leftarrow (G - J).$

$B' \leftarrow (B' \cup I)$ and $B \leftarrow (B - I).$

Go to Step 2.

6. Multiply the prices of goods in J and α 's of buyers in I by $x.$

Initialize $x \leftarrow 1,$ and decrease x continuously until:

A new edge (j, i) becomes active, for $j \in J$ and $i \in (B - I).$

Add (j, i) to $N(\mathbf{p})$ and compute a balanced flow in it.

If for some $i \in I, \beta_i \geq 0,$ go to Step 2.

Else, **Update sets(I)** and go to Step 5.

7. Output "The game is infeasible".

HALT.

Algorithm 19 (Stage II of the Algorithm for ADNB)

Stage II

1. **(New Phase)** Compute a balanced flow in $N(\mathbf{p})$.
If $\forall i \in B, \theta_i = 0$, go to Step 4.
2. **Find sets(II)**.
3. **(New Iteration)** Multiply prices of goods in J and α 's of buyers in I by x .
Initialize $x \leftarrow 1$, and raise x continuously until:
 - a). A new set $S, \emptyset \subset S \subseteq J$, goes tight.
If so, go to Step 1.
 - b). A new edge (j, i) becomes active, for $j \in (G - J)$ and $i \in I$.
If so, add (j, i) to $N(\mathbf{p})$ and compute a balanced flow in it.
Update sets(II) and go to Step 3.
4. Output the current allocations and prices.
HALT.

12 Proof of Correctness

In order to prove that Algorithm 18 correctly declares a game to be infeasible in Step 7 in Stage I, we will first give an LP for determining if the given game is feasible. Clearly, the game is feasible iff

$$\max_{v \in \mathcal{N}} \min_{i \in B} : (v_i - c_i) > 0.$$

Equivalently, it is feasible iff the optimal solution to the following LP is greater than zero. This LP is best thought of as maximizing t ; however, in order to obtain a convenient dual, we will write it as minimizing $-t$:

$$\begin{aligned} \text{minimize} \quad & -t & (8) \\ \text{subject to} \quad & \forall i \in I : \sum_{j \in J} u_{ij} x_{ij} \geq c_i + t \\ & \forall j \in J : -\sum_{i \in I} x_{ij} \geq -1 \\ & \forall i \in I, \forall j \in J : x_{ij} \geq 0 \end{aligned}$$

Let y_i 's and z_j 's be the dual variables corresponding to the first and second set of inequalities, respectively. The dual program is:

$$\begin{aligned}
& \text{maximize} && \sum_{i \in I} c_i y_i - \sum_{j \in J} z_j && (9) \\
& \text{subject to} && \forall i \in I, \forall j \in J : u_{ij} y_i - z_j \leq 0 \\
& && \sum_{i \in I} y_i = 1 \\
& && \forall i \in I : y_i \geq 0 \\
& && \forall j \in J : z_j \geq 0
\end{aligned}$$

Lemma 20 *If Algorithm 18 declares a game to be infeasible in Step 7 in Stage I, then it is indeed the case; i.e., if $\sum_{i \in B} \beta_i \geq 0$, then the given game is infeasible.*

Proof : Consider the prices of goods when the algorithm transfers control to Step 7. Using these prices, obtain $y_i = 1/\gamma_i$, for $i \in B$, and $z_j = p_j$, for $j \in G$. We will first show that (y, z) is a feasible solution for the dual LP (9). By definition of maximum bang-per-buck,

$$\forall i \in B, \forall j \in G : \gamma_i \geq \frac{u_{ij}}{p_j} \Rightarrow u_{ij} y_i \leq z_j,$$

hence establishing the first set of inequalities. Clearly, by appropriately scaling all the utilities, u_{ij} 's, we can ensure the second constraint of the dual:

$$\sum_{i \in B} \frac{1}{\gamma_i} = 1.$$

The objective function value of this dual solution is

$$\sum_{i \in B} c_i y_i - \sum_{j \in G} z_j = \sum_{i \in B} \alpha_i - \sum_{j \in G} p_j = \sum_{i \in B} \beta_i \geq 0.$$

Therefore, at optimality, $-t \geq 0$, i.e., $t \leq 0$, thereby establishing infeasibility of the game. \square

The proof of the following lemma is straightforward from Lemma 5.

Lemma 21 *If Algorithm 19 terminates in Step 4 in Stage II, then the current allocations and prices form an equilibrium.*

13 Running Time Analysis

We first define some parameters of the given problem instance. Recall that $g = |G|$ and $n = |B|$. Let $U = \max_{i \in B, j \in G} \{u_{ij}\}$, $C = \max_{i \in B} c_i$, and $\Delta = nCU^n$. Let $1/\mu$, $\mu \in \mathbf{Z}^+$, be the lower bound on the equilibrium price of a good, at the end of Initialization, as established in [DPSV08] (see also Theorem 5.1 in [Vaz07]).

The following enhanced version of Lemma 12 will be needed in both stages.

Lemma 22 *Let f be a balanced flow in network $N(\mathbf{p})$. Then, for $0 < x \leq b$, the flow $x \cdot f$ is a balanced flow in $N(x\mathbf{p})$.*

Proof : For $i, j \in B$ assume that $1 + x\beta_i < 1 + x\beta_j$. Since $x > 0$, $1 + \beta_i < 1 + \beta_j$, i.e., w.r.t. flow f in $N(\mathbf{p})$, the surplus of i is smaller than that of j . Since f is a balanced flow in $N(\mathbf{p})$, by Property 1, there is no path from i to j in the residual graph. Therefore, w.r.t. flow $x \cdot f$ in $N(x\mathbf{p})$ also there is no path from i to j in the residual graph. Therefore, flow $x \cdot f$ in $N(x\mathbf{p})$ satisfies Property 1 and hence is a balanced flow. \square

13.1 Stage I

Throughout Stage I, we will consider a partitioning of B into two sets, B_1 and B_2 , containing buyers having $\beta_i < 0$ and $\beta_i \geq 0$, respectively. For Stage I, we will work with the following potential function:

$$\Phi = \sum_{i \in B_1} \beta_i^2.$$

As Stage I proceeds, buyers move from B_1 to B_2 and vice versa. For this reason, it will be convenient to define Φ using an n -dimensional vector, ψ , whose i -th component, $\psi_i = \min(\beta_i, 0)$. Observe that this vector simply zero's out the beta's of buyers in B_2 . Now, an alternative definition for the potential function is:

$$\Phi = \|\psi\|^2.$$

Lemma 23 *In Stage I, a phase consists of at most ng iterations.*

Proof : Each iteration that ends without the phase ending must either move a good from J to $G - J$ or a buyer from $B - I$ to I . Clearly, there can be at most $|J| < |G|$ contiguous iterations of the first type and a total of at most $|B - I_0| < |B|$ iterations of the second type, where I_0 is the set I at the start of the phase. \square

The central fact established below is that Φ drops by a factor of $(1 - 1/(gn^2))$ in a phase (Lemma 28). Towards this end, assume that a given phase consists of k iterations. Let I_0 denote set I at the start of the phase and let I_l denote the set I at the end of the l -th iteration, $1 \leq l \leq k$. Assume that at the start of this phase, $\max_{i \in B_1} \{|\beta_i|\} = \delta = \delta_0$. Let

$$\delta_l = \min_{i \in I_l} \{|\beta_i|\}, \text{ for } 1 \leq l < k, \text{ and } \delta_k = 0.$$

The potential function Φ drops monotonically in each iteration in the phase. Within an iteration, we will account for the drop in two steps. First, as prices of goods in J are reduced, the β_i 's of buyers $i \in I$ increase, leading to a reduction in Φ . Second, when a newly active edge (j, i') is added to the network, the flow becomes more balanced, leading to a further drop. We will account for these two reductions separately, via different arguments (see Lemma 27).

For the first step, we work with l_1 norm, establishing an increase in $\sum_{i \in B_1} \beta_i$. In the second step, $\sum_{i \in B_1} \beta_i$ may not change at all, if $i' \in (B_1 - I)$. Instead, we establish a decrease in $\|\psi\|^2$, using an l_2 norm based argument. We observe that the latter argument is difficult to apply to the first step since the money of buyers change as prices change. Also, we do not know of a simple one step argument that accounts for the entire reduction in an iteration.

Next, we prove a key fact that accounts for the second decrease. Just before new edge (j, i') becomes active, let N be the network and \mathbf{p} be the prices of goods. Let N' be the network obtained by adding this edge to N ; of course, the prices remain unchanged. Let f and f^* be balanced flows in N and N' , respectively. For each buyer i , denote by ψ_i (ψ_i^*) the value of $\min(\beta_i, 0)$ w.r.t. flow f in N (flow f^* in N').

Lemma 24 *W.r.t. networks N and N' , if for some $i \in I$ and $\delta > 0$, $\psi_i^* = \psi_i + \delta < 0$, then $\|\psi\|^2 - \|\psi^*\|^2 \geq \delta^2$.*

Proof : Since the Invariant holds and the prices are unchanged, f and f^* have the same value. Therefore, flow $f^* - f$ will consist of circulations using the edge (j, i') . These circulations will have the effect of increasing the surplus of certain buyers in I and decreasing the surplus of others in $(B - I)$.

Among these circulations, consider only those that use the edge (t, i) . Add these circulations to f to get flow f' . Suppose that these circulations go from t to i to some i_l and back to t , for $1 \leq l \leq k$. For each buyer i , denote by ψ'_i the value of $\min(\beta_i, 0)$ w.r.t. flow f' in N' . We will show that $\|\psi\|^2 - \|\psi'\|^2 \geq \delta^2$, where $\psi'_i = \psi_i^* = \psi_i + \delta$. The effect of remaining circulations in $f^* - f$ will be similar, thereby yielding the required inequality.

For each buyer i_l , $1 \leq i_l \leq k$, there is a path from i to i_l in the corresponding circulation and hence there is a path from i_l to i in the residual graph w.r.t. flow f^* . Since f^* is balanced, by Property 1, the surplus of buyer i_l w.r.t. f^* is at least as large as that of i . By the observations made in the first paragraph of the proof, the surplus of i_l w.r.t. f^* is at most its surplus w.r.t. f' . Therefore, $\psi'_{i_l} \geq \psi_{i_l}$. Furthermore, we have $\psi'_{i_l} = \psi_{i_l} - \delta_l$, for some $\delta_l \geq 0$. These δ_l 's satisfy $\sum_{l=1}^k \delta_l \leq \delta$. (Observe that a δ_l may be 0; this happens if $\psi'_{i_l} = 0$, i.e., i_l is in B_2 in both N and N' . The strict inequality $\sum_{l=1}^k \delta_l < \delta$ will hold if there is a buyer i_l with $\beta_{i_l} > 0$.)

The inequality $\|\psi\|^2 - \|\psi'\|^2 \geq \delta^2$ now follows from Lemma 25. \square

Lemma 25 *Let $\delta, \delta_l \geq 0$, $l = 1, 2, \dots, k$, with $\delta \geq \sum_{l=1}^k \delta_l$. If $a + \delta \leq b_l - \delta_l \leq 0$, for $l = 1, 2, \dots, k$ then*

$$\|(a, b_1, b_2, \dots, b_k)\|^2 - \|(a + \delta, b_1 - \delta_1, b_2 - \delta_2, \dots, b_k - \delta_k)\|^2 \geq \delta^2.$$

Proof :

$$\left(a^2 + \sum_{l=1}^k b_l^2 \right) - \left((a + \delta)^2 + \sum_{l=1}^k (b_l - \delta_l)^2 \right)$$

$$\begin{aligned}
&\geq \left((a + \delta - \delta)^2 + \sum_{l=1}^k (b_l + \delta_l - \delta_l)^2 \right) - \left((a + \delta)^2 + \sum_{l=1}^k (b_l - \delta_l)^2 \right) \\
&\geq \delta^2 + 2(a + \delta) \left(\sum_{l=1}^k \delta_l - \delta \right) \geq \delta^2.
\end{aligned}$$

□

Let ψ^0 denote vector ψ at the start of the phase and ψ^l denote ψ at the end of iteration l , for $1 \leq l \leq k$.

Lemma 26 *In the l -th iteration, there is a buyer $i \in I_{l-1}$ such that $|\beta_i|$ decreases by at least $(\delta_{l-1} - \delta_l)$, for $1 \leq l < k$.*

Proof : By the definition of set I' in procedure **Update sets(I)** and Property 1, there is a buyer $i \in I_{l-1}$ which achieves $\min_{i \in I_l} \{|\beta_i|\}$ at the end of iteration l . Clearly, β_i increases (and hence $|\beta_i|$ decreases) by at least $(\delta_{l-1} - \delta_l)$ in the l -th iteration. □

Lemma 27 *For $1 \leq l \leq k$,*

$$\|\psi^{l-1}\|^2 - \|\psi^l\|^2 \geq (\delta_{l-1} - \delta_l)^2.$$

Proof : We first prove the statement for $1 \leq l < k$. By Lemma 26, there is a buyer $i \in I_{l-1}$ such that β_i increases by at least $(\delta_{l-1} - \delta_l)$ in the l -th iteration. Let us split this increase into two parts, the increase due to decrease in the prices of goods in J and that due to a new edge entering the network. Let these be δ' and δ'' , respectively. Therefore,

$$\delta' + \delta'' \geq \delta_{l-1} - \delta_l.$$

Let ψ' be the vector ψ just before the new edge is added to the network in iteration l , i.e., right after all the decrease in prices of J has happened. As prices in J decrease, the beta's of buyers in I increase, each leading to a decrease in $\|\psi'\|^2$; clearly, the beta's of buyers in $B - I$ remain unchanged. Let β_i be the value of beta of buyer i at the beginning of iteration l . Then,

$$\|\psi^{l-1}\|^2 - \|\psi'\|^2 \geq \beta_i^2 - (\beta_i + \delta')^2 \geq \delta'^2 - 2\beta_i\delta'.$$

By Lemma 24,

$$\|\psi'\|^2 - \|\psi^l\|^2 \geq \delta''^2.$$

Adding the two we get

$$\|\psi^{l-1}\|^2 - \|\psi^l\|^2 \geq \delta'^2 - 2\beta_i\delta' + \delta''^2 \geq (\delta' + \delta'')^2 \geq (\delta_{l-1} - \delta_l)^2,$$

where the second last inequality follows from the observation that $\delta'' \leq -\beta_i$.

Finally, in the k -th iteration, there is a buyer $i \in I_{k-1}$ whose ψ_i changes from $\beta_i < 0$ to 0. Therefore,

$$\|\psi^{k-1}\|^2 - \|\psi^k\|^2 \geq \beta_i^2 \geq (\delta_{k-1} - \delta_k)^2,$$

since $\delta_{k-1} \leq -\beta_i$ and $\delta_k = 0$. □

Lemma 28 *In a phase in Stage I, the potential drops by a factor of*

$$\left(1 - \frac{1}{n^2g}\right).$$

Proof : Now, $\|\psi^0\|^2 - \|\psi^k\|^2$ can be written as a telescoping sum of k terms, each of which is the decrease in the potential in one of the k iterations. Lemma 27 gives a lower bound on each of these terms. The total lower bound is minimized when each of the differences $(\delta_{l-1} - \delta_l)$ is equal. Now using the fact that $\delta_0 = \delta$ and $\delta_k = 0$, we get:

$$\|\psi^0\|^2 - \|\psi^k\|^2 \geq \frac{\delta^2}{k}.$$

Finally, since $\|\psi^0\|^2 \leq n\delta^2$, and by Lemma 23 $k \leq ng$, we get:

$$\|\psi^k\|^2 \leq \|\psi^0\|^2 \left(1 - \frac{1}{n^2g}\right).$$

□

Lemma 29 *At any point in Stage I, if $\sum_{i \in B_2} \beta_i > 0$, then*

$$\sum_{i \in B_2} \beta_i \geq \frac{1}{U^n \mu^g}.$$

Proof : First observe that the maximum bang-per-buck of buyers in B_2 remains unchanged throughout Stage I, and is determined by prices of goods found in the Initialization. Let $J_2 = \Gamma(B_2)$. By Property 1, there is no flow from a good in J_2 to a buyer in B_1 , and therefore, all flow from J_2 must go to buyers in B_2 . Therefore,

$$\sum_{i \in B_2} \beta_i = \sum_{i \in B_2} \theta_i - |B_2| = \sum_{i \in B_2} \alpha_i - \sum_{j \in J_2} p_j.$$

Now if $\sum_{i \in B_2} \beta_i > 0$, then the denominator of this sum is a product of at most n u_{ij} 's and at most g p_j 's, and is therefore bounded by $U^n \mu^g$, proving the lemma. □

Lemma 30 *The execution of Stage I requires at most*

$$O\left(n^4 g(n \log U + g \log \mu)\right)$$

max-flow computations.

Proof : By Lemma 28, the square of the potential drops by a factor of half after $O(n^2 g)$ phases. At the start of the algorithm, the potential is at most n .

If at any point, $\sum_{i \in B_2} \beta_i = 0$, i.e., $\forall i \in B_2 : \beta_i = 0$, each phase must in Step 5 and not Step 6, i.e., some buyers will be removed from consideration. Therefore, from this point on, at most n more phases are needed for Stage I to terminate.

Next, let's assume that $\sum_{i \in B_2} \beta_i > 0$ throughout Stage I. If so, by Lemma 29, once the potential drops below $\frac{1}{(U^n \mu^g)^2}$, the phase must end (in Step 7). Therefore the number of phases is

$$O(n^2 g \log(U^{2n} \mu^{2g})).$$

By Lemma 23 each phase consists of at most $n^2 g$ iterations and each iteration requires n max-flow computations for finding a balanced flow. The lemma follows. \square

13.2 Stage II

When $\forall i \in B : \beta_i < 0$, the algorithm starts with Stage II. Since in this stage the algorithm only raises prices of goods (i.e., increases x), by Lemma 12, $\forall i \in B : \beta_i < 0$ holds until termination.

In this section, we will work with the θ_i 's of buyers, rather than their β_i 's. Thus, throughout Stage II, $\forall i \in B : \theta_i < 1$. For Stage II, we will work with the following potential function:

$$\Phi = \sum_{i \in B} \theta_i^2.$$

Lemma 31 *In Stage II, at the termination of a phase, the prices of goods in the newly tight set must be rational numbers with denominator $\leq \Delta$.*

Proof : Let S be the newly tight set. Consider the subgraph of the network induced on the bipartition $(S, \Gamma(S))$, and view this as an undirected graph, say H . Assume w.l.o.g. that this graph is connected (otherwise we prove the lemma for each connected component of H). Pick a spanning tree in H .

Pick any good $j \in S$, and find a path in the spanning tree from j to each good $j' \in S$. If j reaches j' with a path of length $2l$, then $p_{j'} = ap_j/b$ where a and b are products of l utility parameters (u_{ik} 's) each. Since alternate edges of this path contribute to a and b , we can partition the u_{ik} 's of edges in the spanning tree into two sets, T_1 and T_2 , such that a uses u_{ik} 's from T_1 and b uses them from T_2 .

Next, consider $\alpha_i = c_i/\gamma_i$, for $i \in \Gamma(S)$. Now, $\gamma_i = u_{i,j'}/p_{j'}$, where (i, j') is any edge in the network. Find the path in the spanning tree from i to j and use the first edge on this path for computing γ_i (it is easy to see that all these edges come from set T_2), and substitute $p_{j'}$ using the expression stated above, i.e., $p_{j'} = ap_j/b$.

Since S is a tight set,

$$\sum_{j' \in S} p_{j'} = \sum_{i \in \Gamma(S)} 1 + \alpha_i.$$

In this equation, substitute for $p_{j'}$ and α_i using the expressions constructed above to get an equation with one variable, i.e., p_j . Now, it is easy to see that the denominator of p_j is $\leq \Delta$. \square

Lemma 32 *In Stage II, consider two phases P and P' , not necessarily consecutive, such that good j lies in the newly tight sets at the end of P as well as P' . Then the increase in the price of j , going from P to P' , is $\geq 1/\Delta^2$.*

Proof : Let the prices of j at the end of P and P' be p/q and r/s , respectively. Clearly, $r/s > p/q$. By Lemma 31, $q \leq \Delta$ and $r \leq \Delta$. Therefore the increase in price of j ,

$$\frac{r}{s} - \frac{p}{q} \geq \frac{1}{\Delta^2}.$$

\square

Lemma 33 *In Stage II, a phase consists of at most g iterations.*

Proof : After each iteration, other than the last one, at least one good will move from $G - J$ to J . \square

The structure of the rest of the argument is quite similar to that of Stage I. Once again, the central fact established is that Φ drops by a substantial factor, of $(1 - 1/n^2)$ in a phase (Lemma 38). Once again, assume that a given phase consists of k iterations. Let I_0 denote set I at the start of the phase and let I_l denote the set I at the end of the l -th iteration, $1 \leq l \leq k$. Assume that at the start of this phase, $\max_{i \in B} \{\theta_i\} = \delta = \delta_0$. Let

$$\delta_l = \min_{i \in I_l} \{\theta_i\}, \text{ for } 1 \leq l < k, \text{ and } \theta_k = 0.$$

As in Stage I, we will account for the drop in Φ in two steps in each iteration. First, as prices of goods in J are increased, the θ_i 's of buyers $i \in I$ decrease, leading to a reduction in Φ . Second, when a newly active edge (j, i) is added to the network, the flow becomes more balanced, leading to a further drop. As in Stage I, we will account for the first drop using l_1 norm and the second drop using l_2 norm.

We first account for the second decrease. Just before new edge (j, i) becomes active, let N be the network and \mathbf{p} be the prices of goods. Let N' be the network obtained by adding this edge to N ; of course, the prices remain unchanged. Let f and f^* be balanced flows in N and N' , respectively. Denote by θ (θ^*) the surplus vector w.r.t. flow f in N (flow f^* in N').

Lemma 34 *W.r.t. networks N and N' , if for some $i \in I$ and $\delta > 0$, $\theta_i^* = \theta_i - \delta \geq 0$, then $\|\theta\|^2 - \|\theta^*\|^2 \geq \delta^2$.*

Proof : Since the Invariant holds and the prices are unchanged, f and f^* have the same value. Therefore, flow $f^* - f$ will consist of circulations using the edge (j, i) . These circulations will have the effect of decreasing the surplus of certain buyers in I and increasing the surplus of certain buyers in $(B - I)$.

Among these circulations, consider only those that use the edge (i, t) . Add these circulations to f to get flow f' . Suppose that these circulations go from i to t to some i_l and back to i , for $1 \leq l \leq k$. Denote by θ' the surplus vector w.r.t. flow f' in N' . We will show that $\|\theta\|^2 - \|\theta'\|^2 \geq \delta^2$. The effect of remaining circulations in $f^* - f$ will be similar, thereby yielding the required inequality.

For each buyer i_l , $1 \leq i_l \leq k$, there is a path from i_l to i in the corresponding circulation and hence there is a path from i to i_l in the residual graph w.r.t. flow f^* . Since f^* is balanced, by Property 1, the surplus of buyer i w.r.t. f^* is at least as large as that of i_l . Clearly, the surplus of i_l w.r.t. f^* is at least its surplus w.r.t. f' . Therefore, $\theta'_{i_l} \geq \theta'_{i_l}$. Furthermore, we have $\theta'_{i_l} = \theta_{i_l} + \delta_l$, for some $\delta_l > 0$. These δ_l 's satisfy $\sum_{l=1}^k \delta_l = \delta$.

The inequality $\|\theta\|^2 - \|\theta'\|^2 \geq \delta^2$ now follows from Lemma 35. □

Lemma 35 *If $a \geq b_i \geq 0, i = 1, 2, \dots, k$ and $\delta \geq \sum_{j=1}^k \delta_j$ where $\delta, \delta_j \geq 0, j = 1, 2, \dots, k$, then*

$$\|(a + \delta, b_1 - \delta_1, b_2 - \delta_2, \dots, b_k - \delta_k)\|^2 - \|(a, b_1, b_2, \dots, b_k)\|^2 \geq \delta^2.$$

Proof :

$$(a + \delta)^2 + \sum_{i=1}^k (b_i - \delta_i)^2 - a^2 - \sum_{i=1}^k b_i^2 \geq \delta^2 + 2a(\delta - \sum_{i=1}^k \delta_i) \geq \delta^2.$$

□

Let θ^0 denote the surplus vector at the start of the phase and let θ^l denote the surplus vector at the end of iteration l , for $1 \leq l \leq k$.

Lemma 36 *In the l -th iteration, there is a buyer $i \in I_{l-1}$ whose surplus decreases by at least $(\delta_{l-1} - \delta_l)$, for $1 \leq l < k$.*

Proof : By the definition of set I' in procedure **Update sets(II)** and Property 1, there is a buyer $i \in I_{l-1}$ which achieves $\min_{i \in I_l} \{\theta_i\}$ at the end of iteration l . Clearly, the surplus of i decreases by at least $(\delta_{l-1} - \delta_l)$ in the l -th iteration. □

Lemma 37 For $1 \leq l \leq k$,

$$\|\theta^{l-1}\|^2 - \|\theta^l\|^2 \geq (\delta_{l-1} - \delta_l)^2.$$

Proof : We first prove the statement for $1 \leq l < k$. By Lemma 36, there is a buyer $i \in I_{l-1}$ whose surplus decreases by at least $(\delta_{l-1} - \delta_l)$ in the l -th iteration. Let us split this decrease into two parts, the increase due to increase in the prices of goods in J and that due to a new edge entering the network. Let these be δ' and δ'' , respectively. Therefore,

$$\delta' + \delta'' \geq \delta_{l-1} - \delta_l.$$

Let θ' be the surplus vector just before the new edge is added to the network in iteration l , i.e., right after all the increase in prices of J has happened. As prices in J increase, the surpluses of buyers in I decrease, but those of buyers in $B - I$ remain unchanged. Let θ_i be the surplus of buyer i at the beginning of iteration l . Then,

$$\|\theta^{l-1}\|^2 - \|\theta^l\|^2 \geq \theta_i^2 - (\theta_i - \delta')^2 \geq \delta'^2 + 2\theta_i\delta'.$$

By Lemma 34,

$$\|\theta^l\|^2 - \|\theta^l\|^2 \geq \delta''^2.$$

Adding the two we get

$$\|\theta^{l-1}\|^2 - \|\theta^l\|^2 \geq \delta'^2 + 2\theta_i\delta' + \delta''^2 \geq (\delta' + \delta'')^2 \geq (\delta_{l-1} - \delta_l)^2,$$

where the second last inequality follows from the observation that $\delta'' \leq \theta_i$.

Finally, in the k -th iteration, there is a buyer $i \in I_{k-1}$ whose surplus changes from $\theta_i > 0$ to 0. Therefore,

$$\|\theta^{k-1}\|^2 - \|\theta^k\|^2 \geq \theta_i^2 \geq (\delta_{k-1} - \delta_k)^2,$$

since $\delta_{k-1} \leq \theta_i$ and $\delta_k = 0$. □

Lemma 38 In a phase in Stage II, the potential drops by a factor of

$$\left(1 - \frac{1}{n^2}\right).$$

Proof : Now, $\|\theta^0\|^2 - \|\theta^k\|^2$ can be written as a telescoping sum of k terms, each of which is the decrease in the potential in one of the k iterations. Lemma 37 gives a lower bound on each of these terms. The total lower bound is minimized when each of the differences $(\delta_{l-1} - \delta_l)$ is equal. Now using the fact that $\delta_0 = \delta$ and $\delta_k = 0$, we get:

$$\|\theta^0\|^2 - \|\theta^k\|^2 \geq \frac{\delta^2}{k}.$$

Finally, since $\|\theta^0\|^2 \leq n\delta^2$, and by Lemma 33 $k \leq n$, we get:

$$\|\theta^k\|^2 \leq \|\theta^0\|^2 \left(1 - \frac{1}{n^2}\right).$$

□

Lemma 39 *The execution of Stage II requires at most*

$$O\left(n^4(\log n + n \log U + \log C)\right)$$

max-flow computations.

Proof : By Lemma 38, the potential drops by a factor of half after $O(n^2)$ phases. At the start of the algorithm, the potential is at most n . Once its value drops below $1/\Delta^4$, the algorithm requires at most n more phase to compute equilibrium prices. This follows from Lemma 31 and Lemma 32. Therefore the number of phases is

$$O(n^2 \log(\Delta^4 n)) = O(n^2(\log n + n \log U + \log C)).$$

By Lemma 33 each phase consists of n iterations and each iteration requires n max-flow computations for computing a balanced flow. The lemma follows. □

Lemmas 30 and 39 give:

Theorem 40 *Algorithms 18 and 19 solve the decision and search versions of Nash bargaining game **ADNB** using*

$$O\left(n^4 g(\log n + n \log U + \log C + g \log \mu)\right)$$

max-flow computations.

14 l_1 -norm Does Not Suffice

We give a family of examples showing that the DPSV algorithm, for Fisher's linear case, may end up making only inverse exponential progress in a phase if the potential function used is the l_1 norm of the surplus vector.

We will define the example in terms of 2 parameters, δ and H , which will be set at the end. Assume $B = \{b_0, b_1, \dots, b_{n-1}, b_n\}$ and $G = \{g_0, g_1, \dots, g_{n-1}, g_n\}$. At the start of the phase, the only edges present in the network are (g_i, b_i) , for $0 \leq i \leq n$. The money of the buyers are as follows:

$$m_0 = 1 + \delta, \quad \text{and for } 1 \leq i \leq n-1, \quad m_i = \frac{\delta}{2^i}, \quad \text{and } m_n = H + \frac{\delta}{n}.$$

The prices of goods are as follows:

$$p_0 = 1, \quad \text{and for } 1 \leq i \leq n-1, \quad p_i = \frac{\delta}{2^i}, \quad \text{and } p_n = H + \frac{\delta}{n}.$$

Hence, at the start of the phase, the surplus of b_0 is δ , and that of the rest of the buyers is 0.

We will set $\delta = 1$ and H to be a large number, say n . The phase starts with $I = \{b_0\}$ and $J = \{g_0\}$. Assume that at the end of iteration i , edge (g_i, b_{i-1}) enters the network, and as a result, b_i enters I and g_i enters J , for $1 \leq i \leq n$. The increment in price in each iteration is very small – this is easily arranged by choosing the right utilities u_{ij} 's.

To keep the description clean, let us assume the increments in price are all zero; the numbers can be easily modified by inverse exponential amounts to yield the desired outcome, even if the prices need to increase in each iteration. If so, at the end of all this, the surplus of b_i is

$$\frac{\delta}{2^{i+1}}, \quad \text{for } 0 \leq i \leq n-1,$$

and that of b_n is $\frac{\delta}{2^{n-1}}$.

Finally, in iteration $n+1$, a very slight increase in x leads to set $\{g_n\}$ going tight. Observe that the reason for choosing H to be a large number is to ensure that this slight increase in x will not make a larger set go tight. Observe that $\Gamma(\{g_n\} = \{b_n, b_{n+1}\})$. Now, the increase in the price of g_n needed for this is $\frac{\delta}{2^{n-1}}$. Since H is a large number and the increase in x is very small, the total increase in the prices of other goods is at most a constant factor more.

In summary, the total increase in the l_1 norm of \mathbf{p} in this phase is an inverse exponential factor.

15 Subclasses of LNB: UNB and SNB

SNB is a subclass of UNB, which in turn is a subclass of LNB. We first define UNB. Let \mathcal{G} be a game in LNB whose feasible set of utilities, \mathcal{N} , is described by packing constraints, i.e., linear constraints of the form \leq with all coefficients and rhs being nonnegative and all variables being constrained to be nonnegative. Further, assume they are written in the form

$$\forall j : \sum_{i \in B} a_{ij} v_i \leq d_j,$$

where v_i is buyer i 's utility, index j varies over constraints, $d_j \geq 0$, and each v_i is constrained to be nonnegative.

It will be useful to view each constraint j as defining an abstract good which is desired by only the buyers having non-zero coefficients in this constraint. Thus each b_j specifies the supply of resource j , and for a fixed j the different a_{ij} 's encode the different ways in which agents use this resource.

In case the given game was in the class RNB, the constraints representing \mathcal{N} can be obtained from the constraints for the original game using the Fourier-Motzkin elimination method (see

[Sch86]), and hence, the constraints describing \mathcal{N} are linear combinations of the original constraints. Clearly, the dual variables corresponding to the constraints of \mathcal{N} , which are prices of the abstract goods, are the same linear combination of the prices (dual variables of the original inequalities) of the concrete goods. Hence, we can transform prices of the abstract goods into prices of the concrete goods and vice versa.

Consider the special case in which for each j , all agents interested in this resource use it in the same way, i.e., all a_{ij} 's are restricted to be 0/1. Clearly, there can be at most 2^n such relevant constraints. The b_j 's for these constraints can be encoded via a function $z : 2^B \rightarrow \mathbf{R}_+$ and the constraints can then be written as:

$$\forall S \subseteq B : \sum_{i \in S} v_i \leq z(S)$$

$$\forall i \in B : v_i \geq 0$$

We will impose the additional condition on function z that it should satisfy the *covering property*: Assume that the cost of any set $S \subseteq B$ is given by $z(S)$. Then, for any $S \subseteq B$, $z(S)$ must be bounded by the cost of any fractional covering of S . Clearly, such a function satisfies *monotonicity*, i.e., if $S \subseteq T \subseteq B$ then $z(S) \leq z(T)$, and the fact that $z(\emptyset) = 0$.

The subclass of LNB defined via such functions z will be called *uniform utility Nash and nonsymmetric bargaining games*, and abbreviated UNB. The solution to such a bargaining game is given by the following convex program.

$$\begin{aligned} & \text{maximize} && \sum_{i \in B} w_i \log(v_i - c_i) && (10) \\ & \text{subject to} && \forall S \subseteq B : \sum_{i \in S} v_i \leq z(S) \\ & && \forall i \in B : v_i \geq 0 \end{aligned}$$

An important special case arises when z is a *polymatroid function*, i.e., it is submodular⁴, monotone, and $z(\emptyset) = 0$. The subclass of UNB containing such bargaining games will be called *submodular utility Nash and nonsymmetric bargaining games*, and abbreviated SNB.

Next we show how to transform a game \mathcal{G} in UNB to a flexible budget market \mathcal{M} . The set of buyers in \mathcal{M} is B and the set of goods corresponds to the power set of B . For $S \subseteq B$, the amount of good S available is $z(S)$. Agent $i \in B$ gets utility v_i only if she obtains v_i units of good S for each set S containing i . Let $p(S)$ denote the price of one unit of good S . Then, the cost to buyer i for one unit of utility is

$$\text{cost}(i) = \sum_{S: i \in S} p(S).$$

Therefore her bang-per-buck, $\gamma_i = \frac{1}{\text{cost}(i)}$. Her money in \mathcal{M} is defined to be $w_i + \frac{c_i}{\gamma_i} = w_i + c_i \text{cost}(i)$. This completes the definition of \mathcal{M} . By a proof similar to that of Theorem 4 it

⁴Function z is said to be *submodular* if $z(S) + z(T) \geq z(S \cap T) + z(S \cup T)$, for any sets $S, T \subseteq B$.

is easy to see that:

Theorem 41 *The game \mathcal{G} is feasible iff market \mathcal{M} is. Moreover, if both are feasible, \mathbf{v} and \mathbf{p} are optimal utilities and dual for I iff they are equilibrium utilities and prices for market \mathcal{M} .*

16 Algorithms for Games in SNB

We will show that there is a combinatorial, strongly polynomial algorithm for solving each game in SNB; as a corollary, all such games are rational. Our result and algorithm generalizes that of [JV08], giving a combinatorial, strongly polynomial algorithm for finding the equilibrium for any Eisenberg-Gale market in the class SUA (submodular utility allocation markets). In the interest of brevity, we will only present the new ideas involved; the reader is advised to look at [JV08] first.

We begin by presenting an algorithm for a concrete game in this class – this algorithm will contain all the key ideas for developing the general algorithm. This game is **RNB1** for the special case that there is single source, though multiple sinks.

Let us set up notation for this game. We are given a directed graph $G = (V, E)$, with $c_e \in \mathbf{Q}^+$ specifying the capacity of edge $e \in E$. $s \in V$ is the source node, and $T = \{t_1, \dots, t_k\} \subset V$ is the set of sinks. Each sink also represents a player the game and has its own disagreement utility (flow value) c_i , for $1 \leq i \leq k$. In the nonsymmetric version, we are also given the clouts w_i of each player. The object is to find the Nash or nonsymmetric bargaining solution. Let \mathcal{G} denote the given instance of this game.

Define function $z : 2^T \rightarrow \mathbf{Q}^+$ as follows: for $S \subseteq T$, $z(S)$ is the maximum total flow possible from s to sinks in S . In a classic paper, dealing with fair allocation of flows in single-source multiple-sink networks, Megiddo [Meg74] shows that function z is submodular. As a consequence, this game is in SNB.

Reduce game \mathcal{G} to a flexible budget market, say \mathcal{M} , as per Theorem 41. This market generalizes the single-source multiple-sink Eisenberg-Gale market studied in [JV08]. We indicate how to generalize the algorithm of [JV08] to this setting; below, we will follow the exposition given in Section 5.14 in [Vaz07].

The algorithm iteratively assigns prices to cuts in G . At any point in the algorithm, for $t_i \in T$, define $\text{cost}(i)$ to be the cost of the cheapest path from s to t_i in G . As stated in Theorem 41, the money of player i is defined to be $m_i = w_i + c_i \text{cost}(i)$. The flow demanded by t_i at any point in the algorithm is

$$f_i = \frac{m_i}{\text{cost}(i)} = \frac{w_i}{\text{cost}(i)} + c_i.$$

For $S \subseteq V$, define $d(S)$ to be the sum of disagreement utilities of all sinks in S .

The algorithm starts by finding a maximal min-cut separating T from s , say (S, \bar{S}) . If the capacity of this cut is bounded by $d(\bar{S})$, market \mathcal{M} is infeasible. Otherwise, this cut is currently over saturated (since all sinks are demanding infinite flow). The algorithm raises the prices of all edges in this cut until it is deemed to be saturated. The latter condition happens when

there is a cut, say (U, \bar{U}) , with $S \subset U$, such that the difference in the capacities of the two cuts is precisely equal to the flow demanded by sinks in $(\bar{S} - \bar{U})$. Let (U, \bar{U}) be the maximal such cut (it will be unique). If the capacity of this cut is bounded by $d(\bar{U})$, market \mathcal{M} is infeasible. If $U = V$, the algorithm halts. Otherwise, cut (U, \bar{U}) must be over-saturated – it assumes the role of (S, \bar{S}) and the algorithm goes to the next iteration.

On termination, the price of each edge will be the sum of prices of all cuts it is in. Observe that the algorithm raises the prices of a nested sequence of cuts. Using standard uncrossing arguments, that apply because of submodularity, one can prove that this is true for any game in SNB. It is easy to see that the flows and edge prices satisfy all KKT conditions for program (10).

The only remaining question is how to find the next cut (U, \bar{U}) . This requires a very small modification to the algorithm given in Section 5.14.1 in [Vaz07] for the analogous question for the single-source multiple-sink Eisenberg-Gale market. The modification is that the capacity of edge (t_i, t) in graph G' needs to be defined to be

$$\frac{w_i}{(p' + p)} + c_i.$$

Observe that the algorithm sketched above involved only 2 changes from the algorithm of [JV08] – checking for feasibility in each iteration and the modified definition of the capacity of edge (t_i, t) in graph G' . These are precisely the 2 changes that need to be made to the algorithm given in Section 7 in [JV08] for an SUA market in order to extend it to an algorithm for an arbitrary game in SNB.

Theorem 42 *There is a combinatorial, strongly polynomial algorithm for each game in SNB. Moreover, all such games are rational.*

It is easy to see that a function $z : 2^B \rightarrow \mathbf{Q}^+$ which satisfies the covering property and $z(\emptyset) = 0$ is submodular if $|B| = 2$. As a result, for the case of 2 players, the games **RNB2** and **RNB3** are in SNB and hence solvable in strongly polynomial time. For the case of 3 or more players, both these games may have only irrational solutions, as shown for the case of Eisenberg-Gale markets in [JV08].

17 Discussion

17.1 What is a Combinatorial Algorithm?

Giving a formal definition of a “combinatorial algorithm” is as futile as giving a formal definition of a greedy algorithm, or for that matter, of computability itself – no matter what definition one gives, it is easy, via some contortions, to include in it algorithms that are obviously not “combinatorial”. Therefore, rather than “define” such algorithms, we need to come to an agreement with the reader that the proposed framework captures the main essence of what is intended.

Let us say that a *combinatorial algorithm* is one that searches over a discrete space (typically exponential in the size of the given instance), which contains the answer as one of its elements. The algorithm may need to solve other problems in order to move from point to point in this space; however, each of those computations also need to be combinatorial in the same sense, i.e., a search over a discrete space.

Consider the linear programming problem. Since an optimal solution to this problem is achieved at a vertex of the given polytope, the simplex algorithm, which searches over the set of vertices of the given polytope, is a combinatorial algorithm, although we do not know of a polynomial time implementation of this idea yet. On the other hand, the ellipsoid algorithm and interior point methods, although polynomial time, are not combinatorial.

The vulnerability of the “definition” given above is easily illustrated. By artificially redefining the linear programming problem – by rounding all numbers to a chosen accuracy – an interior point algorithm can be made to fit the framework given above. However, observe that the discrete space, which is simply a grid, is quite inane for the problem at hand.

17.2 Open problems and issues

Two obvious ways to extend this work is to develop combinatorial algorithms for the nonsymmetric extension of **ADNB** and to go beyond linear utility functions; in particular, consider **ADNB** with piecewise-linear and concave utilities. The nonsymmetric extension of **ADNB** is particularly interesting since its convex program generalizes the EG-program and hence captures Fisher’s linear case as well. Our algorithm for **ADNB**, just like the DPSV algorithm, is polynomial time but not strongly polynomial. In both cases, obtaining strongly polynomial extensions appear to be interesting and difficult questions.

As in the case of the EG-program, the optimal solutions of convex program (5) resemble those of a linear program rather than a nonlinear program. So, we repeat a question raised in [Vaz07] namely, can the solution to **ADNB** be captured via a linear program? We believe the answers to these questions are “no” and that establishing this in a suitable formal framework will provide new insights into the boundary between linear and nonlinear programs.

The reader can see that the algorithm for **ADNB** exploits a surprisingly rich and clean structure which is, in some ways, reminiscent of the magestic structure of matching. In our experience, such structure does not occur in isolation and we believe that what we see so far is the tip of an iceberg. This leads to the question: what does the rest of the iceberg look like?

One possibility is to seek combinatorial approximation algorithms for solving specific classes of nonlinear convex programs. In this respect, important hints may be obtained from the way the primal-dual paradigm was extended from solving linear programs exactly to obtaining near-optimal solutions to linear programs within the area of approximation algorithms. The mechanism involved in all of the latter algorithms was that of relaxing complementary slackness conditions, which was first formalized in [WGMV93]. We also note that in the setting of approximation algorithms, the primal-dual paradigm has been successful primarily for minimization problems. So, our more precise question is, “Is there a natural way of relaxing KKT conditions to obtain primal-dual (combinatorial) algorithms for near-optimally solving interesting classes of (perhaps minimization) nonlinear convex programs?”

18 Acknowledgements

I am indebted to Ehud Kalai and Nimrod Megiddo for sharing their insights on the Nash bargaining game and starting me off on this research.

References

- [And09] M. Andrews. Personal communication. 2009.
- [AQS05] M. Andrews, L. Qian, and A. Stolyar. Optimal utility based multi-user throughput allocation subject to throughput constraints. In *INFOCOM*, 2005.
- [BS00] W. C. Brainard and H. E. Scarf. How to compute equilibrium prices in 1891. *Cowles Foundation Discussion Paper*, (1270), 2000.
- [CGV⁺09] D. Chakrabarty, G. Goel, V. V. Vazirani, L. Wang, and C. Yu. Submodularity helps in Nash and nonsymmetric bargaining games. Submitted to *Games and Economic Behavior*, 2009.
- [DPSV08] N. Devanur, C. H. Papadimitriou, A. Saberi, and V. V. Vazirani. Market equilibrium via a primal-dual-type algorithm. *JACM*, 55(5), 2008.
- [Edm65] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards. Section B*, 69:125–130, 1965.
- [EG59] E. Eisenberg and D. Gale. Consensus of subjective probabilities: the Pari-Mutuel method. *The Annals of Mathematical Statistics*, 30:165–168, 1959.
- [GLS88] M. Grotschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1988.
- [GV09] G. Goel and V. V. Vazirani. A perfect price discrimination market, its welfare theorems, and an efficient algorithm for computing its equilibria. Submitted, 2009.
- [JV08] K. Jain and V. V. Vazirani. Eisenberg-Gale markets: Algorithms and game-theoretic properties. *Games and Economic Behavior (to appear)*, 2008.
- [Kal77] E. Kalai. Nonsymmetric Nash solutions and replications of 2-person bargaining. *International Journal of Game Theory*, 6(3):129–133, 1977.
- [Kal85] E. Kalai. Solutions to the bargaining problem. In L. Hurwicz, D. Schmeidler, and H. Sonnenschein, editors, *Social Goals and Social Organization*, pages 75–105. Cambridge University Press, 1985.
- [Kel97] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997.

- [KV] F. P. Kelly and V. V. Vazirani. Rate control as a market equilibrium. Unpublished manuscript 2002. Available at: <http://www.cc.gatech.edu/~vazirani/KV.pdf>.
- [Meg74] N. Megiddo. Optimal flows in networks with multiple sources and sinks. *Mathematical Programming*, 7:97–107, 1974.
- [MV07] N. Megiddo and V. V. Vazirani. Continuity properties of equilibrium prices and allocations in linear Fisher markets. In *Proceedings of The 3rd Workshop on Internet and Network Economics*, 2007.
- [Nas50] J. F. Nash. The bargaining problem. *Econometrica*, 18:155–162, 1950.
- [Nis09] N. Nisan. Report from ALGO 2009. In *Algorithmic Game Theory Blog*, September 10, 2009.
- [NRTV07] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007.
- [OR94] M. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press, 1994.
- [Sch86] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, NY, 1986.
- [Sch03] A. Schrijver. *Combinatorial Optimization*. Springer-Verlag, 2003.
- [TL89] W. Thomson and T. Lensberg. *Axiomatic Theory of Bargaining With a Variable Population*. Cambridge University Press, 1989.
- [Vaz06] V. V. Vazirani. Spending constraint utilities, with applications to the Adwords market. Submitted, 2006.
- [Vaz07] V. V. Vazirani. Combinatorial algorithms for market equilibria. In N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, editors, *Algorithmic Game Theory*, pages 103–134. Cambridge University Press, 2007.
- [Vaz09] V. V. Vazirani. 2-player Nash and nonsymmetric bargaining games: algorithms and structural properties. Submitted to *Games and Economic Behavior*, 2009.
- [VW09] V. V. Vazirani and L. Wang. Continuity properties of equilibria in some Fisher and Arrow-Debreu market models. In *Proceedings of The 5th Workshop on Internet and Network Economics*, 2009.
- [VY10] V. V. Vazirani and M. Yannakakis. Market equilibria under separable, piecewise-linear, concave utilities. In *Proceedings of The First Symposium on Innovations in Computer Science*, 2010.
- [WGMV93] D.P. Williamson, M.X. Goemans, M. Mihail, and V.V. Vazirani. A primal-dual approximation algorithm for generalized Steiner network problems. In *Proceedings, 25th Annual ACM Symposium on Theory of Computing*, 1993.