Network Bargaining: Creating Stability Using Blocking Sets

by

David Steiner

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2012

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

David Steiner

Abstract

Bargaining theory seeks to answer the question of how to divide a jointly generated surplus between multiple agents. John Nash proposed the Nash Bargaining Solution to answer this question for the special case of two agents. Kleinberg and Tardos extended this idea to network games, and introduced a model they call the Bargaining Game. They search for surplus divisions with a notion of fairness, defined as balanced solutions, that follow the Nash Bargaining Solution for all contracting agents. Unfortunately, many networks exist where no balanced solution can be found, which we call unstable. In this thesis, we explore methods of changing unstable network structures to find fair bargaining solutions. We define the concept of Blocking Sets, introduced by Biro, Kern and Paulusma, and use them to create stability. We show that by removing a blocking set from an unstable network, we can find a balanced bargaining division in polynomial time. This motivates the search for minimal blocking sets. Unfortunately this problem is NP-hard, and hence no known efficient algorithm exists for solving it. To overcome this hardness, we consider the problem when restricted to special graph classes. We introduce a $O(1)$-factor approximation algorithm for the problem on planar graphs with unit edge weights. We then provide an algorithm to solve the problem optimally in graphs of bounded treewidth, which generalize trees.

Acknowledgements

First, I would like to offer a deep and sincere thank you to my supervisors: Prof. Jochen Könemann for being ever-enthusiastic and supportive in my work, and Prof. Kate Larson for her direction and suggestions. They introduced me to a wide range of interests, and encouraged me to push my boundaries whenever possible. This research would have been impossible without their helps and guidance.

Many thanks go to Prof. Alejandro López-Ortiz and Prof. Chaitanya Swamy, the readers of this thesis, for their thoughtful comments. Also, I would like to thank the staff, faculty, and students of the School of Computer Science at the University of Waterloo for providing me a home for the past seven years.

I would like to thank my friends here at the University of Waterloo, especially the members and coaches of the volleyball programs, who played a very important role in making my time more enjoyable, and offered their daily support. I am fortunate to have such excellent companies.

Last but not certainly not least, my incredible family, who was always at my side, receive my deepest love for their sweetest care and never-ending encouragement. I wish to thank my beautiful sister-in-law Cait, my wise grandmother Marta, my loving parents Judit and George, and my ever-supportive brother Adam. You are my foundation. This work is dedicated to all of you.

Contents

List of Figures

Chapter 1

Introduction

The study of *bargaining* has a long history in many disciplines such as economics and sociology. Bargaining theory asks the question of how to divide a jointly generated surplus between multiple agents. In the 1950s, John Nash [24] proposed a solution to this question for the special case of two agents. His *Nash Bargaining Solution* is a celebrated result still used today in traditional economic research. Sociological study has focused on finding experimental evidence for the way in which people behave in bargaining situations. On the other hand, mathematical research has concentrated on developing models that are both mathematically tractable, and agree with the results of the sociological experiments. More recently, bargaining theory has been studied in algorithmic game theory settings. In this thesis, we direct our focus to solutions based on Nash's ideas for bargaining games with more than two agents. We are particularly interested in efficient algorithms for finding these solutions.

Consider a situation where two agents are negotiating how to divide one unit of money. Each agent is self-interested with a simple monotonically increasing linear utility function. Additionally, each agent has an *alternate* option, an amount that it can collect in case the negotiations fail and no division can be agreed upon. Nash's Bargaining Solution [24] predicts that the two agents in such a case will agree upon money shares that evenly split the value in excess of their combined alternate options.

Recent work by Kleinberg and Tardos [18] extended Nash's ideas to the network setting. They introduced a model for studying bilateral negotiations between multiple agents called the *Bargaining Game*. Given a graph, the model positions a player at every vertex, and assigns a money value to every edge of the graph. Two players joined by an edge may enter into a contract and split the value on the edge, but only if they can agree on a division. Each player is allowed to contract with at most one neighbour.

Network exchange theory focuses on identifying and analyzing structural conditions of bargaining power in networks. Configurations or positions, resources, and connections determine the distribution of strength across the participating players. High-power positions gain more favourable contract ratios. Kleinberg and Tardos [18] define two solution concepts to help identify the factors influencing bargaining power. Let x_u and x_v be the corresponding shares for players u and v of a negotiated contract with dollar value w_{uv}. Let α_u and α_v be the respective alternatives of the agents. *Stable outcomes* (Chapter 3) are bargaining solutions with the property that $x_u \geq \alpha_u$ for every player $u \in V$, and so no agent has incentive to change contracts. A solution is a *Balanced outcome* (Chapter 3) if in addition to being a stable outcome, for every contracting pair of players (u, v), the division is according to Nash's solution. More formally, that is if $x_u + x_v = w_{uv}$ and $(x_u - \alpha_u) = (x_v - \alpha_v)$. Kleinberg and Tardos [18] argue that any stable or balanced outcome will divide a total surplus of $\nu(G)$, the size of a maximum weight matching in the underlying graph. They also give a polynomial time algorithm to compute the set of balanced outcomes, proving many structural properties in the process.

Kearns and Chakraborty [8], expand on the model introduced by Kleinberg and Tardos. They allow players to make an arbitrary number of deals, and each agent to have its own unique utility function. By using utility functions that satisfy certain natural conditions such as being continuous and concave, they prove that for general networks there exist states of equilibrium. These are bargaining divisions where all the edges are balanced for various solution concepts, including the Nash Bargaining Solution. In later work, Chakraborty, Kearns and Khanna [9] show that these equilibrium states are not unique in general graphs, consistent with the findings of Kleinberg and Tardos on balanced solutions in their model.

A different approach to the bargaining game is with cooperative game theory, a field which studies the problem of *fair division*. A *cooperative game* is a pair (v, N) where N is a finite set of players, and $v : 2^N \to \mathbb{R}$. Various solution concepts are defined for cooperative games, each searching for some notion of fairness when dividing $v(N)$ among the players. The bargaining game we described can then be considered a transferable-utility cooperative game. N is the vertex set of the underlying graph, and for all subsets $S \subseteq N$, $v(S) = \nu(S)$, which is the size of the maximum weight matching in $G[S]$, the subgraph of G induced by S. This form of the game was first introduced by Shapley and Shubik [30], and is known as the *matching game*. The *core* and *prekernel* (formal definitions in Chapter 2) are two solution concepts for cooperative games. For the matching game, it has been shown that the core is exactly the set of stable outcomes [30], and the intersection of the core and prekernel is the set of balanced outcomes (as was observed subsequently and independently by Bateni et al. [4]).

Unfortunately, the core of a matching game is empty for a large sets of graphs. Using structural properties of networks we can identify these unstable graphs, those without a stable outcome, where negotiations will break down as agents will not agree on fair divisions.

Consider the simple situation with three players whose connections form a triangle, as shown in Figure 1.1. Each contract is worth one dollar. As negotiations take place, there will always be one agent who will be left out and receive nothing (blue vertices). This agent will always be able to offer one of his neighbours more than their existing contract (red edges), forming a beneficial switch. The negotiators will never be able to agree, as there will be a continuous spiral of improving offers, causing disorder. Our main work focuses on establishing stability for networks in anarchy. We explore various methods of changing the composition of underlying graphs to re-create order.

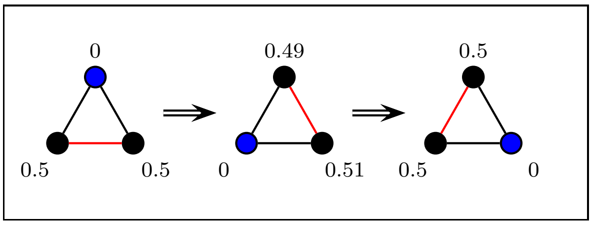

Figure 1.1: An Unstable Graph

A common approach for changing network structure to establish desirable properties is edge deletion [23, 34]. More formally, that is finding a minimal subset of edges $E^- \subset E$ to remove, such that the graph $G^- = (V,\ E \setminus E^-)$ has a property $\mathcal{P}$. Our work is based on this idea. We define the concept of Blocking Sets (Chapter 4), first introduced by Biro, Kern and Paulusma [5], as a way to change network structure. Formally, for an outcome of the bargaining game x on a graph G, the blocking set is the set of edges $\{(u, v) \in E\ :\ x_u + x_v < w_{uv}\}$.

By removing a blocking set B for an allocation x, we are left with a graph $G_B = (V,\ E \setminus B)$. We do not change the bargaining game divisions defined by x, and so we still distribute a total $\nu(G)$ surplus value across the contracting players. This is likely greater than $\nu(G_B)$, but the allocation x is then such that $x_u + x_v \geq w_{uv}$ for all remaining edges, making it a stable outcome. We relax the condition of balanced outcomes by allowing $x_u+x_v \geq w_{uv}$ for two players in a contract, as we reassign extra surplus gained from the blocked edges. Our first main contribution shows that if we remove a blocking set from a graph, we can compute a series of transfers that result in a balanced bargaining outcome for the resulting subgraph (Chapter 4). All possible contracting players will have monetary shares consistent with Nash's Bargaining Solution. An example is shown in Figure 1.2. The red edge is a blocking set for the graph G, the blue edges are negotiated contracts, and the red vertex values have been increased by the gained

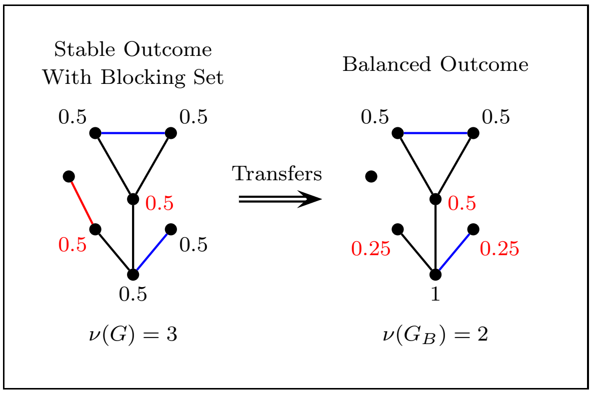

Figure 1.2: Balanced outcome after removal of a blocking set

blocking set surplus.

Theorem 1.1. *Consider the bargaining game on a graph $G = (V,\ E)$, with an empty core. Let (B, x) be a feasible blocking set and allocation pair on G. Then we can find a balanced bargaining allocation on the subgraph $G_B = (V,\ E \setminus B)$ in polynomial time.*

Naturally, this motivates the problem of finding minimal blocking sets, called the BLOCKING PAIRS PROBLEM. Unfortunately, this is an NP-complete problem in general graphs, so we focus our attention on restricted graph classes (Chapter 5). Our second and third main contributions are a polynomial time approximation scheme (PTAS) for finding minimal blocking sets in planar graphs with unit value contracts, and an algorithm that finds the optimal blocking set in graphs of bounded treewidth.

Theorem 1.2. *For planar graphs with unit edge weights, the* BLOCKING PAIRS *problem is $O(1)$-factor approximable.*

Theorem 1.3. *For a graph $G = (V,\ E)$ with tree decomposition $(Y',\ T')$ of width k, an optimal blocking set and allocation pair can be found in time $O(k \cdot 9^k \cdot |V|^3)$.*

This thesis is organized as follows. We start with relevant background information in Chapter 2. In this chapter, we explain the terminology and notation we use throughout the thesis, and discuss concepts used in later chapters. We formally define Gallai-Edmonds decompositions, solution concepts from cooperative game theory, tree decompositions and treewidth. Furthermore, we present an example of the general dynamic programming approach for solving problems on graphs of bounded treewidth.

Chapter 3 describes the bargaining game model we use, and formally defines stable and balanced solutions. We then survey previous work done with the model, the majority by Kleinberg and Tardos [18], and Bateni et al. [4]. We explain the relationship between stable and balanced solutions and the core and prekernel solution concepts from cooperative game theory. We then recall a proof by Bateni et al. that a graph is stable if and only if it has a balanced solution, and that we can efficiently construct all balanced solutions.

Chapter 4 is devoted to Blocking Sets. In this chapter, we formally define the BLOCKING PAIRS problem, and show Biro et al.'s proof that it is NP-hard [5], reducing from MAXIMUM INDEPENDENT SET. We then show that starting with an unstable graph, if we remove a blocking set to create stability, using ideas due to Stearns [31] and Faigle [12], we can find a balanced solution to the bargaining game.

Chapter 5 contains the main contributions of this thesis. It is concerned with algorithms for finding minimal blocking sets on certain graph classes. First, we provide an approximation algorithm for the Blocking Pairs problem on planar graphs. Then we present a polynomial time algorithm for optimally solving the problem on graphs of bounded treewidth.

Finally, in Chapter 6 we conclude and discuss a list of open problems and potential extensions for future work that we are interested in.

Chapter 2

Background

The topics and concepts used in this thesis are all quite broad. In this chapter, we cover a sufficient part of each for later application purposes. The reader is referred to references in each section to obtain any further knowledge.

We explain some basic preliminaries and notation in Section 2.1. In Section 2.2 we describe the Galai-Edmonds decomposition of a graph and its properties. We define the concepts of representing a graph as a tree (tree decomposition) and treewidth in Section 2.3. We also present an example of the general dynamic programming approach to solving several NP-hard problems on graphs of bounded treewidth. We conclude by defining the Core and Prekernel solution concepts of cooperative game theory in Section 2.4.

2.1 Preliminaries

We assume the reader is familiar with general concepts of linear programming such as primal and dual linear program (LP) pairs, strong and weak duality, complimentary slackness, and integer programs and their linear relaxations. The reader is referred to standard references if additional information is required [32].

We also assume the reader is familiar with concepts of graph theory such as undirected, directed, and bipartite graphs, trees, cycles, planarity, matchings, and connected components. Again, the reader is referred to standard references for appropriate background knowledge [33].

Our graph terminology is described here. All graphs are finite, simple and undirected. A graph G is represented by $G = (V,\ E)$, where V is the set of vertices and E is the set of edges. We may use $V(G)$ and $E(G)$ to refer to the set of vertices and edges of G respectively, if the graph being discussed is not evidently clear. We represent an edge in graph G between vertices u and v by $e = (u, v)$. We call u and v the *endpoints* of edge e. We say that vertices u and v are adjacent in graph G if there exists an edge $(u, v) \in E$.

We define n to be the number of vertices in a graph G, and m to be the number of edges. We define $\nu(G)$ to be the size of the maximum matching in G. A perfect matching is a matching M in a graph such that $|M| = \frac{n}{2}$. We may at some point consider the maximum fractional matching of a graph, defined as $\nu_f(G)$, where we allow fractional weights for matching edges. We say a matching M *covers* a vertex u if there is an edge $(u,v) \in M$. A matching M *exposes* a vertex u if it does not cover u. We call a vertex u *essential* if it is covered in every maximum matching of G. A vertex is *inessential* if it is not essential, that is, there exists at least one maximum matching in G where v is exposed.

We define $\delta(u) = \{v\ :\ (u,v) \in E\}$ to be the *neighbourhood* of a vertex u in graph G. We expand this definition to set notation for a subset $S \subseteq V$, with $\delta(S) = \{v\ :\ (u,v) \in E,\ u \in S,\ v \in V \setminus S\}$. We may specify the set of edges across which we are interested in a neighbourhood of vertices. We let $\delta_{E'}(u) = \{v\ :\ (u,v) \in E'\}$ for some $E' \subseteq E$, with a similar definition for a subset $S \subseteq V$.

An *n-clique* (K_n) is a graph G with n vertices in which every pair of vertices is connected by an edge. A graph G represented by K_{n_1,n_2} if it has a bipartition V_1 and V_2 such that $|V_1| = n_1$ and $|V_2| = n_2$ and there exists an edge (v_1, v_2) for every pair $v_1 \in V_1$, $v_2 \in V_2$.

A graph $G' = (V',\ E')$ is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$. A graph $G' = (V',\ E')$ is an *induced subgraph*, denoted by $G[V']$, if $V' \subseteq V$ and E' contains all edges with both endpoints in V'. We may use the notation $G \setminus v$ for some $v \in V$ to indicate $G' = (V \setminus v,\ E \setminus \delta(v))$, or $G[V \setminus v]$.

A graph (or component) G is said to be *factor-critical* if it has the property that for every vertex $v \in V$, the subgraph $G \setminus v$ has a perfect matching.

The *symmetric difference* of two sets of edges E_1 and E_2 is denoted as $E_1 \Delta E_2 = \{(u,v)\ :\ (u,v) \in E_1 \cup E_2,\ (u,v) \notin E_1 \cap E_2\}$.

2.2 Gallai-Edmonds Decompositions

The *Gallai-Edmonds decomposition* [21] for a graph G was independently discovered by Jack Edmonds and Tibor Gallai, and it provides many useful properties for discussing matchings. It is a partition of the vertex set $V(G)$ into three sets $\{D,\ A,\ C\}$. The set D contains the inessential vertices of G. The set A contains all the essential neighbours of D. That is, $A = \{u \in \delta(D)\}$. The set C contains the remaining $V \setminus (D \cup A)$ essential vertices.

We recall many useful properties about Gallai-Edmonds decompositions that we use in later chapters.

Lemma 2.1. *Let $G' = G \setminus v$ be the graph obtained from a graph G by removing a vertex $v \in A$. The Gallai-Edmonds decomposition of G' is $\{D,\ A \setminus \{v\},\ C\}$.*

Proof. Let $G' = G \setminus v$ for some $v \in A$. Since v is essential in G, removing it from G decreases the size of the maximum matching, that is $\nu(G') = \nu(G) - 1$. Consider now any maximum matching M in G, and let $e_v \in M$ be the matching edge that is incident to v. Then $M' = M \setminus e_v$ is a matching on G' of size $\nu(G')$, so it is a maximum matching.

Consider now an inessential vertex u in G. The above statement implies that u is inessential in G' as well. Consider the matching M_u on G that exposes u. Removing the edge incident to v in M_u creates a maximum matching in G' that exposes u.

Conversely, let u now be an inessential vertex in G'. We show that u is also inessential in G. Suppose for contradiction that u is essential in G. We know there is a maximum matching M'_u on G' that exposes u. Since $v \in A$, it must have an inessential neighbour $x \in D$. Let M_x be a maximum matching on G that exposes x.

Since M_x covers u, but M'_u does not, the symmetric difference $M_x \Delta M'_u$ contains a path P starting at u, whose first edge belongs to M_x. Note that P cannot end in an M'_u edge as otherwise $M'_u \Delta P$ is a maximum matching exposing u. So P's last edge also belongs to M_x. This means that the matching $M'_u \Delta P$ has one more edge than M'_u, and so can not be contained in G' (since M'_u is a maximum matching). Since every internal vertex of P is covered by both an M'_u edge and an M_x edge, and $u \in V(G')$, it must be the case that P's other endpoint is v. Then since x is M_x exposed, $x \notin P$, and so let the path $P' = P \cup (v, x)$. Then the matching $M_x \Delta P'$ is a matching of size $\nu(G)$ that exposes u, which is a contradiction. ☐

Lemma 2.2. *Every connected component in the graph $G[D]$ is factor-critical. The graph $G[C]$ has a perfect matching.*

Proof. Let $G_A = G \setminus A$ be the graph obtained from G by removing the set of vertices A and set of edges $\delta(A)$. By Lemma 2.1, the Gallai-Edmonds decomposition for G_A is $\{D, \emptyset, C\}$.

It is known that a graph G consisting of only inessential vertices is factor-critical, as proven in Theorem 3.1.13 in [21]. Since D is a set of inessential vertices in the graph G_A, each component of the graph $G[D]$ contains exclusively inessential vertices and so is factor critical.

Now consider the graph $G[C]$. Since C is a set of essential vertices in the graph G_A, it means that the graph $G[C]$ must contain a perfect matching. ☐

Lemma 2.3. *Let M be a maximum matching of a graph G. M contains a perfect matching from the graph $G[C]$, and a maximum matching from each connected component in the graph $G[D]$. Every vertex $v \in A$ is matched to a vertex $u \in D$.*

Proof. Let M be a maximum matching on G and let $M_A \subseteq M$ be the matching edges that are incident to vertices in A. Repeating the argument from Lemma 2.1 shows that $M' = M \setminus M_A$ is a maximum matching on the graph $G' = G \setminus A$. Lemma 2.2 implies that M' (and so M) must contain a perfect matching on the graph $G[C]$ and a maximum matching in each component of the graph $G[D]$. This proves the first part of the Lemma.

To prove the second part, consider any maximum matching M on the graph G. Let some vertex $v \in A$ be matched to a vertex u. Remove v from G to get the graph $G' = G \backslash v$, and consider the matching $M' = M \setminus (u, v)$. Since $|M'| = \nu(G) - 1$, M' is a maximum matching on G' that exposes u. By Lemma 2.1, this means that u must be inessential in G, and so $u \in D$. □

We will use Gallai-Edmonds decompositions to characterize stable and unstable graphs, to be defined in Chapters 3 and 4.

2.3 Tree Decompositions and Treewidth

In this section we describe graphs with bounded treewidth, which are known for their good algorithmic-properties. Many intractable problems can be solved in polynomial time on graphs of bounded treewidth.

The treewidth of a graph was first introduced by Robertson and Seymour [27] in their work on graph minors. Before we can describe this concept, we must first define the representation of a graph as a tree, which will be the basis of an algorithm in a later chapter.

Definition 2.4 ([27])**.** A tree decomposition of a graph $G = (V,\ E)$ is a pair $(Y,\ T)$ where T is a tree with node set I, and $Y = \{Y_i\ :\ i \in I\}$ is a family of subsets of V, with the following properties:

(TW1) $\bigcup_{i \in I}(Y_i) = V$

(TW2) For every edge $e = (u, v) \in E$, there exists an $i \in I$ such that $\{u, v\} \in Y_i$

(TW3) *(Running Intersection Property)* For $i,\ j,\ k \in I$, if j lies on a path in T from i to k, then $X_i \cap X_k \subseteq X_j$.

The *width* of a tree decomposition, is defined as $\max_{i \in I}(|X_i| - 1)$. The *treewidth* of a graph G is the minimum $w \geq 0$ such that G has a tree decomposition of width $\leq w$. Then, for example, trees and forests both have treewidth 1.

It has been shown when given a graph G and integer k, the problem if G has treewidth at most k is NP-complete. Fortunately, for many values of k, there are characterizations of graph classes with treewidth at most k based on forbidden minors. We do not go into more detail here, but rather refer the reader to work by Bodlaender for extra knowledge [7].

As was mentioned, many NP-complete problems have polynomial-time algorithms when restricted to graphs of bounded treewidth. While the class of graphs with bounded treewidth is not very large, these algorithms can lead to approximation schemes for larger graph classes, for example planar graphs. Baker [3] introduced a method for finding approximation algorithms for planar graphs by decomposing them into k-outerplanar subgraphs. It has be shown that k-outerplanar graphs have treewidth at most $(3k-1)$ [7], [26]. Using our efficient algorithms to solve problems in the k-outerplanar subgraphs, we can sometimes find PTASs for intractable problems in planar graphs.

We now describe a dynamic programming algorithm to solve the Maximum Independent Set problem on graphs of bounded treewidth. An independent set is a subset of vertices $I \subseteq$ such that there are no edges between any two members of I. The general problem is defined as follows:

Maximum Independent Set
Instance: a graph $G = (V,\ E)$ and an integer $k \geq 0$.
Question: does G contain an independent set I with $|I| \geq k$?

We begin by stating a useful Lemma also due to Bodlaender. The reader is referred to the referenced material for the proof.

Lemma 2.5 ([7])**.** *For a graph $G = (V,\ E)$, with tree decomposition $(Y',\ T')$ of width k, we can in linear time transform $(Y',\ T')$ into a new tree decomposition $(Y,\ T)$ of equal width k, where T is a binary tree (a binary tree decomposition). T will have $O(k \cdot |V|)$ nodes.*

Due to Lemma 2.5, without loss of generality we can assume that any given tree decomposition of a graph G is a binary tree decomposition.

Let $(Y,\ T)$ be a binary tree decomposition of G with treewidth k, where T is made up of the nodes $\{t_1, t_2, \cdots\}$. Arbitrarily pick a node to be the root of T, and let it be t_1. Let the bag at each node t_i be Y_i. For a node t_i, let D_i be the set of all vertices that appear in the bags of the proper descendents of t_i in T.

We define a function $A(S,\ t_i)$, where $S \subset Y_i$ is an independent set, to be the size of a maximal independent subset $I \subseteq (Y_i \cup D_i)$ such that $I \cap Y_i = S$. We also define a second function $B(S',\ \ell,\ u)$ for an independent set S', and for two adjacent nodes t_ℓ and t_u, where t_ℓ is further (lower) from the root of T than t_u (upper). We let $B(S',\ \ell,\ u)$ represent the size of the largest independent set $I \subset (Y_\ell \cup D_\ell)$ such that $(I \cap Y_\ell \cap Y_u) = S'$.

The algorithm does a bottom up traversal of the tree, computing the $A(S,\ t_i)$ values at every node for every subset $S \subset Y_i$. Recursive formulas for A and B are given as follows, where t_p and t_q are the children of a node t_i:

$$A(S,\ t_i) = |S| + \sum_{j=\,p,\ q} \Big(B(S \cap Y_j,\ j,\ i) - |S \cap Y_j| \Big)$$

and

$$B(S',\ \ell,\ u) = \max_{S_o \subset Y_\ell \,:\, S_o \cap Y_u = S'} A(S_o,\ t_\ell)$$

Naturally, we remove the corresponding summation indices in the expression for $A(S,\ t_i)$ if the node t_i does not have two children. At every node t_i, for each of the 2^k possible subsets $S \subset Y_i$, the algorithm includes S in a candidate independent set solution for G, and searches for an expansion of this partial solution. Including B in the calculation of A ensures that when we choose an independent subset $S \subset Y_i$, we find the largest independent set amongst the vertices of D_i to combine with S. Once we have finished computing the $A(S,\ t_1)$ values for all subsets $S \subset Y_1$, we find the maximum A value stored at the root t_1, which is the size of the maximum independent set for G.

We will use a similar dynamic programming technique to solve the Blocking Pairs problem on graphs of bounded treewidth (to be formally defined in Chapter 4).

2.4 Solution Concepts in Cooperative Game Theory

In this section we will define cooperative games and their solution concepts. We assume the reader is familiar with basic game theoretic concepts such as strategies, utility, payoff vectors, and coalitions. We refer the reader to the referenced material for more information. [25]

A *cooperative game* is a game where groups of players, or coalitions, enforce competitive strategies. The game is played between coalitions rather than individual players. Formally, a game is a cooperative game if it contains:

(a) A finite set of players N

(b) A characteristic value function $v : 2^N \Rightarrow \mathbb{R}$, assigning a payoff to every subset $S \subseteq N$, and $v(\emptyset) = 0$.

The characteristic function can instead be defined as a *cost* function $c : 2^N \Rightarrow \mathbb{R}$, in which case we call the game a *cost game*. For the purposes of this thesis, we use value functions as they are more appropriate for the games we study.

A main assumption in cooperative game theory is that the grand coalition N will always form. Usually the challenge is allocating the value $v(N)$ among the players in some fair way to form a solution $x \in \mathbb{R}^N$. We define three different solution concepts based on notions of fairness. We use the shorthand $x(S) = \sum_{i \in S} x_i$ for any vector x and subset $S \subseteq N$ throughout the remainder of this thesis.

We begin with the *Core* of a cooperative game, whose modern definition is sometimes attributed to Donald B. Gilles [15] from his work in 1959.

Definition 2.6. Let $v : 2^N \Rightarrow \mathbb{R}$ be the value function for a game with player set N. The core of the game, $C(v)$, is the set of payoff vectors

$$C(v) = \left\{ x \in \mathbb{R}^N \ : \ x(S) \geq v(S), \ \forall \ S \subseteq N; \quad x(N) = v(N) \right\}$$

Intuitively, for any solution $x \in C(v)$, no player or coalition has an incentive to deviate from the grand coalition N, as they will lower their combined utility. Core members are highly desirable solutions to games, unfortunately the core is often empty. In these situations, we try to find solutions with different ideas for fairness.

Before we can define our next solution concept, the *Prekernel*, we must first define a concept of the *power* of player i with respect to player j.

Definition 2.7. Let x be a solution to some cooperative game with characteristic function $v : 2^N \Rightarrow \mathbb{R}$ and player set N. The power of player i with respect to player j is defined as

$$s_{ij}(x) = \max_{S \subset N} \left\{ v(S) - x(S) \ : \ i \in S, \ j \notin S \right\}$$

We are now ready to define the prekernel, which was first introduced by Morton Davis and Michael Maschler [10] in 1965.

Definition 2.8. Let $v : 2^N \Rightarrow \mathbb{R}$ be the value function for a game with player set N. The prekernel of the game, $P(v)$, is the set of payoff vectors such that for every pair $i, j \in N$, we have that $s_{ij}(x) = s_{ji}(x)$.

While the core of the game provides a notion of stability where no coalition has an incentive to leave the grand coalition, the prekernel instead provides a set of solutions that are balanced with each other in terms of power.

Next, we define a unique solution, the *nucleolus* of a game, which was introduced by Schmeidler [28] in 1969. Let the *excess* of a coalition be the value $e(S) = v(S) - x(S)$ for $S \subseteq N$. Let $\phi : \mathbb{R}^N \to \mathbb{R}^{2^N}$ be a function that computes the vector $\phi(x)$ of excesses for an allocation x, arranged in non-increasing order. That is, $\phi_i(x) \geq \phi_j(x)$ for $i < j$. Note that an allocation y is a core member if $\phi_1(y) \leq 0$.

Definition 2.9. Let $v : 2^N \Rightarrow \mathbb{R}$ be the value function for a game with player set N. The nucleolus of the game is the allocation x that lexicographically minimizes the vector of excesses $\phi(x)$, resulting from the function $\phi : N \to 2^N$ on x.

The nucleolus tries to combine the two ideas of fairness formulated by the core and the prekernel. Schmeidler [28] proved that the nucleolus of a game is always contained in the prekernel, and that if the core is non-empty, the nucleolus is a core member. Its definition forces a unique solution, and hence it is often used to model an ideal fair allocation.

While there are many other solution concepts in cooperative game theory such as the *least-core* or the *stable set*, our work focuses on the core and the prekernel in Chapters 3 and 4. Even though the nucleolus defines a unique allocation, we are more interested in efficiently computing any balanced solution to our games.

Chapter 3

Bargaining Solutions in Social Exchange Networks

This chapter surveys current significant known results for solutions to bargaining games in social exchange networks. In Section 3.1, we build a model for the game and define the form of a solution. We also introduce some notation we will use throughout this chapter. In Section 3.2, we define two bargaining solution concepts and discuss their individual properties.

In Section 3.3, using graph theoretical and combinatorial arguments, we explain a relationship between bargaining solutions and solution concepts from cooperative game theory. We discuss the process of finding bargaining solution instances with these results. Finally, we finish this chapter by extending our model in Section 3.4, and present a similar relationship between the extended bargaining game and cooperative game theory solutions.

3.1 A Model for Bargaining Games

In the network bargaining game model, there is a set N of n agents. For each pair of agents $i,\ j \in N$, we are given a weight w_{ij} representing the value of a potential contract between i and j. Each agent is limited to making a single deal. The contract weights describe an edge-weighted graph $G = (V,\ E)$. We create a vertex v_i for each agent $i \in N$. We add an edge (i, j) with weight w_{ij} to E for every pair of agents $i,\ j \in N$ with $w_{ij} > 0$. We may refer to a game as *the bargaining game on* $G = (V,\ E)$, where G is a weighted graph. We use G to define the set of agents ($N = V$) and potential contracts, (the edge set E with weights w_{ij}).

Every agent is rational and self-interested. All agents are equal in bargaining skill, and have full knowledge of the tastes and preferences of the others. In addition, all agents know the weight w_{ij} of every potential contract between any pair i and j.

Let x_i denote the surplus each agent i receives. We define a solution to the game as a pair (x, M) where $M \subseteq E$ is a set of bargained contracts such that $x_i + x_j = w_{ij}$ for all $(i, j) \in M$, and $x = \{x_i \ : \ i \in N\}$.

We define the *alternative option* (or outside option) of an agent i as the best deal that the agent could make with someone he has not already made a contract with.

Definition 3.1. Let x be a solution where agents i and j are in a contract. The *outside option*, α_i, of agent i is

$$\alpha_i = \max_{k:(i,k)\in E\setminus M} \Big\{ w_{ik} - x_k \Big\}$$

If the set $\{k \ : \ (i,k) \in E \setminus M\}$ is empty, we define the outside option of i to be zero.

3.2 Stable and Balanced Solutions

The theory of bargaining in some sense can be viewed as a way to select the most reasonable outcome of a negotiation. Every agent seeks a division where they receive more than their outside option. We want to identify those solutions that are also balanced for all players. In this section we define two solution concepts we will use consistently throughout this thesis.

Intuitively, if an agent i makes a deal in which they earn less than their outside option α_i, then they can improve their earnings by switching to form a contract with the agent offering that outside option. We seek to find solutions where no switch is beneficial for any agent. Kleinberg and Tardos formulated this idea and defined it formally as a *stable solution.*

Definition 3.2 ([18])**.** Let $i \in N$ be an agent in a bargaining game, and let (x, M) be a solution to the game. We call (x, M) a *stable solution* if for all $(i, j) \in M$, $x_i \geq \alpha_i$.

We call a graph $G = (V,\ E)$ *stable* if the bargaining game on G has a stable solution. Conversely, we call G *unstable* if the bargaining game on G does not have a stable solution.

In 1950, Nash [24] argued that a solution to the bargaining game should have an extra notion of equality for contracting agents. He proposed his Nash Bargaining Solution with an idea of balancedness, suggesting that two agents will agree on a division of surplus that lies halfway between the extremes of their alternate options. We use the terminology of Kleinberg and Tardos [18], and call these solutions *balanced.* We define them formally as follows.

Definition 3.3 (**Nash Bargaining Solution**)**.** Let (x, M) be a solution to the bargaining game. Let i and j be two contracting agents negotiating a deal worth w_{ij}. Let α_i and α_j be the outside options for agents i and j respectively. A *balanced solution* is the agreement that splits the surplus $s = w_{ij} - \alpha_i - \alpha_j$ evenly between the two agents for all pairs $(i, j) \in M$. That is, $x_i = \alpha_i + \frac{s}{2}$ and $x_j = \alpha_j + \frac{s}{2}$. We often use an equivalent condition that a solution is balanced if for all $(i, j) \in M$, $(x_i - \alpha_i) = (x_j - \alpha_j)$.

We will show in Theorem 3.5 that for a stable solution (x, M), M must be a maximum weight matching on the underlying graph G. Since by definition every balanced solution is also a stable solution, it follows that M is also a maximum weight matching for balanced solutions.

We also note that no solution can distribute a total surplus that is larger than the size of a maximum weight matching, following simply from the definition of matchings.

3.3 Finding Stable and Balanced Solutions

Network bargaining theory tries to understand power imbalances in relationships between pairs of agents in a bargaining game. The general hypothesis is that these imbalances are mainly due to *structural* properties, that is, the position of agents within a social exchange network. Intuitively, we may think that central positions, or those with the highest number of options for potential contracts hold the power in bargaining games. These are sometimes the case, however we can build simple examples that demonstrate that we can not characterize powerful positions this easily.

Consider the bargaining game on the simple 5-vertex path, shown in Figure 3.1. Our first thought may be that the central agent c will dominate negotiations, however upon further analysis and experiments, we find that agents b and d hold most of the power in the network. The central position of c does not confer any real bargaining power, since its two potential contracting neighbours, b and d, both have alternate and very weak neighbours of their own.

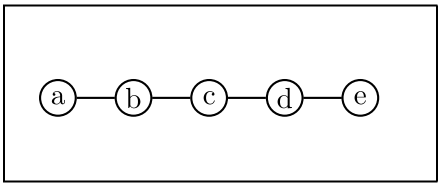

Figure 3.1: A 5-Vertex Path

A natural question that is aggressively pursued is to find simple structural concepts that explain these power imbalances, and that can also confirm results established from experimental work with human agents. In this section we will discuss how some of these graph and game theoretic concepts help in finding stable and balanced bargaining solutions, which assign values based on the power of an agent in the network.

We begin by giving an efficient characterization of stable graphs for the unweighted case using a combinatorial argument due to Kleinberg and Tardos [18]. It uses the Galai-Edmonds decomposition of a graph G, which we defined in Section 2.2.

Theorem 3.4. *Consider the bargaining game on a graph $G = (V,\ E)$ with Galai-Edmonds decomposition $V = (D,\ A,\ C)$. Let $\{D_1, D_2, \cdots\}$ be the connected components of the graph $G[D]$. G is a stable graph if and only if $|D_k| = 1$ for all k.*

Proof. Suppose first that for all k, $|D_k| = 1$. Then consider the solution x on any maximum matching M defined as follows:

$$x_i = \begin{cases} 0 & : \text{ if } i \in D \\ 1 & : \text{ if } i \in A \\ \frac{1}{2} & : \text{ if } i \in C \end{cases}$$

We show that x is a stable solution. Since $|D_k| = 1$ for all k, $\alpha_d = 0$ for all $d \in D$, and so $x_d = \alpha_d = 0$. By Lemma 2.3, since every $a \in A$ is matched to some $d \in D$, it must be that a has at least two neighbours in D (as otherwise d would be an essential vertex). This means that for every $a \in A$, $\alpha_a = 1$, but since $x_a = 1$, it is the case that $x_a = \alpha_a = 1$. Finally, consider some node $c \in C$. Since c only has neighbours in $A \cup C$, $\alpha_c \leq \frac{1}{2}$. Then $x_c \geq \alpha_c$, implying that x is a stable solution, and so G is a stable graph.

Next, suppose that G is a stable graph. Let (x, M) be a stable solution for the bargaining game on G where M is a maximum matching. We show that $|D_k| = 1$ for all k. Suppose for contradiction that for some k, $|D_k| \geq 3$ (we say 3 since by Lemma 2.2, D_k is factor-critical and hence is an odd component). It is known that the size of the maximum fractional matching in factor-critical graphs of size $|D_k|$ is $\nu_f(D_k) = \frac{|D_k|}{2}$. This means that the minimum fractional vertex cover is at least $\frac{|D_k|}{2}$, since the problems are duals of each other. Since we are only able to distribute $|M| = \nu(G)$ across the x variables, this means that there must be an edge $(i, j) \in E$ such that $x_i + x_j < 1$. By definition of a solution (x, M), this means that $(i, j) \notin M$. So then $\alpha_i \geq (1 - x_j) > x_i$, contradicting stability. $\square$

While Theorem 3.4 gives us a nice characterization of stable graphs, its use is limited since it only applies to the unweighted case. We wish to provide a similar efficient characterization for general weighted graphs. We draw connections to solution concepts from cooperative game theory to give us our desired result.

We begin by defining a value function for any subset of the vertices of G. We use $\nu(S)$ as our value function for $S \subseteq N$. As in Section 2.1, $\nu(S)$ is the total weight of a maximum weight matching on the graph $G[S]$. We use this value function in the definition from Section 2.4, so the core of the bargaining game consists of any bargaining solution x such that $x(S) \geq \nu(S)$ for all subsets $S \subset N$, and $x(N) = \nu(N)$. We simplify this core membership condition as follows. Let (x, M) be a solution for the bargaining game on some graph $G = (V,\ E)$. For all edges $(i, j) \in E$, if $x_i + x_j \geq w_{ij}$. We show that this condition is in fact equivalent to the core membership condition. Consider a subset $S \subseteq N$, and let M_S be a maximum matching in $G[S]$. Then if for all edges $(i, j) \in M_S$, $x_i + x_j \geq w_{ij}$, it follows that $x(S) \geq \nu(S)$ as required. We use this fact to prove a result due to Bateni et al. [4]

Theorem 3.5. *Let (x, M) be a solution to the bargaining game on a graph $G = (V,\ E)$. x is a stable solution if and only if x is in the core of the bargaining game.*

Proof. We use the strong duality theorem and complementary slackness conditions from linear programming theory to prove the theorem. Suppose first that (x, M) is a stable solution. We show that x must satisfy both conditions for core membership. We begin with the second condition. Since by definition $x(V) = \sum_{(i,j)\in M} w_{ij}$, we must show simply that M is a maximum weight matching. Consider the following LP and its dual.

$$
\begin{array}{llcccll}
(LP_C) & \text{Min} & \displaystyle\sum_{i\in N} x_i & & & & \\
 & \text{subject to} & x_i + x_j & \geq & w_{ij} & \forall\ (i,j) \in E & (3.1) \\
 & & x_i & \geq & 0 & \forall\ i \in V &
\end{array}
$$

$$
\begin{array}{llcccll}
(D_C) & \text{Max} & w^{\intercal} y & & & & \\
 & \text{subject to} & \displaystyle\sum_{j\in\delta(i)} y_{ij} & \leq & 1 & \forall\ i \in V & (3.2) \\
 & & y_{ij} & \geq & 0 & \forall\ (i,j) \in E &
\end{array}
$$

Let y be a solution to (D_C) such that $y_{ij} = 1$ if $(i, j) \in M$ and $y_{ij} = 0$ otherwise. For every edge $(i, j) \in M$, by definition we have $x_i + x_j = w_{ij}$. So consider an edge $(i, j) \notin M$. By the definition of an outside option, $\alpha_i \geq w_{ij} - x_j$. Since x is a stable solution, $x_i \geq \alpha_i$ by definition. So it follows that $x_i + x_j \geq w_{ij}$. Then x and y are feasible solutions to (LP_C) and (D_C) with equal objective values, which by strong duality means they are optimal solutions. Since (D_C) is simply the matching LP for the graph G, we conclude that M is in fact a maximum matching, and so $x(V) = \nu(V)$ as required. The first core membership condition is satisfied by constraint (3.1) since x is a feasible solution to (LP_C).

Next we prove that if x is a member of the core of the bargaining game, it must correspond to a stable bargaining solution (x, M) on G where M is a maximum matching. Since x is a member of the core, it must be a feasible solution to the above linear program (LP_C), with an objective value of $\nu(V)$. Consider any maximum-weight matching M and set $y_{ij} = 1$ if $(i, j) \in M$, and $y_{ij} = 0$ otherwise. Then y is a feasible solution to (D_C). Since $x(V) = |M| = \nu(V)$, by strong duality it must be that x and y are optimal solutions to (LP_C) and (D_C) respectively. Then by complimentary slackness, for every edge (i, j) with $y_{ij} > 0$, $x_i + x_j = w_{ij}$. So (x, M) is a solution to the bargaining game. Now consider any vertex $i \in V$. Let j be the neighbour of i that induces the outside option of i, Ie.

$\alpha_i = (w_{ij} - x_j)$. Since $x_i + x_j \geq w_{ij}$, it follows directly that $x_i \geq \alpha_i$ for all $i \in V$. So (x, M) is in fact a stable bargaining solution. □

In addition to showing a one-to-one correlation between members of the core and stable solutions, the proof of Theorem 3.5 gives us the desired characterization of stable graphs.

Corollary 3.6 ([4]). *The bargaining game on a graph G has a non-empty core, and so G is a stable graph, if and only if (D_C) has an integral optimal solution for finding a maximum-weight matching.*

Now that we have found an efficient way to decide whether a general weighted graph has a stable solution, we steer our focus towards finding balanced bargaining solutions. As with stable solutions, we begin by restricting our discussion to graphs where the edge weights are all $w_{ij} = 1$ for $(i, j) \in E$. With a combinatorial argument, Kleinberg and Tardos [18] prove that a graph G contains a balanced solution if and only if it contains a stable solution. We state the theorem and then sketch its proof, and refer the reader to the referenced material for a more details.

Theorem 3.7. *For a graph $G = (V,\ E)$ with unit edge weights, a balanced solution exists to the bargaining game on G if and only if a stable solution exists to the bargaining game on G.*

The existence of a stable solution if a balanced solution exists follows directly from their definitions. We suppose then that there exists a stable solution to the game on G. We argue the existence of a balanced solution by describing an algorithm that finds such a solution.

First, we find the Gallai-Edmonds decomposition $V = (D \cup A \cup C)$ of G. By Theorem 3.4, D must be an independent set. We begin with the partial stable solution (x, M) where we set $x_v = 0$ for $v \in D$, and $x_v = 1$ for $v \in A$, and M is any maximum matching in G. Define a set $S' = \{D \cup A\}$. The x values for vertices in the set S' will not change throughout the algorithm. We will iteratively assign x-values to vertices and include them in S'. Let the slack of an edge be $\sigma_{uv} = x_u + x_v - 1$. Note that in a stable solution the slack at every edge is non-negative. Kleinberg and Tardos [18] define a linear program which finds the maximum slack across the non-matching edges with at least one endpoint in $V \setminus S'$. We repeatedly solve the LP, which identifies specific subgraph structures that limit the maximum slack. The endpoints of matching edges (u, v) in the limiting structure are then gradually assigned x values. The slack is spread through the graph as much as possible while balancing matched edges, and we expand S' with the newly-assigned vertices. We repeat this step until $V \setminus S' = \emptyset$, and (x, M) is a balanced bargaining solution.

Theorem 3.7 combined with Theorem 3.4 gives us a nice way of showing existence of balanced bargaining solutions in unweighted graphs. Again, we wish to take this result further and find a simple characterization for graphs with general edge weights. Similarly to stable solutions, we shift our focus towards concepts from cooperative game theory to find our desired result.

We use the same value function as we did for the proof of Theorem 3.5, $\nu(S)$, which is equal to the maximum matching on the graph $G[S]$. We relate balanced solutions to the prekernel of the bargaining game. Recall the necessary conditions of prekernel membership, $s_{ij}(x) = s_{ji}(x)$ for every pair $i, j \in N$, as in Definition 2.8. The condition uses the notion of the power of agent i with respect to agent j, defined as

$$s_{ij}(x) = \max_{S \subset N} \left\{ v(S) - x(S) \ : \ i \in S, \ j \notin S \right\}$$

We first prove a useful claim, which simplifies the calculation of the power of an agent with respect to another.

Lemma 3.8 ([4]). *Let (x, M) be a stable solution to the bargaining game on a graph $G = (V, \ E)$. The formula for $s_{ij}(x)$ can be simplified as follows: $s_{ij}(x) = \max_{k \in \delta(i)}\{w_{ik} - x_i - x_k : k \neq j\}$, taking the maximum to be $-x_i$ over the empty set. That is, it is sufficient to only consider coalitions of the form $S = \{i, k\}$ in the definition of the power of agent i with respect to agent j.*

Proof. Suppose for contradiction that $s_{ij}(x) \neq s' := \max_{k \in \delta(i); k \neq j}\{w_{ik} - x_i - x_k\}$. Then there exists a subset $S \subseteq V$ containing i but not j such that $\nu(S) - x(S) > s'$. Let M_S be any maximum matching in the graph $G[S]$. Every edge $(u, v) \in M_S$ where $u \neq i, v \neq i$ contributes w_{uv} to $\nu(S)$ and at least w_{uv} to $x(S)$ since x is a stable solution, implying that $x_u + x_v \geq w_{uv}$ by Theorem 3.5. So adding any set of vertices to S that expands the maximum matching M_S can not be of any help when calculating $s_{ij}(x)$. Clearly, including any vertex that does not expand the maximum matching M_S does not help either when maximizing $s_{ij}(x)$. So we consider only those coalitions that include a single neighbour of i when maximizing $s_{ij}(x)$. ☐

We do note that the formula given for $s_{ij}(x)$ in Lemma 3.8 does not take into account the case in the weighted instance when the maximizing coalition may be $S = \{i\}$. This happens when $\alpha_i = 0$ and $x_k > w_{ik}$ for all $k \in \delta(i)$, $k \neq j$. This slight modification does not change our discussion for the rest of this chapter, but we will formally include it in the following chapter.

Using Lemma 3.8, we now prove a result that will directly imply the main presented result of this section.

Theorem 3.9 ([4])**.** *Let (x, M) be a solution to the bargaining game on a graph $G = (V,\ E)$. x is stable and balanced if and only if x is in the intersection of the core and prekernel of the bargaining game.*

Proof. Suppose first that (x, M) is a solution to the bargaining game, and that x is stable and balanced. We have already shown by Theorem 3.5 that x is a member of the core. For two agents i and j we consider two cases and show that in both x is in the prekernel.

1. $(i, j) \in M$: Consider the simplified formula for $s_{ij}(x)$. x_i is a fixed value in the formula, so we consider $\max_{k \in \delta(i); k \neq j}\{w_{ik} - x_k\}$. Note that this is the exact definition of α_i, the outside option of i. In any balanced solution, by definition we have that for any $(i, j) \in M$, $x_i - \alpha_i = x_j - \alpha_j$. This implies that $s_{ij}(x) = s_{ji}(x)$ as required.

2. $(i, j) \notin M$: If i is matched in M, that is there exists some k such that $(i, k) \in M$, then $s_{ij} \geq w_{ik} - x_i - x_k = 0$. Since x is in the core, $s_{ij}(x) \leq 0$, implying that $s_{ij}(x) = 0$. If there is no such k, then $x_i = 0$ by definition. Then consider any edge $(i, k') \in E$, $k' \neq j$. Again since x is in the core and $x_i = 0$, it must be that $x_{k'} = w_{ik'}$, again implying that $s_{ij}(x) = 0$. Finally if there was no such $k' \in V$, then $s_{ij}(x) = -x_i = 0$. We use a similar argument for j to show that $s_{ji}(x) = 0$, and thus $s_{ij}(x) = s_{ji}(x)$ as desired.

Suppose now that x is in the intersection of the prekernel and the core of the bargaining game. Consider an edge $(i, j) \in M$. By the simplified definition of $s_{ij}(x)$, $s_{ij}(x) = \alpha_i - x_i$. Similarly, $s_{ji}(x) = \alpha_j - x_j$. By the definition of prekernel, for every pair $i, j \in V$, we know that $s_{ij}(x) = s_{ji}(x)$. This directly implies that x is in fact a balanced bargaining solution. □

Kleinberg and Tardos [18] establish the following main result, by extending the combinatorial argument used to prove Theorem 3.7. We argue its correctness based on the mentioned results from cooperative game theory.

Theorem 3.10. *The bargaining game on a graph G has a stable solution if and only if it has a balanced solution, and the set of all balanced solutions can be constructed in polynomial time.*

The proof of the first part of Theorem 3.10 follows from Theorem 3.9 and the known economic fact that if the core is non-empty, then the core intersect prekernel is also non-empty [28]. The constructibility of all balanced solutions follows from work done by Meinhardt [22], who proves using an LP-based algorithm that the prekernel can be efficiently constructed for zero-monotonic cooperative games (ZMCG). ZMCG's are defined as follows:

Definition 3.11. A game is called *zero-monotonic* if $\nu(S) \leq \nu(T) - \sum_{i \in T \setminus S} \nu(\{i\})$ whenever $S \subseteq T \neq \emptyset$.

The bargaining game on a graph G is then a ZMCG, since $\nu(\{u\}) = 0$ for any single agent $u \in N$, and $\nu(S) \leq \nu(T)$ if $S \subseteq T$.

Meinhardt's algorithm, using a set of LPs, iteratively computes a series of maximal transfers between agents (we will explore this idea further in the next chapter). It then computes the intersection of the solution sets from the solved LPs, which include elements of the kernel of the game (similar solution concept to prekernel). This implies that we can construct the prekernel in polynomial time, as it was shown that the prekernel and kernel coincide for ZMCG's. Hence, we can efficiently construct all balanced solutions.

3.4 Extending the Model - Agent Capacities

In this section we discuss an extension to the model where we allow agents to participate in multiple contracts. Each agent i is assigned a capacity $c_i > 0$. Bateni et al. established many of the results from Section 3.3 for this model. In order to discuss these results, we must first redefine some concepts.

Let (x, M) be a solution to the bargaining game with capacities. Here, x_i is the aggregate of the surpluses that agent i receives from all the deals it makes. Let z_{ij} be the amount of money i earns from its contract with j. If $(i, j) \in M$ then $z_{ij} + z_{ji} = w_{ij}$. Conversely, if $(i, j) \notin M$ then $z_{ij} = z_{ji} = 0$. We may refer to a solution to the bargaining game as (z, M), where $x_i = \sum_{j \in \delta(i)} z_{ij}$.

Definition 3.12. Let (z, M) be a solution to the bargaining game with capacities. The outside option α_i of agent i is

$$\alpha_i = \max_{k:(i,k) \in E \setminus M} \left\{ \max_{j:(j,k) \in M} (w_{ik} - I_k z_{kj}) \right\}$$

where $I_k = 1$ if agent k is utilized to capacity, and $I_k = 0$ otherwise. If the set $\{k : (i, k) \in E \setminus M\}$ is empty, we define the outside option of i to be zero.

Definition 3.13. Let (z, M) be a solution to the bargaining game on a graph $G = (V,\ E)$. z is a *stable solution* if for all $(i, k) \in M$, $z_{ik} \geq \alpha_i$, and $\alpha_i = 0$ if i has residual capacity.

Definition 3.14. Let (z, M) be a solution to the bargaining game on a graph $G = (V,\ E)$. z is a *balanced solution* if for all $(i, k) \in M$, $z_{ik} - \alpha_i = z_{ki} - \alpha_k$.

Definition 3.15. Let $S \subseteq V$ be a coalition in the bipartite bargaining game with arbitrary capacities on the graph $G = (V,\ E)$. Let $v(S)$ be a value function equal to the optimum value of $(LP1)$ for S, defined as follows.

$$
\begin{array}{llllll}
(LP1) & \text{Max} & \sum_{(i,j)\in S} x_{ij} w_{ij} & & & \\
& \text{subject to} & \sum_{j\in\{\delta(i)\cap S\}} & \leq & c_i & \forall\, i \in S \\
& & x_{ij} & \leq & 1 & \forall\, i,j \in S \\
& & x_{ij} & \geq & 0 & \forall\, i,j \in S
\end{array}
\tag{3.3}
$$

Intuitively, $v(S)$ finds the maximum total payout for all agents in S over all possible feasible contract schemes.

We begin with a Lemma giving an example of a bipartite network with arbitrary capacities. We will show that core solutions are not necessarily stable solutions, and so we will have to restrict the set of example graphs in our discussion.

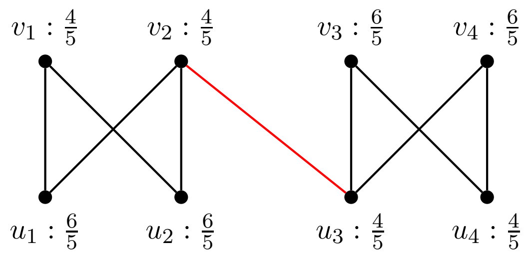

Figure 3.2: Core $\not\subseteq$ Stable solutions in bipartite networks with arbitrary capacities

Lemma 3.16 ([4]). *Let (x, M) be a core solution to the bargaining game on $G = (V,\ E)$ with arbitrary capacities. Then (x, M) is not necessarily a stable solution.*

Proof. We prove this Lemma with a counterexample. Consider the network shown in Figure 3.2. Let all the agents have capacity two, and let every contract be worth one. Let (x, M) be a solution such that $x_{v_1} = x_{v_2} = x_{u_3} = x_{u_4} = \frac{4}{5}$, and $x_{u_1} = x_{u_2} = x_{v_3} = x_{v_4} = \frac{6}{5}$ as shown, and every agent is in two contracts. Consider the agents v_2 and u_3. By the pigeonhole principle, each must have a contract from which they earn less than $\frac{1}{2}$. So they have incentive to cancel those contracts and split the contract between themselves, (v_2, u_3), in half. So (x, M) can not be a stable solution to the game.

We now show that (x, M) is in fact in the core. In this network, $\nu(V) = 8$, and $x(V) = 8$ as required. Consider a subset $S \subseteq V$. We analyze the core condition $x(S) \geq$

$\nu(S)$ for varying sized coalitions S. If $|S| = 0$ or $|S| = 1$, then the core condition is satisfied as $\nu(S) = 0$. If $|S| = 2$, then $\nu(S) \leq 1$, and $x(S) \geq \frac{8}{5}$. For $|S| = 3$, $\nu(S) \leq 2$, and $x(S) \geq \frac{12}{5}$, so the condition is satisfied. If $|S| = 4$ and $\nu(S) = 4$, then we know S must contain the endpoints of all four black edges from one side of the network, Ie. either $S = \{v_1, v_2, u_1, u_2\}$ or $S = \{v_3, v_4, u_3, u_4\}$. In this case $x(S) = 4$ as well. Otherwise $\nu(S) \leq 3$, and then the core condition is satisfied as $x(S) \geq \frac{16}{5}$. By a similar argument, coalitions of size $|S| = 5$ and $|S| = 7$ have $\nu(S)$ values of at most 4 and 6, respectively. In both cases the core condition is satisfied as they receive $x(S) \geq \frac{22}{5}$ and $x(S) \geq \frac{34}{5}$. This leaves the case when $|S| = 6$. Here $x(S) \geq \frac{28}{5}$, and $\nu(S) \leq 5$, as two vertices will not be able to exhaust their capacities in a six vertex coalition.

The above case analysis shows that the built example allocation x is in fact a member of the core of the bargaining game on G, however is not a stable solution. □

The example in the proof of Lemma 3.16 implies that the core does not fully coincide with the set of stable solutions for bipartite graphs with arbitrary vertex capacities. Bateni et al. restrict their work to a set of graphs they call *constrained capacity bipartite graphs*. These are bipartite graphs with bipartition $V = (V_1 \cup V_2)$, where one side of the network is limited to a single deal, and the other side of the network has arbitrary capacities. Formally, $c_i = 1$ for $i \in V_1$, and $c_i \geq 0$ for $i \in V_2$. While this restriction to the model has its limitations, it does still present a practical setting. Consider an example network where on one side the agents are screenwriters, each with a movie script for sale. The agents forming the other side of the network are production studios that are interested in buying the scripts. Each studio can produce a number of movies, however each script can only be sold to a single producing company.

By only considering constrained bipartite graphs, it is possible to relate stable and balanced solutions to the core and prekernel solution concepts from cooperative game theory. Using $(LP1)$, as defined in Definition 3.15, with its dual and complimentary slackness, we can show analogous results to the ones we presented in Section 3.3. We state these results for completeness, and refer the reader to the referenced article for the proofs.

Theorem 3.17 ([4])**.** *For the bargaining game on a constrained bipartite graph $G = (V,\ E)$, a solution (x, M) is a stable solution if and only if it is in the core of the game*

Theorem 3.18 ([4])**.** *For the bargaining game on a constrained bipartite graph $G = (V,\ E)$, a solution (x, M) is a stable and balanced solution if and only if it is in the core intersect prekernel of the game*

Chapter 4

Blocking Sets

In this chapter, we explore an approach to finding fair solutions in unstable graphs. We discuss the idea of blocking edges without reducing the amount we can distribute to the vertices.

As mentioned in Section 3.2, an unstable graph $G = (V,\ E)$ does not allow a stable solution. We study the concept of the *blocking pairs* of an allocation $x \in \mathbb{R}^V$ as edges with under-assigned endpoints. Formally, we define the set of all blocking pairs, the *blocking set* of an allocation x, as $B(x) = \{(u, v) \mid x_u + x_v < w_{uv}\}$. We call (B, x) a blocking set and allocation pair where $B = B(x)$, and say B is the blocking set induced by x.

Biro, Kern and Paulusma studied the connection between blocking sets and testing if a weighted graph allows a stable solution, Ie. has a non-empty core. Any core member of the matching game on $G = (V,\ E)$ is an allocation x with $B(x) = \emptyset$. If the core is empty, we can try to minimize the number of blocking pairs. This leads to the decision problem:

Blocking Pairs
Instance: a matching game on $G = (V,\ E)$ and an integer $k \geq 0$.
Question: does G allow an allocation x with $|B(x)| \leq k$?

We note this problem does differ from the Edge-Deletion Stability problem, defined formally in Chapter 6, which searches for a minimal edge-set to remove such that the remaining graph is stable. Removing the edges of a minimum blocking set for a graph G does not necessarily produce a stable graph. Consider the graph K_5, with a single additional edge (a, b) forming a separate component, as shown in Figure 4.1. The allocation $x_u = \frac{1}{2}$ if $u \neq a$, $x_a = 0$ induces a blocking set of size 1, shown in red. Conversely, a minimum edge-deletion stabilizer set for this graph has size 4, also shown in red. By removing any less than three edges in the neighbourhood of a single vertex, the odd cycle contained in the K_5 component is not broken, and all the vertices in that component will remain inessential. Removing a subset of three edges in the neighbourhood of a vertex v also does not create stability, as matching v to its remaining neighbour will still leave an odd cycle between the unmatched vertices.

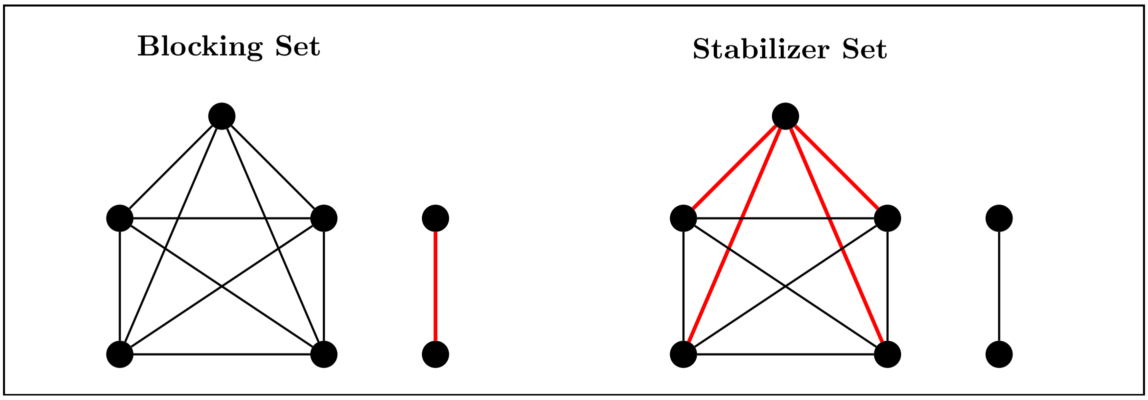

Figure 4.1: BLOCKING PAIRS is not equivalent to EDGE-DELETION STABILITY

The rest of this chapter is organized as follows. We restate the NP-completeness theorem for the blocking pairs decision problem, as proven by Biro, Kern and Paulusma [5]. We then motivate the reason for finding blocking sets, as we can remove a blocking set and find a balanced bargaining allocation on the resulting graph.

4.1 Complexity of the Blocking Pairs Problem

In this section, we provide a proof to show that the BLOCKING PAIRS problem is NP-complete. We start with a lemma to be used later.

Lemma 4.1 ([5]). *Let K be a complete graph with vertex set $\{1, \cdots, \ell\}$ for some odd integer ℓ, and let $x \in \mathbb{R}_+^K$. If $x(K) < \frac{\ell}{2}$ then $|B(x)| \geq \frac{\ell-1}{2}$ holds.*

Proof. We write $\ell = 2q+1$ and use induction on q. For the base case, if $q = 1$, then the statement holds trivially. Suppose $q \geq 2$. We can assume without loss of generality that $x_1 \leq x_2 \leq \cdots \leq x_{2q+1}$. Since $x(K) < \frac{\ell}{2}$, we have that $x_1 < \frac{1}{2}$ by the pigeonhole principle. If $x_1 + x_{2q+1} < 1$, then $x_1 + x_i < 1$ for $2 \leq i \leq 2q+1$. So we have $2q \geq \frac{\ell-1}{2}$ blocking pairs. Suppose that $x_1 + x_{2q+1} \geq 1$. Then $x_2 + x_3 + \cdots + x_{2q} < \frac{2q-1}{2}$. By induction this yields $q-1$ blocking pairs. Note also that $x_2 < \frac{1}{2}$ holds by the same pigeonhole principle. Hence $x_1 + x_2 < 1$, and we have at least $q = \frac{\ell-1}{2}$ blocking pairs. $\square$

Theorem 1 ([5]). Blocking Pairs *is* NP*-complete.*

Proof. Clearly, this problem is in NP. To prove NP-completeness, we will reduce from Maximum Independent Set (MIS) which tests if a graph G contains an *independent set* of size at least k, Ie., a set U (with $|U| \geq k$) such that there is no edge in G between any two vertices in U. Garey, Johnson and Stockmeyer [13] show that the MIS problem is already NP-complete for the class of *3-regular* connected graphs, Ie., graphs, in which all vertices are of degree three. So we assume that G is 3-regular and connected. We may without loss of generality then assume that $k \leq \frac{1}{2}n$; as otherwise G does not have an independent set of size k. We also assume that we have unit edge weights, as variable weights are then a generalization.

From $G = (V,\ E)$ we construct a new graph G^* by introducing a set Y of np new vertices for some integer p, the value of which we will determine later. We denote the vertices in Y by $y_1^u, \cdots, y_p^u$ for each $u \in V$. We connect each y_i^u to its associated vertex u. All the original V vertices now have degree $3+p$, and all the vertices of Y have degree exactly one. Next, let K be a complete graph on ℓ vertices, where ℓ is some odd integer larger than np, the value again of which will be made clear later. We add $2(n-k)$ copies $K^1, \cdots, K^{2(n-k)}$ of K to G^* without introducing any further edges. The resulting graph consists of $2(n-k)+1$ components and is denoted by $G' = (V',\ E')$. We note that $\{uy_1^u \ : \ u \in V\}$ is a maximum matching in G^* of size n. Because of this and because ℓ is odd, we obtain that $\nu(G') = \frac{1}{2}(\ell-1)2(n-k)+n = \ell(n-k)+k$.

We show that the following statements are equivalent for suitable choices of ℓ and p, proving the Theorem.

(i) G has an independent set U of size $|U| \geq k$

(ii) $|B(x)| \leq (n-k)p + \frac{3}{2}n - 3k$ for some allocation x on G'.

(i) $\Rightarrow$ (ii): Suppose G has an independent set U of size $|U| \geq k$. We define an allocation x as follows, $x \equiv \frac{1}{2}$ on each $K^h, x \equiv 1$ on U' for some subset $U' \subseteq U$ of size $|U'| = k$ and $x \equiv 0$ otherwise. Then the blocking set is

$$\{(u, y_i^u) \ : \ u \in V \setminus U', 1 \leq i \leq p\} \cup \{(u,v) \ : \ u,v \in V \setminus U' \text{ and } uv \in E\}.$$

We observe that $|\{(u, y_i^u) \ : \ u \in V \setminus U', 1 \leq i \leq p\}| = (n-k)p$. Furthermore, because G is 3-regular, $U' \subseteq U$ is an independent set and $k \leq \frac{1}{2}n$, we find that $|\{(u,v) \ : \ u,v \in V \setminus U'\}| = |E| - 3k = \frac{3}{2}n - 3k \geq 0$. Hence, $|B(x)| = (n-k)p + \frac{3}{2}n - 3k$.

(ii) $\Rightarrow$ (i): Suppose $|B(x)| \leq (n-k)p + \frac{3}{2}n - 3k$ for some allocation x on G'. We may without loss of generality assume that x has a minimum number of blocking pairs. We first prove a number of claims.

Claim 1. *We may without loss of generality assume that $x_i \leq 1$ for all $i \in V'$.*

Proof. We prove Claim 1 as follows. Suppose $x_i = 1 + \alpha$ for some $\alpha > 0$. We set $x_i := 1$ and redistribute α over all vertices $j \in V' \setminus Y$ with $x_j < 1$. We ensure while redistributing that we do not add more than $1 - x_j$ to any x_j. The resulting allocation would have an equal or smaller number of blocking pairs. □

Claim 2. *We may without loss of generality assume that $x_y = 0$ for each $y \in Y$.*

Proof. We prove Claim 2 as follows. Suppose $x_y > 0$ for some $y \in Y$. Let u be the unique neighbour of y. We set $x_y := 0$ and $x_u := \min\{x_u + x_y, 1\}$. If necessary we redistribute the remainder over $V \cup \bigcup_j K^j$ without violating Claim 1. This is possible since $x(N) = \ell(n-k) + k < 2\ell(n-k) + n = |V| + |\bigcup_j K^j|$. The resulting allocation would have an equal or smaller number of blocking pairs. □

Claim 3. $x(\bigcup_j K^j) = \ell(n-k)$.

Proof. We prove Claim 3 as follows. First suppose $x(\bigcup_j K^j) > \ell(n-k)$. Then we set $x_i := \frac{1}{2}$ for each $i \in \bigcup_j K^j$ and redistribute the remainder over V without violating Claim 1. This is possible, since after setting $x_i := \frac{1}{2}$ for each $i \in \bigcup_j K^j$, we have $x(V') - x(\bigcup_j K^j) = \ell(n-k) + k - \ell(n-k) = k \leq n$. The resulting allocation would have a smaller or equal number of blocking pairs. Hence we may assume that $x(\bigcup_j K^j) \leq \ell(n-k)$ holds.

Suppose $x(\bigcup_j K^j) < \ell(n-k)$. Then there is some K^j with $x(K^j) < \frac{\ell}{2}$. By our earlier Lemma, there are at least $\frac{\ell-1}{2}$ blocking pairs in K^j. We choose $\ell \geq 2np + 2|E| + 2$ and obtain $|B(x)| \geq \frac{l-1}{2} > (n-k)p + |E|$. However, let x^* be given by $x^* \equiv \frac{1}{2}$ on $\bigcup_j K^j$, $x^* \equiv 0$ on Y, $x^* \equiv 1$ on some $U \subset V$ of size $|U| = k$ and $x^* \equiv 0$ on $V \setminus U$. Then x^* is an allocation as $x^*_i \geq 0$ for all $i \in V'$, and $x^*(V') = \ell(n-k) + k = \nu(G')$. We observe that $|B(x^*)| < (n-k)p + |E|$. Hence x is not an allocation with the minimum number of blocking pairs. This proves Claim 3. □

We now continue with the proof. Combining Claims 2 and 3 leads to

$$x(V) = x(V') - x(\bigcup_{j=1}^{2(n-k)} K^j) - x(Y) = \nu(G') - \ell(n-k) = \ell(n-k) + k - \ell(n-k) = k.$$

Let R be the set of vertices v in G with $x_v < 1$. We first show that $|R| \leq n-k$. Suppose $|R| \geq n-k+1$. Since $x(Y) = 0$ due to Claim 2, we find that $(n-k)p + \frac{3}{2}n - 3k \geq |B(x)| \geq |R|p \geq (n-k)p + p$. This is not possible if we choose $p = 2n$. Hence, indeed $|R| \leq n-k$ holds.

Let U consist of all vertices $u \in V$ with $x_u = 1$. Note that $U = V \setminus R$ due to Claim 1. Since $x(V) = k$ as we deduced above, we find that $|U| \leq k$ and thus $|R| \geq n-k$. As we already know that $|R| \leq n-k$, we find that $|R| = n-k$, and consequently, $|U| = k$. The latter equality implies that $x_v = 0$ for all $v \in R$.

Now because G is 3-regular, G has $\frac{3n}{2}$ edges. Then $|B(x)| \geq (n-k)p + \frac{3n}{2} - 3|U|$, with equality only if U is an independent set. Equality must hold since we assume that $B(x) \leq (n-k)p + \frac{3n}{2} - 3k$ and $|U| = k$. Hence, indeed U is an independent set of size k. □

We note that while the reduction used in Theorem 1 shows NP-completeness, it is not approximation preserving. Suppose we construct the expanded graph $G' = (V',\ E')$ as we did in the reduction from our input graph $G = (V,\ E)$ by adding vertex set Y and $2(n-k)$ copies of K_ℓ. Consider the 4-approximation algorithm for the Blocking Pairs problem on G', where we simply return the edge set:

$$B_{approx} = \Big\{(u,v)\ :\ (u,v) \in E\} \cup \{(u, y_i^u)\ :\ u \in V,\ 1 \leq i \leq p\}\Big\}$$

This is a 4-approximation for any general 3-regular graph G, where the allocation producing an optimal blocking set is x^*, as

$$|B_{approx}| = np + \frac{3n}{2} \leq 2np \leq 4(n-k)p + 6n - 12k = 4 * |B(x^*)|$$

Now suppose we pick G to have an independent set of size $\varepsilon > 0$. We can not find an approximation for Maximum Independent Set using the approximation algorithm for Blocking Pairs as we have no information about the structure of the graph from B_{approx}.

4.2 Balanced Bargaining Solutions from Blocking Sets

In this section we discuss the motivation for studying Blocking Sets. We explain how removing a blocking set from a graph G leaves us with a subgraph in which a balanced bargaining allocation exists.

Recall from Section 2.4 the definition of *power* of player i with respect to player j:

$$s_{ij}(x) = \max_{S \subset N} \Big\{ \nu(S) - x(S)\ :\ i \in S, j \notin S \Big\}$$

The prekernel is defined as the set of allocations where for every pair $i, j \in N$, $s_{ij}(x) = s_{ji}(x)$. Bateni et al. [4] showed that any allocation x in both the prekernel and the core is a balanced outcome for the bargaining game in general graphs. We use this result with blocking sets to find a balanced allocation. Let $G_O = (V_O,\ E_O)$ be a graph with an empty core, and let $N = V_O$ for our discussion. We begin with a blocking set and allocation pair (B, x) for G_O, where by definition x has the property that $x_u + x_v \geq w_{uv}$ for all $(u, v) \in \{E_O \setminus B\}$. We let the graph $G_B = (V_O,\ E_O \setminus B)$. For the bargaining game on G_B, x may not be a core member as $x(V) = \nu(G_B)$ may not be true. We relax our definition of a stable outcome to allow allocations with $x(V) = \nu(G)$ when $\nu(G) > \nu(G_B)$. Clearly, $\nu(G) - \nu(G_B) \leq |B|$, so picking a minimal blocking set is keeps the relaxation small. With this new definition, we call any allocation x on a graph $G_B = (V,\ E \setminus B)$ stable if $x_u + x_v \geq w_{uv}$ for all $(u, v) \in E \setminus B$.

We will proceed to compute a series of transfers between x variables that will converge to a balanced allocation x^*, with $(\alpha_u - x^*_u) = (\alpha_v - x^*_v)$ for all contracting pairs (u, v). Again, $x^*(V)$ may be greater than $\nu(G_B)$, but additionally the transfers may also result in pairs of players in a contract with combined money shares that are larger than the contract value. We relax our definition of a balanced outcome to allow $x^*_u + x^*_v > w_{uv}$ for players (u, v) in a deal, as we reassign extra surplus gained from the blocked edges. Again, the total excess surplus divided between the agents is at most $w(B)$, so picking a minimal blocking set is aimed at keeping the relaxation small. With this relaxed definition, we call an allocation x^* balanced if every contract is split according to the Nash Bargaining Solution, that is $(\alpha_u - x^*_u) = (\alpha_v - x^*_v)$ for all contracting agents (u, v).

For the remainder of this section, we refer to G_B simply as $G = (V,\ E)$. Since x satisfies the condition $x_u + x_v \geq w_{uv}$ for all edges in E, for all subsets of vertices $S \subseteq V$, we claim that $x(S) \geq \nu(S)$. This is because the edges in the maximum matching on S can each provide exactly w_{uv} to $\nu(S)$. The endpoints of (u, v) provide at least w_{uv} to $x(S)$, proving the claim. We will show that we do not violate this condition with any of our transfers to maintain a notion of stability.

Next, we consider the power of agent i with respect to agent j. Similar to the result by Bateni [4], we will show that we can simplify $s_{ij}(x)$ as follows: $s_{ij}(x) = \max\big\{-x_i,\ \max_{k\,:\,(i,k)\in E,\ k\neq j}\{w_{ik} - x_i - x_k\}\big\}$. Note that if $s_{ij}(x) = -x_i$, it implies that $\alpha_i = 0$.

Suppose that x, for any pair $(i, j) \in V$, satisfies the property that $s_{ij}(x) = s_{ji}(x)$. Then, using the simplified formula for $s_{ij}(x)$, $s_{ij}(x) = s_{ji}(x)$ implies that $(\alpha_i - x_i) = (\alpha_j - x_j)$ for all contracting pairs (i, j). So x is a balanced bargaining allocation. We focus on finding such an allocation starting from the aforementioned blocking set and allocation pair, (B, x).

For the remainder of this section, let $s_{ij}(x) \leq s_{ji}(x)$ for all allocations x and all ordered pairs of players $(i, j) \in V$. We begin by assuming there exists some pair (i, j) such that $s_{ij}(x) = s_{ji}(x) - 2\mu$, as otherwise no transfers are needed and x is balanced. Let x' be an allocation defined by

$$x'_v = \begin{cases} x_v - \mu & : \ \text{if } v = i \\ x_v + \mu & : \ \text{if } v = j \\ x_v & : \ \text{otherwise} \end{cases}$$

We say that the allocation x' arises from x by a *transfer from i to j (of size μ)* or simply an (i, j)-transfer.

We say that the coalition $S \subseteq V$ *separates* the ordered pair (i, j) of vertices if $i \in S$ and $j \in \{V \setminus S\}$. Given the allocation x, the coalition $S \subseteq V$ is called (i, j, x)-*maximal* if S separates (i, j) and $s_{ij}(x) = \nu(S) - x(S)$. That is, S is the coalition that maximizes the equation for $s_{ij}(x)$.

We now prove multiple Lemmas, slightly modifying the known results to apply them to our setting.

Lemma 4.2 ([12]). *Let the allocation $x' \in \mathbb{R}^V$ arise from x by an (i, j)-transfer of size $\mu > 0$. Then*

(i) $s_{kl}(x) - \mu \leq s_{kl}(x') \leq s_{kl}(x) + \mu$ *for all pairs* $(k, l) \in V$

(ii) $s_{ij}(x') = s_{ji}(x') = s_{ij}(x) + \mu$

Proof. Consider some coalition $S \subset N$. Then

$$x'(S) = \begin{cases} x(S) - \mu & : \text{ if } S \text{ is } (i,j)\text{-separating} \\ x(S) + \mu & : \text{ if } S \text{ is } (j,i)\text{-separating} \\ x(S) & : \text{ otherwise} \end{cases}$$

Now let S be (k, l, x)-maximal for some pair $(k, l) \in V$. Then

$$s_{kl}(x) = \nu(S) - x(S) \leq \nu(S) - x'(S) + \mu \leq s_{kl}(x') + \mu$$

which implies that $s_{kl}(x) - \mu \leq s_{kl}(x')$.

Now similarly, assume that S is (k, l, x')-maximal. Then

$$s_{kl}(x') = \nu(S) - x'(S) \leq \nu(S) - x(S) + \mu \leq s_{kl}(x) + \mu$$

which proves part (i).

To prove part (ii) of the lemma, note first that the subset S' that is (i, j, x)-maximal is also (i, j, x')-maximal. This is because for every subset S' that is (i, j)-separating, $x(S')$ increases by μ. So all subsets that can be considered to be the (i, j, x')-maximal subset increase by the same amount.

Let S_i and S_j be the (i, j, x)-maximal and (j, i, x)-maximal subsets respectively. Then

$$s_{ij}(x) = \nu(S_i) - x(S_i) = \nu(S_i) - (x'(S_i) + \mu) = s_{ij}(x') - \mu$$

and

$$s_{ji}(x) = \nu(S_j) - x(S_j) = \nu(S_j) - (x'(S_j) - \mu) = s_{ji}(x') + \mu$$

implying that $s_{ij}(x') = s_{ji}(x') = s_{ij}(x) + \mu$ as required since $s_{ij}(x) = s_{ji}(x) - 2\mu$. ∎

Lemma 4.3. *Let x be an allocation such that for every $(u, v) \in E$, $x_u + x_v \geq w_{uv}$. Let x' be an allocation that arises from x by an (i, j)-transfer of size μ. Then for every $(u, v) \in E$, $x'_u + x'_v \geq w_{uv}$.*

Proof. Suppose for contradiction there exists some $(i, k) \in E$ such that $x'_i + x'_k < w_{ik}$. Note that since x'_i is the only variable that decreased during the transfer, vertex i must be adjacent to the under-satisfied edge. Consider the coalition $S = \{i, k\}$, where $\nu(S) - x'(S) > 0$ since $x'_i + x'_k < w_{ik}$. Also, note that S is an (i, j)-separating coalition, so by definition, $s_{ij}(x') \geq \nu(S) - x'(S) > 0$. Then by Lemma 4.2, $s_{ij}(x') = s_{ji}(x') > 0$. However $0 \geq s_{ji}(x) > s_{ji}(x')$, by definition of a transfer, which is a contradiction. ☐

Lemma 4.4. *Let (B, x) be a blocking set and allocation pair for an unstable graph $G = (V,\ E)$, such that for all $(u, v) \in E \setminus B$, $x_u + x_v \geq w_{uv}$. Let x' be an allocation that arises by some number of transfers. The simplified formula $s_{ij}(x') = \max\big\{-x'_i,\ \max_{k\,:\,(i,k)\in E\setminus B,\, k\neq j}\{w_{ik} - x'_i - x'_k\}\big\}$, can be used for computing the power values of x' for all pairs $(i, j) \in V$.*

Proof. Let $m := \max\big\{-x'_i,\ \max_{k\,:\,(i,k)\in E\setminus B,\, k\neq j}\{w_{ik} - x'_i - x'_k\}\big\}$. Suppose for contradiction that there is some (i, j)-separating coalition S, such that $\nu(S) - x'(S) > m$. Let M_S be any maximum matching induced by S.

If i is not matched in M_S, we claim that $-x_i \geq \nu(S) - x'(S)$. First, if we remove the contribution of the endpoints of the edges $(u, v) \in M_S$ from the formula, since $x'_u + x'_v \geq w_{uv}$ by Lemma 4.3, $0 - x'(S \setminus \{u, v\ :\ (u, v) \in M_S\}) \geq \nu(S) - x'(S)$. Next, remove all vertices other than i from $\{S \setminus M_S\}$. Again this does not decrease the value, and in fact gives us our desired inequality. That is:

$$\nu(S) - x'(S) \leq 0 - x'(S \setminus M_S) \leq -x'_i$$

Since $m \geq -x'_i$ by definition, we have a contradiction.

Next assume that i is matched to some vertex $\ell \neq j$ in M_S. Remove all endpoints of the edges $\{M_S \setminus (i, \ell)\}$, Ie. all matching edges except (i, ℓ). Lemma 4.3 implies that $w_{i\ell} - x'(S \setminus M_S \cup \{i, \ell\}) \geq \nu(S) - x'(S)$. Next, remove all vertices that are not i and ℓ from S. Since none of these vertices are now matched, they do not contribute anything to the ν term of the formula. So this gives us our desired inequality:

$$\nu(S) - x'(S) \leq w_{i\ell} - x'(S \setminus M_S \cup \{i, \ell\}) \leq w_{i\ell} - x'_i - x'_\ell$$

which by definition is at most m, proving the Lemma. ☐

Lemma 4.5 ([12])**.** *Let x' be an allocation that arises from x by an (i, j)-transfer of size μ. Let (k, l) be a pair of players such that $s_{kl}(x) > s_{ji}(x)$. Then $s_{kl}(x') = s_{kl}(x)$.*

Proof. Let the endpoints of $e \in E$ be (k,l,x)-maximal. Then the given hypothesis

$$s_{kl}(x) > s_{ji}(x) > s_{ij}(x)$$

implies that e does not have either i or j as an endpoint (as otherwise it would be (i,j) or (j,i)-separating). So by definition we have that $x(e) = x'(e)$. The argument if $\{k\}$ is the (k,l,x)-maximal coalition is identical, which proves the lemma. $\square$

For an allocation $x \in \mathbb{R}^V$, recall that $s_{ij}(x) \leq s_{ji}(x)$ for all ordered pairs of players $(i,j) \in V$. Then, let

$$s(x) := \begin{cases} -\infty & : \text{ if } s_{kl}(x) = s_{lk}(x) \quad \forall\,(l,k) \in E \\ \max\{s_{ji}(x) \ : \ s_{ji}(x) > s_{ij}(x)\} & : \text{ otherwise} \end{cases}$$

Lemma 4.6 ([12])**.** *Let x' be an allocation that arises from x by an (i,j)-transfer of size μ. Then*

(i) $s_{kl}(x') > s_{kl}(x)$ *implies* $s_{kl}(x') \leq s_{ji}(x') = s_{ji}(x) - \mu$

(ii) $s(x') \leq s(x)$

Proof. To prove part (i), assume that $s_{kl}(x) < s_{kl}(x')$ for some $(k,l) \in E$. Let $S \subseteq N$ be (k,l,x')-maximal. Then S is (i,j)-separating since s_{kl} decreased as a result of the transfer, and so,

$$s_{kl}(x') \leq s_{ij}(x') = s_{ji}(x') = s_{ji}(x) - \mu$$

Part (ii) is a direct consequence of (i). $\square$

Let an (i,j)-transfer from an allocation x be called *canonical* if $s_{ji}(x) = s(x)$.

Lemma 4.7 ([12])**.** *Given an allocation x, we obtain an allocation $x^{(t)}$ satisfying $s(x^{(t)}) < s(x)$ after $t \leq |E|$ canonical transfers.*

Proof. Consider the set $I(x) := \{(k,l) \ : \ s_{kl}(x) = s(x)\}$ of pairs of players. Let x' be an allocation that arises from x by a (i,j) canonical transfer of size μ. Suppose $s(x') = s(x)$. Then $(i,j) \notin I(x')$. Let a new pair $(k,l) \notin I(x)$ enter $I(x')$. If $s_{kl}(x) > s(x)$, then $s_{kl}(x') = s_{kl}(x)$ by Lemma 4.5, and we have a contradiction. If $s_{kl}(x) < s_{ji}(x) = s(x)$, then $s_{kl}(x') > s_{kl}(x)$, and so by Lemma 4.6, $s_{kl}(x') \leq s_{ji}(x) - \mu < s(x) = s(x')$, a contradiction. So no new pair enters $I(x')$, which implies that $|I(x')| \leq |I(x)| - 1$, so after at most $t = |E|$ canonical transfers, $x^{(t)}$ will be an allocation with $s(x^{(t)}) < s(x)$. $\square$

After finding the allocation computed from Lemma 4.7, we could continue to compute a sequence of transfers which would strictly decrease $s(x^{(t)})$. However, it is not clear whether this sequence converges to a point in the prekernel. Instead we follow an idea of Faigle et al. [12] using linear programming.

We order the $s_{ij}(x)$ values in non-increasing order:

$$s_{i_1j_1}(x) \geq s_{i_2j_2}(x) \geq \cdots \geq s_{i_mj_m}(x) \quad \text{for } m \leq 2 \cdot |E|$$

Let $\mathcal{S}_x = \big\{(i,j) \ : \ s_{ij}(x) > s(x)\big\}$, and let $\Delta_x = \min_{(i,j)\in\mathcal{S}_x} s_{ij}(x)$. Also, let e_{ij} be the (i,j,x)-maximal edge. In the special case when $s_{ij}(x) = -x_i$, let $e_{ij} = \{i\}$. Then consider the following linear program, where x is a stable allocation:

$$
\begin{array}{llrcll}
(P_x) & \text{Max} & \delta & & & \\
& \text{subject to} & y(V) & = & \nu(V) & \qquad (4.1)\\
& & y(e_{ij}) & = & x(e_{ij}) \ \ \forall\,(i,j)\in\mathcal{S}_x & \qquad (4.2)\\
& & \nu(e_{ij}) - y(e_{ij}) & \geq & \nu(e) - y(e) \ \ \forall\,(i,j)\in\mathcal{S}_x,\ e\in\delta(i)\setminus(i,j) & \qquad (4.3)\\
& & w_e - y(e) & \leq & \Delta_x - \delta \ \ \forall\,(i,j)\notin\mathcal{S}_x,\ e\in\delta(i)\setminus(i,j) & \qquad (4.4)\\
& & -y_i & \leq & \Delta_x - \delta \ \ \forall\,(i,j)\notin\mathcal{S}_x & \qquad (4.5)\\
& & y_i + y_j & \geq & w_{ij} \ \ \forall\,(i,j)\in E & \qquad (4.6)
\end{array}
$$

We prove two final lemmas which will directly imply the main contribution of this section.

Lemma 4.8. *Let x be a solution to the bargaining game on $G = (V,\ E)$ such that $x_u + x_v \geq w_{uv}$ for $(u,v) \in E$. Consider a sequence of canonical transfers starting from x:*

$$\big\{x = x^0, x^1, \cdots x^q\big\}$$

such that $\mathcal{S}_{x^q} = \mathcal{S}_x$. Then x^q is a feasible solution to (P_x) for some $\delta > 0$.

Proof. The argument in the proof of Lemma 4.7 implies that

$$s(x) = s(x^0) \geq s(x^1) \geq \cdots \geq s(x^q)$$

We will proceed using induction on p, the number of transfers we have done since starting from x. Clearly x^0 is a feasible solution for (P_x) for some $\delta > 0$. Suppose for induction then that x^p is feasible for $p \geq 0$. Suppose further that x^{p+1} is an allocation that arises from x^p by an (i,j)-transfer of size μ. Then

$$x_v^{p+1} = \begin{cases} x_v^p - \mu & : \ \text{if } v = i \\ x_v^p + \mu & : \ \text{if } v = j \\ x_v^p & : \ \text{otherwise} \end{cases}$$

Then, constraint (4.1) clearly holds. For constraint (4.2), since $\mathcal{S}_x = \mathcal{S}_{x^p}$, it follows from Lemma 4.5 that

$$x^{p+1}(e_{ij}) = x^p(e_{ij}) = x(e_{ij})$$

where the last equality holds due to induction.

Suppose constraint (4.3) is not satisfied by x^{p+1}. Consider some $(k,\ell) \in \mathcal{S}_{x^{p+1}} = \mathcal{S}_x$, and let $e \in \delta(k) \setminus (k,\ell)$ be such that $\nu(e_{k\ell}) - x^{p+1}(e_{k\ell}) < \nu(e) - x^{p+1}(e)$. By Lemma 4.6 we know that

$$s_{k\ell}(x^{p+1}) \leq s_{k\ell}(x^p) = \nu(e_{k\ell}) - x^p(e_{k\ell})$$

where the equality is by induction since x^p is feasible. Then:

$$\nu(e_{k\ell}) - x^{p+1}(e_{k\ell}) < \nu(e) - x^{p+1}(e) \leq s_{k\ell}(x^{p+1}) \leq s_{k\ell}(x^p) = \nu(e_{k\ell}) - x^p(e_{k\ell})$$

implying that $x^{p+1}(e_{k\ell}) > x^p(e_{k\ell})$, and so j must be an endpoint of $e_{k\ell}$, but i is not an endpoint, Ie. $e_{k\ell}$ is an (i,j)-separating edge. Then since $(i,j) \notin \mathcal{S}_x$, the found inequality $\nu(e_{k\ell}) - x^{p+1}(e_{k\ell}) > \nu(e_{ij}) - x^{p+1}(e_{ij})$ contradicts the definition of $s_{ij}(x^{p+1})$.

Next consider constraint (4.4). Since $s(x^{p+1}) \leq s(x^p)$ as mentioned, for $(i,j) \notin \mathcal{S}_x$ and $e \in \delta(i) \setminus \{(i,j)\}$, we have that

$$w_e - x^{p+1}(e) \;\leq\; s_{ij}(x^{p+1}) \;\leq\; s(x^{p+1}) \;\leq\; s(x^p) \;\leq\; \Delta_{x^p} - \delta \;=\; \Delta_x - \delta$$

where the last inequality is because $\mathcal{S}_x = \mathcal{S}_{x^p}$.

For constraint (4.5), the argument is similar where for $(i,j) \notin \mathcal{S}_x$, we have

$$-x_i^{p+1} \;=\; s_{ij}(x^{p+1}) \;\leq\; s(x^{p+1}) \;\leq\; s(x^p) \;\leq\; \Delta_{x^p} - \delta \;=\; \Delta_x - \delta$$

Finally, constraint (4.6) is satisfied directly by Lemma 4.3 since x^p was feasible, which proves the lemma. ☐

Lemma 4.9. *Let (x,δ) be an optimal solution to (P_x). Consider a series of allocations ending in x' resulting from canonical transfers. Then $s(x') < s(x)$ implies that $\mathcal{S}_x \subset \mathcal{S}_{x'}$.*

Proof. Since $s(x') \leq s(x)$, it is clear that $\mathcal{S}_x \subseteq \mathcal{S}_{x'}$. Suppose that $s(x') < s(x)$ and $\mathcal{S}_x = \mathcal{S}_{x'}$. This means that the transfers must have lowered every $s_{kl}(x)$ value satisfying $s_{kl}(x) = s(x)$. Note that x' is a feasible solution to (P_x) by Lemma 4.8, and let $\delta' = \Delta_{x'} - s(x')$. This is a feasible value for δ' by the definition of $s(x')$. Then:

$$(\Delta_x - \delta') \;=\; (\Delta_{x'} - \delta') \;=\; s(x') \;<\; s(x) \;\leq\; (\Delta_x - \delta)$$

where the first equality is since $\mathcal{S}_x = \mathcal{S}_{x'}$. This implies that $\delta' > \delta$, which contradicts the optimality of δ, and proves the lemma. ☐

We now provide the main contribution of this section: an algorithm for finding a balanced allocation for an unstable graph G_B, after removing a blocking set B.

Algorithm 4.10.
Input: Unstable graph $G = (V,\ E)$, and a blocking set and allocation pair (B, x)
Output: A balanced bargaining allocation on the graph $G_B = (V,\ E \setminus B)$

(i) Starting with x, perform canonical transfers until an allocation x' with $s(x') < s(x)$ is found.

(ii) Find and return a balanced bargaining solution:

  (a) If $s(x') = -\infty$ then output x' and stop.

  (b) Compute an optimal solution (x^*, δ^*) to the LP (P_x).

  (c) Starting with x^*, perform canonical transfers until an allocation x' with $s(x') < s(x^*)$ is found.

  (d) Go to Step (ii)(a).

Lemma 4.8 and Lemma 4.9 directly imply the correctness and polynomial running time of the algorithm.

Theorem 2. *For an unstable graph $G = (V,\ E)$, when given a blocking set and allocation pair (B, x), a balanced bargaining allocation can be found in polynomial time for the graph $G_B = (V,\ E \setminus B)$.*

Chapter 5

Finding Blocking Sets

In this chapter we explore different methods for finding cheap blocking sets. We describe a natural integer program for solving the problem, and show that the relaxation has a poor integrality gap for general graphs, even with unit edge weights. We explain Iterative Rounding, a method for finding approximations for problems with linear programming formulations. We use an iterative rounding based algorithm to show that the BLOCKING PAIRS problem has is $O(1)$-factor approximable if the graph is planar.

We proceed to present a dynamic programming algorithm for finding the optimal blocking set in polynomial time for graphs with bounded tree width, and unit edge weights: $w_{uv} = 1 \ \forall \ (u, v) \in E$.

5.1 Linear Programming (LP) Approach

In this section we present a natural mixed integer programming formulation of the BLOCKING PAIRS problem. We will then later use the LP-relaxation in rounding schemes to try and find an approximation algorithm.

Consider the following mixed integer programming formulation, where $\nu(G)$ is again the size of the maximum integer matching in a graph $G = (V,\ E)$.

$$
\begin{array}{llrclll}
(IP_{BS}) & \text{Min} & w^{\intercal} z & & & & \\
 & \text{subject to} & x_u + x_v + z_{uv} w_{uv} & \geq & w_{uv} & \forall \ (u, v) \in E & (5.1) \\
 & & \vec{1}^{\intercal} x & \leq & \nu(G) & & (5.2) \\
 & & x_u & \geq & 0 & \forall \ u \in V & \\
 & & z & \in & \{0, 1\} & & (5.3)
\end{array}
$$

Any feasible solution (x, z) to this program is a feasible blocking set and allocation pair in the graph $G = (V,\ E)$, where an edge (u, v) belongs to the blocking set B if

$z_{uv} = 1$. The objective function forces the program to find the minimum weight blocking set.

The following LP is the dual of the canonical LP relaxation, (LP_{BS}), of the above IP where (5.3) is replaced by $z \geq 0$. We do not need to upper bound z with a constraint since clearly no z_{uv} variable will be larger than 1 in an optimal solution. In the dual LP, y is an edge-vector of reals corresponding to the (5.1) edge constraints. γ is a real variable corresponding to constraint (5.2). The program is provided here for completeness, and used later.

$$
\begin{array}{llrcll}
(D_{BS}) & \text{Max} & w^{\intercal}y - \gamma \cdot \nu(G) & & & & \\
& \text{subject to} & y(\delta(v)) & \leq & \gamma & \forall\, v \in V & (5.4) \\
& & w_{uv}y_{uv} & \leq & w_{uv} & \forall (u,v) \in E & (5.5) \\
& & y & \geq & \vec{0} & &
\end{array}
$$

5.2 Iterative Algorithm

Many NP-hard problems have natural mixed integer programming formulations like the one mentioned in the previous section. These problems can sometimes be solved approximately using an iterative method. In this section, we will describe this method, then give an approximation algorithm for the Blocking Pairs problem on planar graphs with unit edge weights.

5.2.1 Iterative Rounding Model

Iterative Rounding was first used by Jain [16] to solve the NP-hard Generalized Steiner Network Problem. It has since been used for finding approximations to many intractable problems.

To apply the method, we first define a general problem along with its linear programming relaxation, and a starting instance I. We exactly solve the LP for I to obtain a solution x, and include elements with integral variable values in our eventual solution set. Then, based on some preset conditions on the LP solution x, we reduce the problem to a residual instance I_R of the general problem. The conditions on x are defined such that $\bar{x}$, the projection of x onto the residual instance, is a feasible solution to the general problem for I_R. Formally, $\bar{x} = (x)\big|_{I_R} = \{x_j \ : \ j \in I_R\}$. We recursively solve the problem for the residual instance, and return the optimal solution $\hat{x}$. Then since $\bar{x}$ was feasible for I_R, $\hat{x} \leq \bar{x}$, implying:

$$\{x \setminus \bar{x}\} + \hat{x} \ \leq \ \{x \setminus \bar{x}\} + \bar{x} \ = \ x$$

We return the combined solution $\{x \setminus \bar{x} \,\cup\, \hat{x}\}$ as our solution to the general problem for I.

There are two simple options for reducing an instance of the general problem to a residual instance, as specified in [19]

1. If there is a variable in the optimal extreme point solution that is set to a value of 1, then include the element in the integral solution.

2. If there is a variable in the optimal extreme point solution that is set to a value of 0, then remove the corresponding element.

Unfortunately, there exist many problems where the natural LP formulations are non-integral. For these we design approximation algorithms where for each instance we solve, we may also use an additional step:

Rounding: Fix a threshold $\alpha \geq 1$. If there is a variable x_i that has a value of at least $\frac{1}{\alpha}$ in the optimal extreme point solution then pick the corresponding element in the solution being constructed.

Using this rounding step allows us to find solutions within an α-ratio of the optimal integral solution. There are many extensions of the iterative method. One is to use a *Relaxation* step instead or alongside of rounding, where we allow some constraints to be violated by a certain fixed amount β. We refer the reader to [20] for further information.

5.2.2 Approximation Algorithm Structure for Blocking Pairs Problem

We outline our approximation algorithm here, and then define it formally. We input a graph $G = (V,\ E)$ with unit edge weights, and maximum matching of size $\nu(G)$. We first build a new bipartite graph $G' = (V',\ E')$ by doubling G, to help our analysis at a later step by having no odd length cycles. For every vertex $u \in V$, we add two vertices to G', u_1 and u_2. For every edge $(u, v) \in E$, we add two edges to G', (u_1, v_2) and (u_2, v_1). We let $\omega_{G'} = 2 \cdot \nu(G)$, and replace all instances of $\nu(G)$ with $\omega_{G'}$ in all instances of (IP_{BS}) and (D_{BS}). The analysis may be possible on non-bipartite graphs, but it is easier with the absence of odd length cycles.

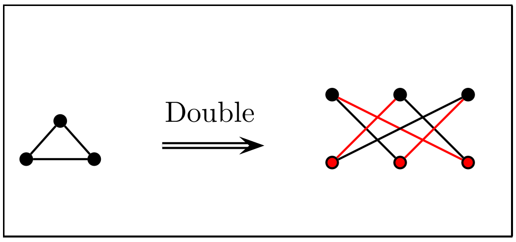

Figure 5.1: Creating G'

For the remainder of this section, let OPT_G be the optimal blocking set on the graph G.

Lemma 5.1. *Let $(\hat{x}', \hat{z}')$ define the optimal (integral) solution to the* Blocking Pairs *optimization problem on the expanded graph $G' = (V',\ E')$. Then we can find a feasible, integral solution (x, z) to (LP_{BS}) for G, defining a blocking set of at most twice the size OPT_G.*

Proof. First, suppose $(\hat{x}, \hat{z})$ is the optimal integral solution to (LP_{BS}) for the graph G that defines the blocking set OPT_G. Then let $x'_{u_1} = x'_{u_2} = \hat{x}_u$, and $z'_{u_1v_2} = z'_{u_2v_1} = \hat{z}_{uv}$. (x', z') is a feasible, integral solution to the LP for G'. It defines a blocking set of size at most $2 \cdot |OPT_G|$. So then, for $OPT_{G'}$,

$$|OPT_{G'}| \leq z'(E') \leq 2 \cdot |OPT_G|$$

Next, suppose we are given the integral LP solution $(\hat{x}', \hat{z}')$ that defines the optimal blocking set for G', $OPT_{G'}$. For every vertex $u \in V$, we set $x_u = \frac{\hat{x}'_{u_1} + \hat{x}'_{u_2}}{2}$. For every edge (u, v) in E, we set $z_{uv} = \frac{\hat{z}'_{u_1v_2} + \hat{z}'_{u_2v_1}}{2}$. If $z_{uv} = \frac{1}{2}$ for any $(u, v) \in E$, we round z_{uv} up to 1. Since $(\hat{x}', \hat{z}')$ is a feasible LP solution for G', for every $(u, v) \in E$, if we add the two corresponding edge constraints together and divide by two:

$$\begin{aligned} & \hat{x}'_{u_1} + \hat{x}'_{u_2} + \hat{x}'_{v_1} + \hat{x}'_{v_2} + \hat{z}'_{u_1v_2} + \hat{z}'_{u_2v_1} \geq 2 \\ \Rightarrow \quad & x_u + x_v + z_{uv} \geq 1 \quad \forall\, (u, v) \in E \end{aligned}$$

So (x, z) defines a feasible blocking set and allocation pair for G.

Since we may have rounded values of $z_{uv} = \frac{1}{2}$ up to 1, the blocking set defined by z will have size at most

$$2 \cdot \sum_{uv \in E} \frac{\hat{z}'_{u_1v_2} + \hat{z}'_{u_2v_1}}{2} = \sum_{u_iv_j \in E'} \hat{z}'_{u_iv_j} = |OPT_{G'}| \leq 2 \cdot |OPT_G|$$

□

From now on we assume that graphs G' are bipartite, and built by doubling the original graph G. We will find a solution for G once we have found a blocking set for G', by using the mapping described in Lemma 5.1.

Unfortunately, even on the bipartite graph G', the Blocking Pairs problem is not integral. Consider the example in Figure 5.2. The boxes with the text K_{1001} inside represent complete subgraphs of 1001 vertices (cliques). The boxes with $K^{-}_{1001,1001}$ in them represent complete bipartite graphs with bipartitions each of size 1001, minus the edges (u_1, u_2) for all $u \in V$. The vertices on the borders of the boxes are members of the respective cliques and bipartitions (coloured black and red).

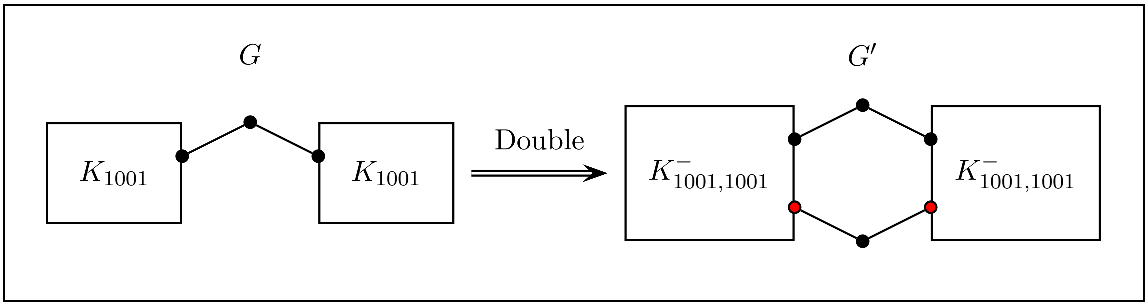

Figure 5.2: Bipartite Graph Structure where Blocking Pairs is non-integral

Lemma 5.2. *Let (LP_{BS}) be the canonical linear programming relaxation of (IP_{BS}). (LP_{BS}) may be non-integral for a bipartite graph G', built from doubling some general graph G.*

Proof. Consider the example in Figure 5.2. To minimize the blocking set on G', it is clear we will assign $x_u = \frac{1}{2}$ to all the vertices u in the bipartite $K^-_{1001,1001}$ subgraphs. Then we have distributed all of the allowed $\omega_{G'} = 2 \cdot \nu(G) = 2002$. This means we will need $z_{uv} = \frac{1}{2}$ values on the four (u, v) edges that are not part of the two $K^-_{1001,1001}$ subgraphs. This solution has an objective value of 2.

We show that this allocation is the optimal solution to (LP_{BS}) by building a dual solution of equal size. First, assign $\gamma = 2$. Then assign $y_{uv} = 1$ for the four (u, v) edges that are not part of the two $K^-_{1001,1001}$ subgraphs. Let E_O be the set of these four edges. Also assign $y_{uv} = 1$ for 6 extra edges that create a 10-cycle when combined with E_O. Consider now two $K^-_{999,999}$ subgraphs, formed from the vertices of the $K^-_{1001,1001}$ subgraphs, but excluding the four *boundary* vertices, and the four extra vertices used in the 10-cycle. Within each of these subgraphs, find two edge-disjoint perfect matchings of size 999 each. Assign $y_{uv} = 1$ for all edges in these four matchings. This is possible since the subgraphs are complete bipartite graphs, minus (u_1, u_2) edges for all $u \in V$.

This solution, (y, γ), has an objective value of $\vec{1}^{\intercal} y - \gamma \cdot \omega_{G'} = 4006 - 2 \cdot 2002 = 2$, implying that (x, z) is an optimal, non-integral solution. $\square$

We apply iterative rounding techniques mentioned in the previous section to a more general version of the blocking pairs problem to be able to formulate the arising residual subproblems. We partition the edge set $E = (E_1 \cup E_2)$, and find a blocking set $B \subseteq E_1$. We call E_2 the set of forbidden edges, as they cannot be a part of any blocking set.

Generalized Blocking Set
Instance: A graph $G = (V,\ E_1,\ E_2)$, a value $\nu(G)$, and an integer k
Question: Does G allow an allocation x with $|B(x)| \leq k$, where $B(x) \subseteq E_1$

$E_1 \cup E_2$ forms the complete edge set of the graph G. This problem can be formulated with an IP similar to (IP_{BS}). We state its canonical linear programming relaxation here for the unit weight case.

$$
\begin{array}{llrclll}
(LP_{GBS}) & \text{Min} & \vec{1}^\intercal z' & & & & \\
& \text{subject to} & x'_u + x'_v + z'_{uv} & \geq & 1 & \forall\, (u,v) \in E_1 & (5.6) \\
& & x'_u + x'_v & \geq & 1 & \forall\, (u,v) \in E_2 & (5.7) \\
& & \vec{1}^\intercal x' & \leq & \nu(G) & & (5.8) \\
& & x'_u & \geq & 0 & \forall\, u \in V & \\
& & z' & \geq & 0 & &
\end{array}
$$

We double G, and solve (LP_{GBS}) for $G^i = (V',\ E',\ \emptyset)$, starting with $i = 1$, to get a solution (x^i, z^i). Note that E_2 starts empty and may grow. We then reduce the problem to a residual instance on a graph G^{i+1}:

1. If there is a vertex u with $x^i_u = 1$, $G^{i+1} = (V^i \setminus \{u\},\ E^i_1 \setminus \delta_{E^i_1}(u),\ E^i_2 \setminus \delta_{E^i_2}(u))$. This removes u and all its adjacent edges. Set $\omega_{G^{i+1}} = \omega_{G^i} - 1$.
2. If there is an edge (u, v) with $z^i_{uv} = 0$, $G^{i+1} = (V^i,\ E^i_1 \setminus \{(u, v)\},\ E^i_2 \cup \{(u, v)\})$.
3. If there is an edge (u, v) with $z^i_{uv} \geq \frac{1}{3}$, then $G^{i+1} = (V^i,\ E^i_1 \setminus (u, v),\ E^i_2)$.

We continue to set variables and solve residual instances on G^i graphs using (LP_{GBS}) until we have removed all edges from E^i_1. We map the built solution back to a solution on the original graph G, as was described in Lemma 5.1, and return the resulting blocking set.

Algorithm 5.3.
Input: Graph $G = (V,\ E)$, $w = \vec{1}$
Output: A blocking set B for the graph G of size $\leq 6 \cdot |OPT_G|$, where OPT_G is the optimal blocking set for the graph G.

(i) Double G to get the bipartite graph $G' = (V',\ E')$:

  (a) For every vertex $u \in V$, add two vertices, u_1, u_2, to V'.
  (b) For every edge $(u, v) \in E$, add two edges, (u_1, v_2) and (u_2, v_1) to E'.
  (c) Set $\omega_{G'} = 2 \cdot \nu(G)$, and replace all instances of $\nu(G)$ with $\omega_{G'}$.
  (d) (x', z') are defined for every vertex and edge respectively in the graph G'.

(ii) Let the graph $G^1 = (V^1,\ E_1^1,\ E_2^1) = (V',\ E',\ \emptyset)$, and $\omega_{G^1} = \omega_{G'}$. Set $i = 1$.

(iii) While $E_1^i \neq \emptyset$

  (a) Find an optimal extreme point solution (x^i, z^i) to (LP_{GBS}) for graph G^i. Define a residual problem:
    (1) If there is a vertex $u \in V^i$ with $x_u^i = 1$, let $G^{i+1} = (V^i \setminus u,\ E_1^i \setminus \{(u, v)\ :\ v \in \delta(u)\},\ E_2^i \setminus \{(u, v)\ :\ v \in \delta(u)\})$. Also, $\omega_{G^{i+1}} = \omega_{G^i} - 1$.
    (2) Else if there is an edge (u, v) with $z_{uv}^i = 0$, then let $G^{i+1} = (V^i,\ E_1^i \setminus (u, v),\ E_2^i \cup (u, v))$. Set $\omega_{G^{i+1}} = \omega_{G^i}$.
    (3) Else if there is an edge (u, v) with $z_{uv}^i \geq \frac{1}{3}$ then let $G^{i+1} = (V^i,\ E_1^i \setminus (u, v),\ E_2^i)$. Set $\omega_{G^{i+1}} = \omega_{G^i}$.
  (b) Remove all singleton vertices $V_S = \{u \in V^{i+1} : \delta(u) = \emptyset\}$ from the residual problem, setting $x_u^i = 0$ for all $u \in V_S$, as they do not affect any (LP_{GBS}) constraints: $V^{i+1} = V^{i+1} \setminus V_S$.
  (c) $i = i + 1$.

(iv) Map the complete resulting (x^i, z^i) solution back to $(\hat{x}, \hat{z})$:

  (a) For every vertex $u \in V$, $\hat{x}_u = \frac{x_{u_1}^i + x_{u_2}^i}{2}$.
  (b) For every edge $(u, v) \in E$, $\hat{z}_{uv} = \frac{z_{u_1 v_2}^i + z_{u_2 v_1}^i}{2}$.

(v) Round up any edges with $\hat{z}$ values of $\frac{1}{2}$ to 1.

(vi) Return $B = \{(u, v) \in E\ :\ \hat{z}_{uv} = 1\}$.

We now analyze the correctness of the algorithm. For the remainder of this section, let $OPT_{LP}(G)$ be the optimal solution to (LP_{GBS}) for the graph $G = (V,\ E_1,\ E_2)$.

Claim 5.4. *If for some graph G, Algorithm 5.3, at every iteration, finds a vertex with $x'_u = 1$ in step (iii)(a)(1), or an edge (u,v) with $z'_{uv} = 0$ in step (iii)(a)(2), or an edge (u,v) with $z'_{uv} \geq \frac{1}{3}$ in step (iii)(a)(3), then it returns a blocking set of size at most $6 \cdot OPT_{LP}(G)$, and so at most $6 \cdot |OPT_G|$.*

Proof. The proof will proceed based on induction on the number of iterations of the algorithm, analyzing the return value from every iteration.

The base case is trivial if only one iteration of the algorithm is needed. We have an extreme point solution (x^1, z^1), where we round any $z^1_{uv} \geq \frac{1}{3}$ up to 1. We then map (x^1, z^1) to $(\hat{x}, \hat{z})$, possibly multiplying $\hat{z}$ values by 2 to obtain an integral solution, as described in Lemma 5.1. This gives us a blocking set of size no more than $6 \cdot OPT_{LP}(G)$.

Let $LP(G^j)$ be the instance of (LP_{GBS}) for the graph G^j at some iteration $j \geq 1$, and let $C_{LP(G^j)}$ be the constraint set of $LP(G^j)$. Also, let $(\bar{x}, \bar{z})$ be the projection of the solution (x^i, z^i) onto the residual instance with graph G^{i+1}. Recall that $\bar{x} = (x^i)\big|_{G^{i+1}} = \{x^i_u\ \forall\ u \in V^{i+1}\}$, and similarly, $\bar{z} = (z^i)\big|_{G^{i+1}} = \{z^i_{uv}\ \forall\ (u,v) \in E^{i+1}_1\}$. We split the remainder of the proof into three cases.

Case (a): Suppose at some iteration i, the optimal LP solution assigned some value $x^i_u = 1$. By step (iii)(a)(1) of Algorithm 5.3, a residual problem is built by removing the vertex u and all its adjacent edges from G^i. Since $x^i_u = 1$, we know that for any v in the neighbourhood of u, $z^i_{uv} = 0$, as otherwise we are contradicting optimality. Then, since (x^i, z^i) was feasible for $LP(G^i)$, and $C_{LP(G^{i+1})} \subset C_{LP(G^i)}$, $(\bar{x}, \bar{z})$ is a feasible solution to $LP(G^{i+1})$, the residual problem.

By induction, from iteration $i+1$, the algorithm returns a blocking set B^{i+1}. Then:

$$|B^{i+1}| \leq 6 \cdot OPT_{LP}(G^{i+1}) \leq 6 \cdot \sum_{uv \in E^{i+1}_1} \bar{z}_{uv} \leq 6 \cdot \sum_{uv \in E^i_1} z^i_{uv} = 6 \cdot OPT_{LP}(G^i)$$

Since $z^i_{uv} = 0$ for all $v \in \delta(u)$, then B^{i+1} is a feasible blocking set for G^i if we set $x_u = 1$, and $\vec{1}^\intercal x = \omega_{G^i}$ holds.

Case (b): Suppose at some iteration i, the optimal LP solution assigned some value $z^i_{uv} = 0$. By step (iii)(a)(2) of Algorithm 5.3, a residual problem is built by moving (u,v) from E_1 in G^i to E_2 in G^{i+1}. Again, $(\bar{x}, \bar{z})$ is a feasible solution to $LP(G^{i+1})$, since $C_{LP(G^{i+1})} = C_{LP(G^i)} \setminus \{x_u + x_v + z_{uv} \geq 1\} \cup \{x_u + x_v \geq 1\}$, and $x^i_u + x^i_v \geq 1$ as $z^i_{uv} = 0$.

By induction, from iteration $i+1$ the algorithm returns a blocking set B^{i+1}. Since $z^i_{uv} = 0$, the same inequalities from Case (a) apply, and B^{i+1} is a feasible blocking set for G^i, when we set $z^i_{uv} = 0$.

Case (c): Suppose at some iteration i, the optimal LP solution assigned some value $z^i_{uv} \geq \frac{1}{3}$. By step (iii)(a)(3) of Algorithm 5.3, a residual problem is built by removing (u, v) from E_1. Again, $(\bar{x}, \bar{z})$, the projection of the solution (x^i, z^i) onto the residual graph G^{i+1}, is feasible for $LP(G^{i+1})$ as again $C_{LP(G^{i+1})} \subset C_{LP(G^i)}$.

By induction, from iteration $i + 1$ the algorithm returns a blocking set B^{i+1} where:

$$|B^{i+1}|+1 \leq 6 \cdot OPT_{LP}(G^{i+1})+1 \leq 6 \cdot \sum_{uv \in E_1^{i+1}} \bar{z}_{uv} + 3 \cdot z^i_{uv} \leq 6 \cdot \sum_{uv \in E_1^i} z^i_{uv} = 6 \cdot OPT_{LP}(G^i)$$

Together, B^{i+1} and (u, v) form a feasible blocking set for G^i of the proper size. This proves the claim. □

Unfortunately, there are instances when Algorithm 5.3 gets stuck, as none of three conditions, step (iii) (a)(1), (2), and (3), are true. We show an example of such an instance in the proof of Lemma 5.5.

The following linear program is the dual of (LP_{GBS}), for graph G with unit edge weights, where y and a are E_1 and E_2 edge-vectors respectively, with real values. γ is a real variable corresponding to constraint (5.8). The program is written here using ν to remain consistent with the statement of (LP_{GBS}). We replace $\nu(G)$ with $\omega_{G'}$ in our analysis.

$$\begin{array}{llrcll}
(D_{GBS}) & \text{Max} & \vec{1}^{\intercal}y + \vec{1}^{\intercal}a - \gamma \cdot \nu(G) & & & \\
& \text{subject to} & y(\delta_{E_1}(u)) + a(\delta_{E_2}(u)) & \leq & \gamma & \forall\, u \in V \qquad (5.9) \\
& & y & \leq & \vec{1} & \\
& & y_{uv} & \geq & 0 & \forall\, (u,v) \in E_1 \\
& & a_{uv} & \geq & 0 & \forall\, (u,v) \in E_2
\end{array}$$

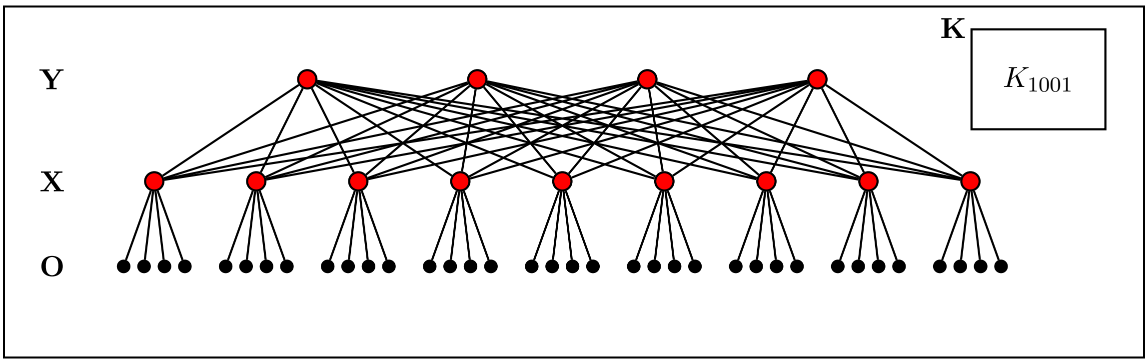

Figure 5.3: Example Structure for proof of Lemma 5.5

Lemma 5.5. *For general graphs, there are instances of the residual problem with optimal solutions, and so extreme points, where none of the three conditions, step (iii) (a)(1), (2), (3) in Algorithm 5.3, are true.*

Proof. Consider the example in Figure 5.3. Let $G = (V = \{X \cup Y \cup O \cup K\},\ E)$ where $|Y| = 4$ and $|X| = 9$. Let the vertices of K form a clique of size 1001. Let every vertex $u \in X$ be adjacent to exactly 4 vertices in O. Let $(u, v) \in E$ for every pair $u \in X$, $v \in Y$ (X and Y form a complete graph). Run Algorithm 5.3 on G. The sets X', Y', O' and K' are the copies of X, Y, O and K respectively in the doubled instance G', made in step (i). $\nu(G) = |X| + 500$, so $\omega_{G'} = 1018$. Clearly, the clique components will be covered by assigning $x_u = \frac{1}{2}$ for all $u \in K \cup K'$, as otherwise it will result in a very large LP solution. So the amount to distribute across the sets X, Y, and O, and their copies, is exactly 17.

Now consider the feasible solution (x, z) defined as:

$$x_u = \begin{cases} \frac{9}{10} & : \text{ if } u \in \{X \cup X'\} \\ \frac{1}{10} & : \text{ if } u \in \{Y \cup Y'\} \\ \frac{1}{2} & : \text{ if } u \in \{K \cup K'\} \\ 0 & : \text{ otherwise} \end{cases}$$

$$z_{uv} = \begin{cases} \frac{1}{10} & : \text{ if } u \in \{X \cup X'\}, v \in \{O \cup O'\} \\ 0 & : \text{ otherwise} \end{cases}$$

Then, our LP solution has an objective value of $4 \cdot (|X| + |X'|)(\frac{1}{10}) = \frac{36}{5}$.
Consider now the feasible dual solution (y, a, γ) defined as:

$$y_{uv} = \begin{cases} 1 & : \text{ if } u \in \{X \cup X'\}, v \in \{O \cup O'\} \\ 0 & : \text{ otherwise} \end{cases}$$

$$a_{uv} = \begin{cases} \frac{36}{5000} & : \text{ if } u, v \in \{K \cup K'\} \\ \frac{4}{5} & : \text{ if } u \in \{X \cup X'\}, v \in \{Y \cup Y'\} \\ 0 & : \text{ otherwise} \end{cases}$$

$$\gamma = \frac{36}{5}$$

This dual solution also has an objective value of $\frac{36}{5}$, which means it is an extreme point as it is equal to the objective value of a feasible primal solution.

Note that once we remove the z_{uv} variables for all (u,v) where $u \in X \cup X', v \in Y \cup Y'$ (by step (iii)(a)(2) of Algorithm 5.3), then $0 < z_{uv} \le \frac{1}{3}$ for all $(u,v) \in E_1$, and there is no vertex $u \in V$ such that $x_u = 1$. This proves the lemma.

□

In fact, using the structure of the example in the proof of Lemma 5.5, we show that (LP_{GBS}) can have an arbitrarily large integrality gap for general graphs.

Theorem 3. *For general graphs, (LP_{GBS}) has $O(V)$ integrality gap, even with unit edge weights.*

Proof. Consider the example in Figure 5.3, but set G to have $|Y| = n$ for some $n \ge 4$. Let $|X| = 2n+1$, and let the clique be K_{1001n}. After doubling, since for any optimal solution we have to assign $x^*_u = \frac{1}{2}$ to all u in the $K^-_{1001n,1001n}$ subgraph, we can assign a total value of $\omega_{G'} - 1001n = 4n+1$ to the remainder of the vertices. Let:

$$x^*_u = \begin{cases} \frac{2n+1}{2n+2} & : \text{ if } u \in \{X \cup X'\} \\ \frac{1}{2n+2} & : \text{ if } u \in \{Y \cup Y'\} \\ \frac{1}{2} & : \text{ if } u \in \{K \cup K'\} \\ 0 & : \text{ otherwise} \end{cases}$$

$$z^*_{uv} = \begin{cases} \frac{1}{2n+2} & : \text{ if } u \in \{X \cup X'\}, v \in \{Y \cup Y'\} \\ 0 & : \text{ otherwise} \end{cases}$$

The optimal solution to (LP_{GBS}) will then have an objective value of at most:

$$4 \cdot (|X| + |X'|) \cdot \frac{1}{(2n+2)} < 4 \cdot (4n+4) \cdot \frac{1}{(2n+2)} = 8$$

Now consider the best integral solution $(\bar{x}, \bar{z})$ to (LP_{GBS}). Because we have to distribute at least $1001n$ to the $K^-_{1001n,1001n}$ subgraph, there must be at least one vertex $u \in X \cup X'$ with value $\bar{x}_u = 0$. This vertex is adjacent to n vertices in $Y \cup Y'$. If we assign these vertices $\bar{x}$ values of 0, then we have a blocking set of size at least $4+n = O(V)$. If we assign $\bar{x} = 1$ values for some number $c \le n$ of these vertices, we must have at least c vertices in $X \cup X'$ with $\bar{x}$ values of 0. Then we have a blocking set of size at least $4(c+1) + (c+1)(n-c) = (4+n-c)(c+1)$. This value is minimized at the extreme values of c, either $c = 0$ or $c = n$. In both cases, it is at least $(4+n) = O(V)$, proving the Theorem. □

5.2.3 Approximation Algorithm for Blocking Pairs on Planar Graphs

Unfortunately, Theorem 3 states that Algorithm 5.3 does not provide an approximation for the Blocking Pairs problem in general graphs. However, this approach does provide benefits on planar graphs.

Suppose we have a planar graph $G = (V,\ E)$. Run Algorithm 5.3 on graph G. The algorithm will either finish, or get stuck at an iteration where none of step(iii)(a) (1), (2), or (3) are true. Suppose we get stuck at some iteration i, on the bipartite graph $G^i = (V^i,\ E_1^i,\ E_2^i)$. We find an extreme point solution (x^i, z^i) to (LP_{GBS}) on G^i. Since none of the (a)(1), (2), and (3) conditions are true, we know that for every $(u, v) \in E_1^i$, $0 < z_{uv}^i < \frac{1}{3}$. We also know that for all $u \in \{X \cup Y\}$, $x_u^i > 0$, since no $x_u = 1$.

We now state two Lemmas to help classify the vertices of V^i. We refer the reader to the referenced material for the proofs.

Lemma 5.6 (Ghouila-Houri [14])**.** *A matrix M is totally unimodular if and only if each subset $J \subseteq [n]$ of the columns of M can be partitioned into two classes J_1 and J_2 such that for each row $r \in [m]$ we have $|\sum_{j \in J_1} m_{ij} - \sum_{j \in J_2} m_{ij}| \leq 1$.*

Lemma 5.7 ([29])**.** *Let M be a totally unimodular matrix and b be an integral vector. Then the polyhedron $P = \{x \,:\, Mx \leq b\}$ is integral.*

Lemmas 5.6 and 5.7 help us prove a structural result.

Lemma 5.8. *For some specific $\alpha_{G^i} \leq 1$, for every $u \in V^i$, $x_u^i \in \{0, (1 - \alpha_{G^i}), \alpha_{G^i}, 1\}$. Also, for every $(u, v) \in E_1^i$, $z_{uv}^i \in \{0, (1 - \alpha_{G^i}), \alpha_{G^i}, 1\}$.*

Proof. Consider (LP_{GBS}) for G^i, but remove constraint (5.8). Call this new linear program (LP_{GBS}^-). Consider the constraint matrix of (LP_{GBS}^-), let it be M. Note that G^i is bipartite, and let $V^i = (A \cup B)$ be its bipartition. Note that every row of M has exactly 3 non-zero entries. We first show that M is totally-unimodular. Take a subset of the columns of M, $J \subseteq [n]$ as in Lemma 5.6. We partition the columns of J as follows. If column j corresponds to some $u \in A$, then include j in the set J_1. If j corresponds to some $u \in B$, include j in J_2. We then have to classify each column c_{uv}, corresponding to the variable z_{uv}. Note first that each such column has exactly 1 non-zero entry. Consider the row r_{uv} where the z_{uv} column is non-zero. Let c_u and c_v be the two other columns where r_{uv} is non-zero. If $c_u, c_v \in J$ or $c_u, c_v \notin J$ then include $c_{uv} \in J_1$. If only one of c_u and c_v are in J, include c_{uv} in the opposite set as the included column, c_u or c_v. Since $m_{ij} = 1$ for all non-zero entries row, this partition scheme for the columns in J shows that M is totally unimodular.

Then by Lemma 5.7, we know (LP_{GBS}^-) is integral. Suppose we now add constraint (5.8) back into the formation of M. Because our extreme point solution (x^i, z^i) is non-integral, it must mean that it satisfies constraint (5.8) with equality. This means the hyperplane defining the equality section of this constraint intersects the feasible region of (LP^{-}_{GBS}) at some point along a line between two extreme points of (LP^{-}_{GBS}). That is, for some $0 < \alpha_{G^i} < 1$, $(x^i, z^i) = \alpha_{G^i} \cdot (x^*_1, z^*_1) + (1 - \alpha_{G^i}) \cdot (x^*_2, z^*_2)$, where (x^*_l, z^*_l) for $l = 1, 2$ are the two mentioned extreme points of (LP^{-}_{GBS}). This proves the lemma. $\square$

To analyze the structure of G^i, we now classify every vertex $u \in V^i$ according to its x^i_u value. We note that because of step (iii)(a) (1), $x^i_u \neq 1$ for any $u \in V^i$.

Definition 5.9. We define three sets, O, X, and Y:

(i) Let $O = \{u \in V^i \ : \ x^i_u = 0\}$.

(ii) Let $X = \{u \in V^i \ : \ x^i_u = \alpha_{G^i}\}$

(iii) Let $Y = \{u \in V^i \ : \ x^i_u = (1 - \alpha_{G^i})\}$

Consider a neighbour v of some $u \in O$. Since for all $(u, v) \in E^i_1$, $0 < z^i_{uv} < \frac{1}{3}$, $x^i_v > \frac{2}{3}$. Without loss of generality, let $\alpha_{G^i} = x^i_v > \frac{2}{3}$. (As otherwise, just switch the labels of the sets X and Y).

We now prove three Lemmas to help us later.

Lemma 5.10. *$O \cup Y$ is an independent set in G^i.*

Proof. We first show that O and Y are individually independent sets.

Suppose that O is not an independent set, and let u, $v \in O$ such that (u, v) exists. Then, since $x^i_u = x^i_v = 0$, $z^i_{uv} = 1$, a contradiction.

Similarly, suppose that Y is not an independent set, and let u, $v \in Y$ such that (u, v) exists. Then $z^i_{uv} = 1 - 2 \cdot (1 - \alpha_{G^i}) = 2\alpha_{G^i} - 1 > \frac{1}{3}$, again a contradiction.

Finally, suppose there exists and edge (u, v) such that $u \in O$ and $v \in Y$. Then $z^i_{uv} = 1 - (1 - \alpha_{G^i}) > \frac{2}{3}$, a contradiction. This proves the lemma. $\square$

The assumption that $\alpha_{G^i} > \frac{2}{3}$ and Lemma 5.10 imply that the vertices of the set X are adjacent to all edges $E^i_1 \cup E^i_2$. Note also that no vertex $v \in O$ can be adjacent to an E^i_2 edge since no $x_u = 1$. Similarly, no vertex $v \in Y$ can be adjacent to an E^i_1 edge. Suppose it was adjacent to some $(u, v) \in E^i_1$, for some $u \in X$. Then $z^i_{uv} = 0$, which is a contradiction.

An extreme point of (LP_{GBS}) is uniquely determined by a set $\{c_1, c_2, \cdots\}$ of tight constraints of type (5.6) or (5.7), and the extra (5.8) constraint. The incidence vectors of the left hand side of these constraints are all linearly independent. We call two constraints linearly independent and linearly dependent if the incidence vectors of their left hand sides are linearly independent or dependent, respectively.

We now state the Rank Lemma which we will use to prove multiple Lemmas later. We refer the reader to the referenced material for the proof.

Lemma 5.11 (Rank Lemma [19]). *Let $P = \{x \ : \ Ax \geq b, \ x \geq 0\}$ and let x be an extreme point solution of P. Let $E = \{e \ : \ x_e > 0\}$ be the set of elements with non-zero variables. Then the size of any maximal set of linearly independent tight constraints equals $|I|$, the number of non-zero variables in the extreme point solution x.*

Consider the constraints of (LP_{GBS}) for G^i. First note that every (5.6) constraint is tight in all extreme point solutions. The set of all such constraints is linearly independent, as each individual $(u, v) \in E_1^i$ constraint is the only equation containing z_{uv}.

Let $G_T = G^i[X \cup Y]$ be the bipartite subgraph of G^i containing the E_2^i edges. Let T be a spanning tree for G_T. It is known that any connected graph on n vertices has a spanning tree containing exactly $n - 1$ edges. We will show later that G^i must be connected. It is also known that adding even a single edge to a spanning tree creates a cycle, in our case, an even cycle since G_T is bipartite. We note that even cycles of E_2^i edges define sets of linearly dependent constraints. To see this, let M_1 and M_2 be the two edge-disjoint perfect matchings of an even cycle $C = \{e_1, e_2, \cdots e_{2p}\}$, where the set of e_i are the incidence vectors of the edges that form C. Then, $M_1 = \{e_i \ : i \text{ is odd}\}$, and similarly, M_2 is the set of edges where i is even. If we take the difference of these two sets:

$$e_1 + e_3 + \cdots + e_{2p-1} - e_2 - e_4 - \cdots - e_{2p} = \vec{0}$$

and so we show by definition that the set of all e_i is linearly dependent. The edges of T in fact then define a maximal set of linearly independent (5.7) constraints of (LP_{GBS}).

Finally, it is clear that constraint (5.8), combined with all of the (5.6) constraints, and the (5.7) constraints defined by the edges of T, form a maximal linearly independent set. So we know that the extreme point solution (x^i, z^i) can have at most $|X| + |Y| + |E_1^i|$ non-zero variables.

Lemma 5.12. *G^i is connected*

Proof. Suppose for contradiction G^i is not connected, and has k connected components. Let $C_1 \cdots C_k$ be the connected components of G^i. For $0 \leq i \leq k$, let X_{C_i}, Y_{C_i} and O_{C_i} be the subsets of X, Y, and O containing the vertices of C_i respectively. Let T_{C_i} be the subtree of T containing the vertices and edges of C_i. Note that every T_{C_i} will have exactly $|X_{C_i}| + |Y_{C_i}| - 1$ edges. This means that in total, the edges of T will define only $|X| + |Y| - k$ linear independent constraints. This combined with the (5.6) and (5.8) constraints will be a maximal linearly independent set of size only $|X| + |Y| - (k-1) + |E_1^i|$, which by the Rank Lemma (Lemma 5.11) is a contradiction due to the number of non-zero variables in our extreme point solution (x^i, z^i). ☐

Before we prove the next Lemma, consider (D_{GBS}), the dual of (LP_{GBS}) at this iteration. By complimentary slackness we then know:

(CS1). Since for every vertex $u \in X \cup Y$ we have $x^i_u > 0$, every corresponding dual constraint is tight. Ie. $y^i(\delta(u)) + a^i(\delta(u)) = \gamma^i$.

(CS2). Since for all $(u, v) \in E^i_1$, $z^i_{uv} > 0$, every edge $(u, v) \in E^i_1$, $y^i_{uv} = 1$. Then, since $\vec{y} = \vec{1}$, $\gamma^i \geq 1$ by constraint (5.9).

(CS3). For all $(u, v) \in E^i_1 \ : \ u, v \in X$, since $x^i_u + x^i_v > 1$, $a_{uv} = 0$.

As was mentioned, every edge in both sets E^i_1 and E^i_2 is adjacent to an X vertex, and any edge between two X vertices has a value of 0. So at this iteration i, by summing over all vertices in X and using **(CS1)**, we find an expression for the optimal solution to (D_{GBS}), $OPT_D(G^i)$:

$$OPT_D(G^i) = \sum_{u \in X} (y^i(\delta_{E^i_1}(u)) + a^i(\delta_{E^i_2}(u))) - \gamma^i \omega_{G^i} = \gamma^i(|X| - \omega_{G^i})$$

Lemma 5.13. $\frac{|X|+|Y|}{2} < \omega_{G^i} < |X|$

Proof. The upper bound is clear, as suppose $\omega_{G^i} \geq |X|$, then set $x_u = 1$ for all $u \in X$. Since X is adjacent to all edges in $E^i_1 \cup E^i_2$, this solution has an objective value of 0, contradicting optimality. So $\omega_{G^i} < |X|$. Note this argument also implies that $|X| > |Y|$, as otherwise x^i would be an allocation with $\vec{1}^\intercal x^i > |X| > \omega_{G^i}$.

We now prove the lower bound. Since constraint (5.8) is satisfied with equality, $\omega_{G^i} = \alpha_{G^i} \cdot |X| + (1 - \alpha_{G^i}) \cdot |Y|$. Rearranging this for $\alpha_{G^i} > \frac{2}{3}$, we get an inequality for ω_{G^i}.

$$\omega_{G^i} > \frac{2 \cdot |X| + |Y|}{3} > \frac{|X| + |Y|}{2}$$

since $|X| > |Y|$. □

The following is a Corollary of Euler's Identity for planar graphs. We state it here and refer the reader to [33] for the proof.

Corollary 5.14. ***(Euler's Identity)*** *In a connected planar graph $G = (V,\ E)$,*

$$|E| \leq 3|V| - 6$$

Lemma 5.15. *Let G be a planar graph. Double G as in step (i) of Algorithm 5.3 to get the bipartite graph $G' = (V',\ E')$. Then, G' may not be planar.*

Proof. Let G be the planar graph K_5 minus one edge. Let $V = \{a, b, c, d, e\}$ and let $E = \{(u, v) \ : \ u, v \in V\} \setminus \{(a, b)\}$. Double G as in Algorithm 5.3 to get G'. We show that G' has a $K_{3,3}$ minor. Take the subgraph $G'[V \setminus \{a, e'\}]$, then contract the two edges (b, c') and (b, d'). The resulting minor is $K_{3,3}$. See Figure 5.4, where we delete the red vertices. The blue vertex is the result of contracting the red dashed edges. This proves the Lemma by Kuratowski's Theorem [33]. □

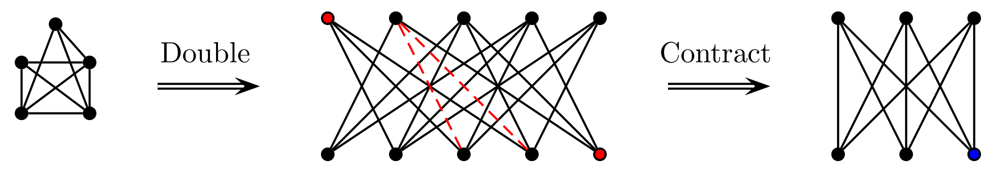

Figure 5.4: Doubled graph may not be planar

We are now ready to prove the main result of this section.

Theorem 4. *For planar graphs with unit edge weights, the* Blocking Pairs *problem is $O(1)$-factor approximable.*

Proof. We are given a planar graph G. Run Algorithm 5.3 on G. If the algorithm completes, we have found a blocking set of size at most $6 \cdot OPT_G$ and we are done. So suppose the algorithm computes an extreme point solution (x^i, z^i) at iteration i, such that none of step(iii)(a) (1), (2), or (3) are true. Define the sets X, Y and O as in Definition 5.9. We are left with a specific $G^i = (X \cup Y \cup O,\ E_1^i,\ E_2^i)$ structure.

For the remainder of the proof, let $m = |X| - \omega_{G^i}$. Suppose the E_2^i edges are no longer forbidden, and can be included in a blocking set (Ie. let them have $z_{uv}^i \geq 0$ variables in (LP_{GBS})). Consider the solution $(\bar{x}, \bar{z})$ defined as follows:

(i) Set $\bar{x}_u = 1$ for ω_{G^i} many vertices in X (to be specified later). Set $\bar{x}_u = 0$ for all other vertices u.

(ii) Set $\bar{z}_{uv}$ as the minimum amount needed to satisfy each respective (5.6) constraint in (LP_{GBS}).

Since by Lemma 5.10, O is an independent set, this solution yields a blocking set $B(\bar{x})$, of size:

$$|B(\bar{x})| \leq \sum_{u \in X : \bar{x}_u = 0} |\delta_{E_1^i}(u)| + |\delta_{E_2^i}(u)| \leq \gamma^i \cdot m + \sum_{u \in X : \bar{x}_u = 0} |\delta_{E_2^i}(u)|$$

where the second inequality is from **(CS2)** and the corresponding (5.9) constraint in (D_{GBS}) for each $u \in X$.

To show this value is bounded, we make use of Euler's identity in planar graphs. By Corollary 5.14, we can bound the second term in the sum expression of $B(\bar{x})$. We started with a planar graph G, and at step (i) of Algorithm 5.3 we double the number of edges to create G'. When the algorithm reaches iteration i, by Lemma 5.12 we are left with a connected structure. Consider the subgraph $G_p = (X \cup Y,\ E_2^i)$, which contains all of E_2^i

as its edge-set. Lemma 5.15 states that G_p may not be planar, but it is a subgraph of the doubled structure of the planar graph G. So then:

$$|E_2^i| \leq 2(3(|\{X \cup Y\}| - 6)$$

by Lemma 5.14.

This means, applying the pigeonhole principle, there exists some $u_0 \in X$ with $|\delta_{E_2^i}(u_0)| \leq \frac{2(3|\{X \cup Y\}|-6)}{|X|}$. Suppose we remove u_0 and all adjacent edges from the graph, and repeat. Our pigeonhole argument still applies, but now we can find a vertex $u_1 \in X \setminus \{u_0\}$ with $|\delta_{E_2^i}(u_1)| \leq \frac{2(3|\{X \cup Y\}|-6)-|\delta_{E_2^i}(u_0)|}{|X|-1}$. We can repeat this process m times, to find a set $D := \{u_0, u_1, \cdots, u_{m-1}\}$ such that:

$$\sum_{u \in D} |\delta_{E_2^i}(u)| \leq \sum_{u \in D} \frac{2(3|\{X \cup Y\}| - 6)}{|X| - m} \leq \frac{2m(3|\{X \cup Y\}|)}{\omega_{G^i}} \leq \frac{6m(|X| + |Y|)}{\frac{|X|+|Y|}{2}} = 12m$$

where the denominator in last inequality comes from the lower bound of $\omega_{G^i} > \frac{|X+|Y|}{2}$ specified in Lemma 5.13. Note that this argument requires all edges to have weight $w_{uv} = 1$, as with arbitrary edge weights we cannot bound the cost $\sum_{u \in D} \sum_{(u,v) \in \delta_{E_2^i}(u)} w_{uv}$.

Putting it all together, in step (i) of constructing $(\bar{x}, \bar{z})$, set $\bar{x}_u = 0$ for all $u \in D$, and $\bar{x}_u = 1$ for $u \in X \setminus D$. Set $\bar{x}_u = 0$ for all $u \in Y \cup O$. Then since $\gamma^i \geq 1$ by **(CS2)**, and constraint (5.9) for all $u \in D$ implies $|\delta_{E_1^i}(u)| \leq \gamma^i$ as mentioned:

$$|B(\bar{x})| \leq (12 + \gamma^i)m \leq 13 \cdot \gamma^i m = 13 \cdot OPT_D(G^i) \leq 13 \cdot OPT_{G^i}$$

□

Let $(\bar{x}, \bar{z})$ be the solution that is constructed in the proof of Theorem 4 of size at most $13 \cdot OPT_{G^i}$. To apply the proven result, we modify Algorithm 5.3 by adding another condition inside the while loop of step (iii). We add the following Step (iii)(a) (4):

Else If return the solution $(\bar{x}, \bar{z})$

We return the approximate blocking set, and map it back to the solution $(\hat{x}, \hat{z})$ in step (iv). Step (v) may round some z_{uv} variables to 1 then step (vi) returns the blocking set $B = \{(u, v) \in E \ : \ \hat{z}_{uv} = 1\}$ for the graph G. The proof of Theorem 4 then implies that $|B| \leq 26 \cdot OPT_{LP}(G) \leq 26 \cdot OPT_G$.

We note that in practice, anytime there exists a vertex $u \in X$ such that $|\delta_{E_1^i}(u)| > 1$, then $\gamma^i > 1$. In these graphs, the approximation factor for this algorithm is much smaller. For example, even with $\gamma^i = 2$, the approximation factor drops to 14.

5.3 Blocking Sets in Graphs of Bounded Tree Width

As was mentioned in Section 2.3, many problems which are intractable in general graphs can be solved polynomially in graphs of bounded treewidth. In this section, we describe a polynomial time dynamic programming algorithm to optimally solve the Blocking Pairs problem in graphs of bounded treewidth. We restrict our analysis to graphs with unit edge weights, then mention the extension of our work to weighted graphs.

We begin by stating a useful lemma that we will use later.

Lemma 5.16. *Let B be an optimal blocking set for a graph $G = (V,\ E)$. Then there exists a blocking set and allocation pair (B, x) such that $x_u \in \left\{0, \frac{1}{2}, 1\right\}$ for all $u \in V$.*

Proof. Suppose for graph $G = (V,\ E)$ we are given an optimal blocking set and allocation pair, (B, x). Let ν^* be the size of the maximum matching in graph G.

Remove the edges $(u, v) \in B$ from G, forming a new graph $G' = (V,\ E \setminus B)$. By definition, x is a vertex cover for G' of size ν^*. Since the canonical vertex cover LP is known to be half-integral, if a vertex cover of size ν^* exists, a vertex cover x^* of size at most ν^* with $x^* \in \left\{0, \frac{1}{2}, 1\right\}$ exists. Then (B, x^*) is a optimal blocking set and allocation pair for graph G. □

Next, we describe some notation used for the remainder of this section. Let $G = (V,\ E)$ be a graph with binary tree decomposition $(Y,\ T)$ of width k, and maximum matching size of ν^*. Let T be a tree on a set of nodes $\{1, 2, \cdots\}$, with bags $\{Y_1, Y_2, \cdots\}$, and suppose node r is the root of T. Additionally, for a node i, let D_i be the set of all vertices that exist in bags at non-root nodes in the subtree rooted at i. Formally, $D_i = \{u \in Y_j\ :\ j \text{ is a proper descendant of } i \text{ in tree } T\}$. Let $E[Y_i]$ be the set of edges of G induced by the vertices in Y_i, $E[Y_i] = \{(u, v)\ :\ (u, v) \in E,\ u, v \in Y_i\}$. Let $G[Y_i] = (Y_i,\ E[Y_i])$ be the subgraph of G induced by the vertices of Y_i.

For some node i, suppose x is an allocation assigning a value to every vertex in Y_i. We define the set of allocation *extensions* of x onto a node j.

Definition 5.17. Let x be an allocation assigning a value to every vertex in Y_i for a node i. Then, $Ext(x,\ j,\ \nu_j)$ is the set of allocations y on the vertices Y_j such that:

(i) For every $u \in (Y_i \cap Y_j)$, $y_u = x_u$.

(ii) For every $u \in (Y_j \setminus Y_i)$, $y_u \in \left\{0, \frac{1}{2}, 1\right\}$ (Lemma 5.16).

(iii) $y(Y_j) \leq \nu_j$.

For some edge set $E[V']$ where $V' \subseteq V$, let $C\big(\ E[V'],\ x\ \big)$ be the set of blocked edges in $E[V']$ induced by an allocation x, that is

$$C\big(\ E[V'],\ x\ \big) = \left\{(u, v)\ :\ (u, v) \in E[V'],\ x_u + x_v < 1\right\}$$

Consider the problem of finding the minimum blocking set on the subtree rooted at a node i. Let p and q be the children of i in T. If i has just one or even no children, then p and q may not exist, in which case, we will adjust our recursive function accordingly by removing the non-existent children.

Let $\beta\big(\ i,\ x,\ \nu\ \big)$ be the minimum blocking set on the subgraph $G[Y_i \cup D_i]$ (Ie. the subtree rooted at i), where x is an allocation on the vertices of Y_i, and $\nu = x(Y_i\ \cup\ D_i)$ is the exact sum of the allocation on the vertices $\big\{u\ :\ u \in (Y_i \cup D_i)\big\}$. We find every $\beta\big(\ i,\ x,\ \nu\ \big)$ by trying all possible $\big\{0, \frac{1}{2}, 1\big\}$ assignments for the vertices in Y_i. Consider one such assignment. It induces a blocking set with members in $E[Y_i]$. For any distribution (ν_p, ν_q) of $\nu - x(Y_i)$ over the two subtrees rooted at p and q, we recursively compute the minimum blocking sets of the subtrees (ensuring the x-assignments agree with the one on the vertices of Y_i). We find the distribution that leads to the minimum blocking set for the subtree rooted at i, and return the blocking set itself.

We formally define the recursive function β, slightly modifying standard notation. The min functions will return the actual sets of minimum cardinality, instead of their numerical size:

$$\beta\big(\ i,\ x,\ \nu\ \big)\ =\ C\big(\ E[Y_i],\ x\ \big) \bigcup \min_{(\nu_p,\nu_q)} \bigcup_{j=p,q} \min_{y\,\in\, Ext(x,\, j,\, \nu_j)} \Big[\beta\big(\ j,\ y,\ \nu_j\ \big) \backslash C\big(\ E[Y_i],\ x\ \big)\Big]$$

where $\nu_p + \nu_q = \nu - x(Y_i)$.

As mentioned, if p or q do not exist because i does not have two children, in the above equation, $\nu_p = 0$ and $\nu_q = 0$ respectively. We remove the elements of $C\big(\ E[Y_i],\ x\ \big)$ from the blocking sets for the subtrees rooted at p and q to eliminate all duplicate counting situations (Ie. when an edge (u, v) exists in both Y_i and Y_j).

For a graph $G = (V,\ E)$ with tree decomposition $(Y,\ T)$, every edge $(u, v) \in E$ must be contained in some bag Y_i by property (TW1) of tree decompositions. This means that every edge will be considered at some bag in the calculation, so we do not need to consider any cross-bag edges (u, v), with $u \in Y_i$ and $v \in Y_j$, $j \neq i$.

Note that it is sufficient to consider just the two possible children of a node i when calculating β. By the running intersection rule, property (TW3) of tree decompositions, any overlapping edge (u, v) that exists in both Y_i and Y_j, must also exist in the bags at every node along the path from i to j. This means that any common vertices in Y_i and D_i will exist in $(Y_p \cup Y_q)$.

The edges in the bags at the children of i may also not be disjoint, but by the same property (TW3), any edge in the bags of both children must also be a part of Y_i, the parent's bag. In case such an edge is blocked, we make sure not to count it more than once.

An outline of our algorithm is as follows. To calculate the blocking set for graph G, we take a tree decomposition $(Y',\ T')$ of G, and if T' is not a binary tree, using Bodlaender's method [7], Lemma 2.5, we convert (Y', T') into a binary tree decomposition (Y, T). If T' is a binary tree, let $(Y, T) = (Y', T')$. We arbitrarily pick a node to be the root of T, and call it node r as above. We then calculate the β and C values by a bottom-up traversal of the tree. The optimal blocking set will be the minimum sized β set across all possible allocations x at the root node.

Algorithm 5.18.
Input: Graph $G = (V, E)$ of bounded treewidth, and tree decomposition (Y', T')
Output: Optimal blocking set B for the graph G

(i) Find a binary tree decomposition (Y, T) of G

  (a) If T' is a binary tree, let $(Y, T) = (Y', T')$.

  (b) If T' is not a binary tree, transform (Y', T') into a binary tree decomposition (Y, T) as mentioned in [7]

(ii) Pick an arbitrary root node r of T

(iii) Using a bottom-up traversal of the tree, calculate $\beta\big(\, i,\ x,\ \nu \,\big)$ for every possible x allocation, and for every $\nu \in \{0, \frac{1}{2}, 1, \cdots, \nu^* - \frac{1}{2}, \nu^*\}$.

(iv) Find and return the minimum blocking set B at the root node r:

$$x^* = \arg\min_x \ |\, \beta\big(\, r,\ x,\ \nu^* \,\big)\, |$$
$$B\ = \beta\big(\, r,\ x^*,\ \nu^* \,\big)$$

We will now prove the algorithm's correctness, and then argue that it requires a number of calculations polynomial in the treewidth value k.

Lemma 5.19. *For a graph $G = (V, E)$, the subsection of the optimal blocking set of G on the graph $G[Y_i \cup D_i]$, is stored at $\beta\big(\, i,\ x,\ \nu \,\big)$ for some allocation x, with $\nu = x(Y_i \cup D_i)$.*

Proof. We prove the lemma using induction on ℓ, the depth of the tree rooted at node $i \in T$.

If $\ell = 0$, then $D_i = \emptyset$ and the result is trivial. The algorithm calculates the blocking set for every possible allocation x that sums to ν and sets $\beta\big(\, i,\ x,\ \nu \,\big)$ to the set.

Suppose for induction the lemma is true for $\ell \leq (d-1)$ for some $d \in \mathbb{Z}^+$. Now suppose that the tree rooted at i has depth $\ell = d$. Let p and q be the children of i. The calculation of β is calculated for every possible allocation x on the vertices Y_i, being minimized over every possible combination of ν_p and ν_q. These are the portions of ν that are assigned to each of the subtrees rooted at p and q. One such allocation and distribution will be consistent with the optimal solution on G. Call this allocation x^*, and the distribution (ν_p^*, ν_q^*). Then, β is minimized over every possible extension $y \in Ext(x^*,\ j,\ \nu_j^*)$ for $j = p, q$.

Then, by induction, for some $y \in Ext(x^*,\ j,\ \nu_j^*)$, $\beta\big(\ j,\ y,\ \nu_j^*\ \big)$ will calculate the subsection of the optimal blocking set of G on the subgraph $G[Y_j \cup D_j]$ for $j = p, q$. The calculation for β at node i will include these allocation extensions since it is minimized over all possible $y \in Ext(x^*,\ j,\ \nu_j^*)$ when it is computed with allocation x^* and distribution (ν_p^*, ν_q^*). This proves the lemma. □

Using Lemmas 2.5, 5.16, and 5.19, we are now ready to prove the main result of this section.

Theorem 5. *For a graph $G = (V,\ E)$ with tree decomposition $(Y',\ T')$ of width k, Algorithm 5.18 finds the optimal blocking set in time $O(k \cdot 9^k \cdot |V|^3)$.*

Proof. By Lemma 5.19, after obtaining a binary tree decomposition of G, we know the optimal blocking set is one of the calculated β values at the root node. We show that all the calculations can be done in time polynomial in k. We can then simply search through the at most 3^k blocking sets calculated at the root node to find the minimum.

Since at any node i there are at most k vertices in the Y_i bag, by Lemma 5.16, there are at most 3^k possible allocations x for the vertices in Y_i. There are also at most $2\nu^* + 1 = O(|V|)$ possibilities for ν, as we must also consider half integral values. This means we must compute $O(3^k \cdot |V|)$ β values at each node.

Let p and q be the children of node i. For every β calculation at node i, it is clear that $C\big(\ E[Y_i],\ x\ \big)$ can be found in $O(1)$ time. In addition, there are again $2\nu^* + 1 = O(|V|)$ possibilities for the pair (ν_p, ν_q) to be split, and finally, at most another 3^k possibilities for the extensions $y \in Ext(x,\ j,\ \nu_j)$ for $j = p, q$. This means for every β calculation at a node i, we must minimize over $O(3^k \cdot |V|)$ possible β values at i's children.

Combining these, we need to calculate the $O(3^k \cdot |V|)$ possible β values at each node, at a time cost of $O(3^k \cdot |V|)$ per calculation. This gives us a total time of $O(3^k \cdot 3^k \cdot |V|^2) = O(9^k \cdot |V|^2)$ for the computation of each node.

By Lemma 2.5, we know there are $O(k \cdot |V|)$ nodes in T, so we get a total time complexity of $O(k \cdot 9^k \cdot |V|^3)$ for Algorithm 5.18. □

We mention that the vertex cover problem is known to be half integral even on graphs with integer weighted edges. If the maximum weight of an edge in G is bounded, with a slight modification, Algorithm 5.18 will continue work in polynomial time. One can simply search through the $O(2 \cdot w_{max})$ options for each vertex in step (iii), where w_{max} is the maximum weight of any edge in G. We must also adjust the definition of C to

$C\Big(\ E[V'],\ x\ \Big) = \Big\{\sum_{(u,v)} w_{uv}\ :\ (u,v) \in E[V'],\ x_u + x_v < w_{uv}\Big\}$. In this section we assumed that all edges have weight $w_{uv} = 1$ to simplify our analysis.

Chapter 6

Conclusions

Finding balanced solutions to the bargaining game is the main focus of this thesis. The discussions can be divided into three main categories.

First, we described a model introduced by Kleinberg and Tardos [18] for the bargaining game on general graphs. We discussed a number of results about the relationship between bargaining solutions and concepts from cooperative game theory due to Bateni et al. [4]. We recalled Bateni's proofs that a stable solution is a member of the core of the bargaining game, and then that a balanced solution corresponds to an allocation in the core intersect prekernel of the game. Using these results, we sketched a proof by Kleinberg and Tardos showing that if a graph has a stable solution, it has a balanced solution. We then discussed another result by Bateni et al., who argued that for stable graphs, all balanced bargaining solutions can be constructed in polynomial time.

Second, we defined the Blocking Pairs problem, which has been proven to be NP-complete by Biro, Kern, and Paulusma [5]. We presented the proof of NP-completeness, reducing from the Maximum Independent Set problem. Using the relationships between the bargaining game and cooperative game theory, we then proved that starting with a blocking set and allocation pair (B, x) for an unstable graph G, we can compute a series of transfers to find a balanced bargaining allocation on $G' = (V,\ E \setminus B)$.

The problem of finding minimal blocking sets in unstable graphs is motivated by the existence of balanced bargaining allocations for graphs obtained by removing a blocking set. We presented a natural integer-programming formulation for the Blocking Pairs optimization problem. Unfortunately we proved that the linear-programming relaxation does not have an integrality gap of 1, even for graphs with unit edge weights. Instead, using iterative rounding, we described an LP-based $O(1)$-approximation algorithm for finding cheap blocking sets in planar graphs, again with unit edge weights. We then presented a polynomial time dynamic programming algorithm to optimally solve the Blocking Pairs problem for graphs of bounded treewidth. With a slight modification, the algorithm works for graphs with arbitrary positive integer edge weights.

6.1 Future Work

In this section we describe our topics of interest moving forwards.

Most of the later results presented in this thesis apply to specific graph classes, some with even further restrictions on edge weights. Our first steps are to expand our results to a wider range of general graph classes.

Instead of considering expanded graph classes, another approach is to allow for varying utility functions among agents. As was mentioned in Chapter 1, Kearns and Chakraborty [8] studied how differing utility functions affect solutions to the bargaining game. Their model differs in that it allows for agents to make an arbitrary number of deals. It would be interesting to expand our results for graphs using a combined model, maintaining our restriction to single deals for agents, but allowing agents to have more complex utility functions.

A third direction for future work is to consider blocking sets when we relax our model allowing agents to make multiple deals in the network. As was mentioned in Section 3.4, Bateni et al. [4] prove results about stable and balanced bargaining solutions for certain graph classes in this relaxed model. We are interested in finding balanced bargaining allocations when removing blocking sets in unstable graphs, in which agents can make a capacity constrained number of contracts.

We have also studied two additional problems closely related to the Blocking Pairs problem. The second was previously mentioned in Chapter 4. We formally define them here and then discuss them in more detail. Our work has only considered the unweighted case. For a graph $G = (V,\ E)$, let D_V and D_E be subsets of V and E respectively.

Vertex-Deletion Stability
Instance: a bargaining game on $G = (V,\ E)$ and an integer $k \geq 0$.
Question: does the graph $G[V \setminus D_V]$ allow a stable solution x with $|D_V| \leq k$?

Edge-Deletion Stability
Instance: a bargaining game on $G = (V,\ E)$ and an integer $k \geq 0$.
Question: does the graph $G^- = (V,\ E \setminus D_E)$ allow a stable solution x with $|D_E| \leq k$?

We note that while the Edge-Deletion Stability problem is very similar to the Blocking Pairs problem, the difference lies in the total value of the allocation. In the Blocking Pairs problem, $\nu(G)$ is distributed across an allocation, where as in the Edge-Deletion Stability problem, we are limited to distributing the size of the maximum matching of the resulting graph $\nu(G^-)$. We note that any edge stability set solution, D_E, is a feasible solution to the Blocking Pairs problem. However, it will often not be optimal.

We suspect that both the Vertex-Deletion Stability and Edge-Deletion Stability problems are NP-hard. They are respectively very similar to two known NP-complete problems, König Vertex-Deletion and König Edge-Deletion, studied by Mishra et al. [23]. To describe the similarities between the problems, we must first define König-Egerváry graphs.

Definition 6.1. Let the size of the minimum integral vertex cover for a graph G be $\mu(G)$. A *König-Egerváry graph* (KEG graph), is a graph where $\mu(G) = \nu(G)$. That is, a graph in which the size of the minimum integral vertex cover equals the size of a maximum matching.

This definition leads to two natural deletion problems. Again let D_V and D_E be subsets of V and E respectively.

König Vertex-Deletion
Instance: a graph $G = (V,\ E)$ and an integer $k \geq 0$.
Question: is $G[V \setminus D_V]$ a KEG graph with $|D_V| \leq k$?

König Edge-Deletion
Instance: a graph $G = (V,\ E)$ and an integer $k \geq 0$.
Question: is the graph $G^- = (V,\ E \setminus D_E)$ a KEG graph with $|D_E| \leq k$?

By definition, we know that any KEG graph is also a stable graph. A vertex cover for a KEG graph is itself a stable allocation. Unfortunately the converse is not true.

Consider K_4, the clique on four vertices, with $\nu(K_4) = 2$ as shown in Figure 6.1 in red. K_4 is clearly a stable graph, as the allocation that assigns $x_u = \frac{1}{2}$ to each vertex is stable. Unfortunately however, $\mu(K_4) = 3$, as is also shown in Figure 6.1 in blue. Hence K_4 is not a KEG graph.

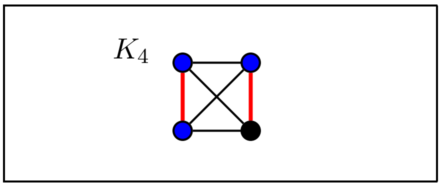

Figure 6.1: K_4 is stable but is not a KEG

The slight variation in definition, allowing non-integral stability allocations, is enough so that we can not naturally reduce either of the Vertex-Deletion Stability and Edge-Deletion Stability problems to their respective König deletion problems for general graphs. Still, their similarities suggest that the stability problems are NP-hard, and we are searching for a reduction.

Bibliography

[1] Yossi Azar, Benjamin Birnbaum, L. Elisa Celiz, Nikhil R. Devanur, and Yuval Peres. Convergence of local dynamics to balanced outcomes in exchange networks. *50th IEEE Symposium Foundations of Computer Science*, 2009.

[2] Yossi Azar, Nikhil R. Devanur, Kamal Jain, and Yuval Rabani. Monotonicity in bargaining networks. *SODA*, 2010.

[3] Brenda S. Baker. Approximation Algorithms for NP-Complete Problems on Planar Graphs. *Journal of the ACM*, 41:153–180, 1994.

[4] MohammadHossein Bateni, MohammadTaghi Hajiaghayi, Nicole Immorlica, and Hamid Mahini. The cooperative game theory foundations of network bargaining games. *Lecture Notes in Computer Science*, 6198:67–78, 2010.

[5] Peter Biro, Walter Kern, and Daniel Paulusma. Computing solutions for matching games. *International Journal of Game Theory*, 2011.

[6] Hans L. Bodlaender. Treewidth: Algorithmic techniques and results. *Mathematical Foundations of Computer Science*, pages 19–36, 1997.

[7] Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1-2):1–45, 1998.

[8] Tanmoy Chakraborty and Michael Kearns. Bargaining solutions in a social network. *LNCS*, 5385:548–555, 2008.

[9] Tanmoy Chakraborty, Michael Kearns, and Sanjeev Khanna. Network bargaining: Algorithms and structural results. 2009.

[10] M. Davis and M. Maschler. The kernel of a cooperative game. *Naval Research Logistics Quarterly*, 12:223–259, 1965.

[11] Kimmo Eriksson and Johan Karlander. Stable outcomes of the roomate game with transferable utility. *International Journal of Game Theory*, 29:555–569, 2001.

[12] U. Faigle, W. Kern, and J. Kuipers. On the computation of the nucleolus of a cooperative game. *International Journal of Game Theory*, 30:79–98, 1999.

[13] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some Simplified NP-complete Graph Problems. *Theoretical Computer Science*, 1:237–267, 1976.

[14] A. Ghouila-Houri. Caracterisation des matrices totalement unimodulaires. *C. R. Acad. Sci. Paris*, 254:1192–1194, 1962.

[15] D. B. Gilles. Solutions to general non-zero sum games. *Contributions to the theory of games (Annals of Mathematics Studies 40)*, 4:47–85, 1959.

[16] K. Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21:39–60, 1998.

[17] Walter Kern and Daniel Paulusma. Matching games: The least core and the nucleolus. *Mathematics of Operations Research*, 28:294–308, 2003.

[18] Jon Kleinberg and Eva Tardos. Balanced outcomes in social exchange networks. *Symposium on Theory of Computing*, 2008.

[19] Lap Chi Lau, R. Ravi, and Mohit Singh. *Iterative Methods in Combinatorial Optimization.* Cambridge University Press, 2011.

[20] Lap Chi Lau and Mohit Sing. Additive approximation for bounded degree survivable network design. *STOC*, 2008.

[21] Laszlo Lovasz and Michael D. Plummer. *Matching Theory.* AMS Chelsea Publishing, 1986.

[22] Holger Meinhardt. An lp approach to compute the pre-kernel for cooperative games. *Computer and Operations Research*, 33:535–557, 2004.

[23] Sounaka Mishra, Venkatesh Raman, Saket Saurabh, Somnath Sikdar, and C.R. Subramanian. The complexity of konig subgraph problems and above-guarantee vertex cover. *Algorithmica*, 61:857–881, 2010.

[24] John F. Nash. The bargaining problem. *Econometrica*, 18:155–162, 1950.

[25] Martin J. Osbourne. *An Introduction to Game Theory.* Oxford University Press, 2003.

[26] Neil Robertson and Paul D. Seymour. Graph minors. iii. planar tree-width. *Journal of Combinatorial Theory, Series B*, 36:49–64, 1984.

[27] Neil Robertson and Paul D. Seymour. Graph minors. ii. algorithmic aspects of tree-width. *Journal of Algorithms*, 7:309–322, 1986.

[28] D. Schmeidler. The nucleolus of a characteristic function game. *SIAM Journal of Applied Mathematics*, 17:1163–1170, 1969.

[29] A. Schrijver. *Theory of linear and integer programming.* John Wiley & Sons, New York, 1986.

[30] L.S. Shapley and M. Shubik. The assignment game i: The core. *International Journal of Game Theory*, 1:111–130, 1972.

[31] R.E. Stearns. Convergent transfer schemes for n-person games. *Trans. Amer. Math. Soc.*, 134:449–459, 1968.

[32] Robert J. Vanderbei. *Linear Programming Foundations and Extensions.* Springer, 2007.

[33] Douglas B. West. *Introduction to Graph Theory.* Prentice Hall, 2001.

[34] Mihalis Yannakakis. Edge-deletion problems. *SIAM Journal on Computing*, 10(2):297–309, 1981.