2012

Network Design Under Competition

Tolga Han Seyhan
Lehigh University

LEHIGH
UNIVERSITY

NETWORK DESIGN UNDER COMPETITION

by

Tolga Han Seyhan

Presented to the Graduate and Research Committee
of Lehigh University
in Candidacy for the Degree of
Doctor of Philosophy

in
Industrial Engineering

Lehigh University

May 2012

Approved and recommended for acceptance as a proposal in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Date

Dr. Lawrence V. Snyder

Dissertation Advisor

Accepted Date

Committee:

Dr. Lawrence V. Snyder, Chairman

Dr. James A. Dearden

Dr. Theodore K. Ralphs

Dr. Aurélie C. Thiele

Acknowledgments

I am deeply grateful to my advisor Larry Snyder. He has given me freedom to find and pursue my research interests, and supported me all along this bumpy road. He is an awesome teacher, researcher and person. I would like to thank my committee members for taking the time and giving me invaluable feedback and their assistance throughout this study. Jim Dearden rekindled my interest in game theory with his great teaching. The final research chapter came to life through our collaboration. My wife Ruken and I owe Aurélie Thiele a lot for her guidance and encouragement. Without Ted Ralphs' push, I would not learn much about bilevel programming.

I would like to thank to all my professors at Lehigh, especially George Wilson, Vladimir Dobric and Jitamitra Desai. I learned a great deal from them. I am thankful to Rita Frey, Kathy Rambo and Brie Lisk who were always there to guide us safely through school bureaucracy; and Kathleen Hutnik, the patron saint of graduate students.

If I feel home at Lehigh — I do — I owe it to my dear friends here. I am thankful to Zeliha Akça, Elçin Çetinkaya, Erdem and Öznur Arslan, Burak and Figen Bekişli, Aykağan and Ilgın Ak, Alper and Burcu Uygur, Oğuz and Zümbül Atan, Gökhan and Hayriye Aydar, Hakan and Pınar Güvelioğlu, Hakan and Hamra Bakırcıoğlu, Özer and Özgül Akuş, Berrin Aytaç, Menal Güzelsoy, Cem Alper Özen, Ana Alexandrescu and several others for their friendship and support all these years.

Despite the thousands of miles between us, I have always felt my family's support next to me. I cannot thank enough to my mother Menşure, my father Abidin, my sister Zübeyde and brothers Faruk and Ali.

And, Ruken — my love, friend, colleague, boss. I do not know if I could do it without you being there for me any moment. It has been five years since we embarked on our quest for knowledge. We have fought the homeworks and slain the exams. We are almost there.

Contents

List of Tables

List of Figures

Abstract

In this dissertation, we study network design problems that involve multiple — usually two — rational decision makers. Within this class of problems, we are particularly interested in hierarchical structures, in which decision makers (DMs) make their optimal decisions by anticipating other DMs' optimal responses. We present a novel method for solving the problem of the leader to near optimality. It relies on heuristically reformulating the follower's problem and embedding it into the leader's problem as constraints. The resulting formulations are more tractable and provide near-optimal suggestions for the leader's decision.

First, we consider a competitive facility location problem where two firms engage in a leader–follower game. Both firms would like to maximize the customer demand that they capture. Given the other player's decision, each player's problem is the classical Maximal Covering Location Problem (MCLP). We use the greedy add algorithm as a proxy for the follower's response and formulate a mixed-integer programming (MIP) model that embeds the follower's heuristic response into the leader's constraints, and solve it as a single-level program. We propose and compare alternative formulation strategies.

In our second study, we consider a network design game played on a given graph. Each player builds her own network, maximizing individual profits. Therefore, each player's problem given the other's decision is a variant of the Prize-Collecting Steiner Tree problem (PCST). We first present an MIP formulation that finds a pure strategy Nash-equilibrium when the players decide simultaneously. Then we consider a Stackelberg setting and present a heuristic reformulation strategy for the follower's problem based on a pruning algorithm.

Finally, we consider the university and job application processes, where an applicant to a particular institution may increase the probability of receiving an offer by demonstrating her interest in that institution. We characterize the optimal set of institutions to which a candidate should demonstrate interest (i.e., send costly signals) from among the set of institutions to which she has applied. We demonstrate that a greedy algorithm implements the optimal decision rule. In addition, we show that the problem can be posed as a nontraditional longest path problem, for which we introduce a more efficient dynamic programming algorithm and which we can formulate using only affine constraints.

Chapter 1

Introduction

1.1 Optimization and Competition

Network design problems, in general, are fairly easy to describe yet computationally difficult to solve. The existence of multiple rational decision makers (DMs) make this decision-making process even more demanding. The reason is that a DM has not only to find an optimal solution to her problem, but also has to consider rational responses of other DMs, whose decisions may alter her results.

Problems involving multiple DMs are the subject of game theory and have been studied by mathematicians and economists over several decades. A specific type of these problems, introduced in 1934 by von Stackelberg (2011, English translation), is the Stackelberg game, in which two firms engage in a leader–follower competition over production quantity. The leader is the first mover in the game. His decision is observed by the follower and is responded to. Depending on the structure of the game, the leader may or may not have a first mover's advantage. In these games, the goal is to arrive at a *subgame perfect Nash equilibrium*, which is to solve the problem from the leader's point of view ensuring that the follower's response is optimal given the leader's decision. These problems yield extensive insight about the economic behavior of humans. However, in the context in which they are studied in economics, they do not aim to solve real-world scale problems.

$$\begin{array}{c}\text{Leader's}\\\text{problem}\end{array}\left\{\begin{array}{ll} \min & f_L(x,y^*) \\ \text{s.t.} & g_L(x,y^*) \le 0 \\ & \\ & \left.\begin{array}{rl} y^* \in \arg\min & f_F(x,y) \\ \text{s.t.} & g_F(x,y) \le 0 \\ & y \in Y \end{array}\right\}\begin{array}{c}\text{Follower's}\\\text{response}\end{array} \\ & x \in X, y^* \in Y \end{array}\right.$$

Figure 1.1: A generic bilevel program

Their introduction to operations research occurred in the early 1970's by Bracken and McGill (1973). Candler and Norton (1977) coined the term multilevel (bilevel for two DMs) programming. The leader's problem is generally called the upper level problem and the follower's problem is called the lower level problem. Figure 1.1 illustrates a generic bilevel program, where subscripts L and F denote the leader and the follower, respectively. f and g and indicate their functions and constraints, respectively. $x \in X$ and $y \in Y$ are, respectively, the leader's and the follower's decision variables, and y^* correspond to the optimal response of the follower given a particular leader decision x. Bilevel programs generalize the idea of sequential nonzero-sum games and are useful in modeling problems with competition. However, these nested programs suffer heavily from computational complexity, mainly because checking the feasibility of a particular solution is itself an optimization problem hence is as difficult as the lower level problem. Even in the simplest case, in which both upper and lower level problems are linear, the resulting bilevel linear programming problem is in the complexity class NP-hard, as shown by Jeroslow (1985). Ben-Ayed and Blair (1990) uses a simpler argument involving the integer Knapsack problem and a bilevel representation of 0-1 variables.

Network design problems involve binary decisions; hence, their bilevel versions are intrinsically more difficult than bilevel linear programming. Jeroslow (1985) shows that these belong to higher complexity classes; for instance, decision problem of the bilevel integer program belongs to Σ_2 in polynomial hierarchy. For more information on computational complexity readers can refer to Arora and Barak (2009) and Papadimitriou (1994). Detailed discussions of polynomial hierarchy can be found in Stockmeyer (1977) and Papadimitriou (1994, chapter 17). As a result, several solution methods have been developed for integer bilevel programming problems. Next, we are going to briefly discuss some of these methods. Then, we are going to discuss application areas of bilevel programming. Finally, we are going to provide an outline of the dissertation research and discuss our contributions. A comprehensive discussion of these is beyond the scope of this dissertation. Interested readers can refer to reviews by Colson et al. (2007), Dempe (2003) and Vicente and Calamai (1994); books by Dempe (2002) and Bard (1998); and bilevel programming related articles in Encyclopedia of Optimization (Floudas and Pardalos, 2009).

1.1.1 Solution Methods

Linear bilevel programs have been studied extensively and information on methods and applications can be found in the aforementioned references. We are going to briefly investigate solution methodologies for bilevel programs, which include integer variables in at least one of the levels. They are deemed very hard to deal with as they bring serious algorithmic challenges, such as relaxations and fathoming rules which do not work as in the case of standard mixed integer programs. Perhaps mainly for that reason, these problems have attracted quite limited attention. Vicente and Calamai (1994) provide a large bibliography of the early developments. Almost all are related to problems without integrality constraints and the authors only cite four papers (Moore and Bard, 1990; Wen and Yang, 1990; Bard and Moore, 1992; Edmunds and Bard, 1992) that are related to integer bilevel programs. The earliest solution approach for integer bilevel programs that we know of is by Moore and Bard (1987), which is probably published as Bard and Moore (1992).

Moore and Bard (1990) introduce a basic implicit enumeration scheme that mirrors the branch-and-bound procedure for solving mixed integer programs. The algorithm is designed for mixed integer linear bilevel programs that can have integer variables on both levels. The algorithm generates upper bounds (for minimization) by discarding the lower level objective. It continues with solving the relaxed problem and branches when the solution is nonintegral. Then it fixes the leader's variables, and solves the follower's problem to obtain a bilevel feasible solution and hopefully a lower bound. It continues in this manner until the bounds are tolerably close.

Wen and Yang (1990) study a mixed integer bilevel linear program in which integrality constraints exist only for the leader's decision variables and constraints are common for both decision makers. They implement a standard branch-and-bound algorithm. They start by solving the follower's problem and then the leader's problem, setting integer variables to zero. They use the objective and dual solutions in the upper bounding step (they consider a maximization problem) and update this bound as they branch on variables by sensitivity analysis. They also propose a heuristic procedure that runs in as many iterations as the number of integer variables. They develop an index to select which integer variables to set to one.

Bard and Moore (1992) take a simple path in solving a pure zero-one integer bilevel programming problem. They use the branch-and-bound approach for solving standard integer programs. On the other hand, they switch the leader's and the follower's objectives and solve the overall problem for the follower. They replace the leader's (maximization) objective with a parameterized constraint of the form ($F(x) \geq \alpha$). They control the number of ones among the leader's binary decision variables by another parameterized constraint ($\sum x \geq \beta$). One important detail is that they assume the constraint and objective coefficients to be integral. They adjust these bounds, increasing the leader's objective until it becomes infeasible to do so anymore. Edmunds and Bard (1992) introduce an algorithm to solve mixed integer nonlinear bilevel programs. The programs they consider have binary integrality constraints for the leader's decision and a convex leader objective function. On the other hand, the follower's objective is convex quadratic and his constraints are linear. They use the common trick for linear follower programs, and use the Karush-Kuhn-Tucker conditions to represent it. They propose a branch-and-bound algorithm. Their approach is the same as the branch-and-bound approach for solving standard mixed integer programs except that they also branch on the complementarity constraints. This is also similar to associating a binary variable (u) to each complementarity constraint, and fixing the primal function value ($g() \leq 0$) or multiplier ($\mu \geq 0$) to zero by, respectively, either ($g() \geq M(u-1)$) or ($\mu \leq M(u)$). Bard's collected studies also appear in Bard (1998, Ch. 6).

After branch-and-bound, the second approach that comes to mind is cutting planes. Thirwani and Arora (1997) introduce a cutting plane algorithm to solve the integer linear fractional bilevel programs. The algorithm starts by finding an integer solution for the leader's problem. Then the follower's problem is solved to check for optimality (i.e. if the leader anticipated the optimal response). Then a special cut is added to find the next best integer solution, which is then checked for optimality. Nonintegral solutions are solved until integrality using Gomory cuts. Dempe (2002, Ch. 8) describes a cutting plane approach. The procedure for computing the Chvátal-Gomory cut is left to an oracle, which computes a cut that a given nonintegral optimal solution (to the linear bilevel program) violates. It is, however, not implementable in general, as the oracle is NP-Complete. Cuts are generated and are added until the convex hull of the follower's problem is described well enough. Due to numerical issues this might result in premature termination with an infeasible solution, where the author suggests branch-and-cut procedures would be needed. Recently, DeNegre and Ralphs (2009) described a basic branch-and-cut algorithm. Dempe (2002) states that it would be desirable to solve a single level program where the lower level problem is described by a set of, perhaps heuristically generated, lower level solutions. This is especially true for the cases in which the follower's problem is a knapsack problem, where good approximations are readily available. This is similar to our approach for solving the competitive facility location.

Since this class of problems are quite hard, introduction of metaheuristics is inevitable. For instance, Wen and Huang (1996) describe a tabu search heuristic where they solve a bilevel linear problem in which integer variables appear only in the leader's decision. They iteratively generate integer solutions through the heuristic procedure and solve the follower's problem. Sahin and Ciric (1998) introduce a dual temperature (resembling the two level structure) simulated annealing approach to solve bilevel programs in general. They report experiments on a mixed integer (binary) bilevel nonlinear problem, which is a process layout design.

Many of the solution approaches are interested in quite restrictive types of problems. Jan and Chern (1994) solve a separable integer monotone bilevel programming problem with monotone objective and constraints, which are separable according to the leader and the follower variables.

They propose a parametric programming approach. The main idea is that they parameterize the leader's variables in the follower's problem and solve it for a number of different parameter values, hence dividing the problem into regions. Using the results, they solve the leader's problem and obtain candidate solutions. A similar approach is taken by Faisca et al. (2007).

Finally, some recent works are as follows. Gumus and Floudas (2005) introduce two approaches to solve mixed integer nonlinear bilevel programs. The first is for problems with mixed integer nonlinear leader and continuous nonlinear follower subproblems. The second is for problems with general mixed integer nonlinear functions in the leader's problem, and for the follower's problem, mixed integer nonlinear leader variables; linear, polynomial, or multilinear follower integer variables, and linear in continuous variables. Their technique is based on the reformulation of the mixed integer follower problem as continuous through its convex hull representation, perform linearization and solve the resulting single level problem by existing means. Saharidis and Ierapetritou (2008) apply Benders decomposition for mixed integer bilevel linear problems, where the integer variables are controlled only by the leader. The main idea is to fix the leader's variables and then combine Karush-Kuhn-Tucker conditions and an active set approach to turn the problem into a single level mixed integer problem. The solution reveals active constraints. Then the linear program with the active sets are solved and dual information generates a cut that is added to the leader's problem. In the most recent paper we found, Mitsos (2010) solve nonlinear mixed integer bilevel programs. He finds successively tighter lower bounds by solving a nonlinear mixed integer program, containing all constraints and parameterized upper bounds for the follower's objective.

1.1.2 Applications

Bilevel structures arise naturally in decentralized organizations. Bard (1983) discusses a multidivisional structure, in which top management is the leader and is followed by the divisions. Subordinate units have their individual objectives, which are not necessarily in line with the superior's. The top management can influence (coordinate) them with carefully taken decisions of their own.

Similarly, policy-making at the government level can be viewed from a bilevel programming perspective. Bard et al. (2000) models a government encouraging biofuel production through subsidies to the petro-chemical industry. In this model, farmers respond by deciding how much of such crops to produce in order to maximize their profit. Amouzegar and Moshirvaziri (1999) present a model in which a central authority provides off-site disposal facilities and sets the associated prices and taxes in order to minimize system (investment, transportation, processing) costs. In response, firms producing hazardous waste make their location-allocation and on-site recycling decisions.

In industry models, firms and customers can be modeled in a bilevel setting. For instance, Hobbs and Nelson (1992) applies bilevel programming in the electric utility industry for demand side management. Whereas firms control rates and subsidize energy conservation, customers take optimal positions in their consumption and investment in conservation. Cote et al. (2003) tackle the joint capacity allocation and pricing problem for the airline industry and hence represent customer behavior with regard to product attributes (i.e., fares). In a natural gas shipping case, Dempe et al. (2005) present a model in which a natural gas shipper maximizes profits. However, the second level problem is minimizing the cash-out penalties due to imbalances between contracted and delivered quantities at the time of delivery.

Bilevel programming is also applied in the modeling of direct conflicts between players. Interdiction problems are of this type. Israeli and Wood (2002) studies the problem of interdicting (destroy or increase cost) the arcs of an enemy network. The attacker's objective is to maximize the shortest path length, which is the enemy's optimization problem—between two nodes. Another interdiction scenario is studied in Scaparra and Church (2008), based on the classical p-median problem. The leader defends the system by selecting facilities to fortify, and an interdictor attacks a given number of facilities to maximize the damage.

Another area in which bilevel programming has attracted significant attention is traffic planning. These problems are similar to the first interdiction example above in construction, since the network use is optimized by the users (e.g. they find shortest paths). Some examples are mentioned below and a survey of these problems can be found in Migdalas (1995). Ben-Ayed et al. (1992) designs a highway network taking into account the reaction of users to improvements made by the designer. Labbe et al. (1998), on the other hand, considers the problem of revenue maximization through toll setting on highways. Drivers respond by altering their driving routes by finding the least-cost (time and money) travel path. In a similar study, Kara and Verter (2004) minimizes risk of transporting dangerous goods by selecting a subset of roads through which dangerous goods can be transported. Firms respond by minimizing their travel costs over the allowed roads.

Bilevel programming is also applied in chemical engineering and bioengineering in the design and control of optimized systems. For example, Clark and Westerberg (1990) optimize a chemical process by controlling temperature and pressure. The resulting system reaches equilibrium as it naturally minimizes Gibbs free energy and this constitutes the second level problem. Burgard et al. (2003) apply the idea to a bioengineering problem. They develop gene deletion strategies to facilitate overproduction of amino acids by the cell. The cell is the follower, responding by optimizing its growth objective subject to biochemical constraints.

1.2 Contributions and Dissertation Outline

Our novel contribution is a method to reformulate these problems so that they are more tractable (unlike the originals) and still bear a high representation power (like the originals). We reformulate the lower-level problems as a heuristic that is specifically developed for that type of problem and embed them into the leader's problem as linear constraints. The major benefit of heuristic lower-level reformulation is being able to solve the problem as an ordinary single-level problem, for which methods and software are ample. Moreover, the heuristics often describe the lower-level problem so accurately that the resulting upper-level solutions are near-optimal. Our computational experiments support these claims, demonstrating reasonably short solution times with low average errors.

This idea relates to the bounded rationality concept of economics. Bounded rationality, introduced by Simon (1955), emerged as an argument against the (unquestioned) assumption of rationality on the part of the decision makers in the 1950's and has become a cornerstone of decision theory since. It relies on empirical evidence regarding satisficing decision makers (employing heuristics), rather than optimizing ones, especially under circumstances where the optimizing capability or willingness of the decision maker is limited.

The study of network design under competition is interested in devising strategies for the leader players, and also in how certain networks would evolve without a central decision maker, especially if the network is important to other stakeholders that are not involved in the decision making process. In this dissertation, we present our research on three network problems involving competition.

Facility Location: In our first research topic, we study competition in a fundamental facility location problem—*maximal covering*. We develop a mathematical programming model to devise a reasonable strategy for a first mover firm, which engages in Stackelberg competition against a follower. The follower problem is described using a greedy heuristic and is encoded into the leader's problem as linear constraints. We present alternative formulations from different players' point of views and then investigate some analytical properties (e.g., bounds) of the method. Our computational analysis demonstrates, numerically, that the method suggests good—optimal in many cases—strategies.

Distribution Network Design: As our second research topic, we study competition in a distribution network design — prize-collecting Steiner tree — game. Each player wants to maximize her profit — node revenue minus arc costs. We develop a model for simultaneous-move setting that finds a pure strategy Nash equilibrium solution. Then we follow a similar approach as in the first topic and define the game as a leader-follower game and introduce a method to find reasonable and good strategies for the leader firm. Here, as a difference from the first topic, we restrict the decision space of the follower, but solve the problem optimally on this restricted space.

College Admissions: We discuss the university and job application process, which does not resemble a network design problem at first sight. The university application process, for instance, involves two parties — schools and students. A student applies to a group of schools and demonstrates her interest through costly signals (e.g., school visit) within a budget (e.g., time). These signals can increase the student's chances of being accepted. We devote the last study to the characterization of the optimal set of institutions to demonstrate interest. This constitutes a first step in the analysis of the game, which is expected to continue with the analysis of the school side in the future. We first show that this problem can be solved with a greedy algorithm. Then, we restate the problem as a longest path problem and propose a dynamic programming algorithm. This algorithm can be implemented using linear constraints, hence allows us to formulate the student problem exactly, which we plan to use in our future study.

The reformulation idea shows itself in three different ways in these problems. In the facility location case, we use a heuristic algorithm, keeping the follower decision space intact, except for the reduction caused by the constraints. In the distribution system design problem, we restrict the opponent to respond on a collection of trees, but the proposed formulation solves the restricted follower problem optimally. Finally, in the college admissions problem, the reformulation yields an optimal solution for the unrestricted problem and hence is capable of solving a bilevel problem whose lower level problem can be solved by the algorithm we proposed.

This dissertation is structured as follows. In Chapter 2, we present our research on the competitive maximal covering location problem. In Chapter 3, we study the simultaneous-move and Stackelberg versions of the competitive prize-collecting Steiner tree problem. In Chapter 4, we analyze the college admissions problem in light of demonstrated interest. Each chapter includes its own literature review regarding the discussed topic. Finally, Chapter 5 concludes with a summary of results and brief discussion of future research.

Chapter 2

A Competitive Facility Location Problem

2.1 Introduction

Facility location problems are pivotal in strategic decision making. Most firms face some sort of facility location problem, and these problems are often difficult to solve. For instance, the set covering, p-median, p-center problems are all in the complexity class NP-Hard. Classical facility location models tend to ignore the effect of competition on the location decision and assume there were a single decision maker. This is reasonable for locating public facilities like airports or fire stations, which have virtually no competitors. However, in a market, taking competitors' responses into account in the location decisions can significantly alter the resulting solutions and improve the realized outcomes.

In this chapter, we consider a competitive version of the Maximal Covering Location Problem (MCLP). There are two firms, who engage in a Stackelberg game, thus sequentially enter a new market. Each firm enters the market by locating multiple facilities, which are selected from a discrete set of potential facilities. Once the locations are decided, customers patronize their preferred facilities. The goal of each firm is to maximize the total customer demand that they serve at the end of the game. The second mover (i.e the follower) observes the first mover's (i.e. leader's) decision and responds optimally. We assume foresight on the part of the leader, and we solve her problem, taking the follower's optimal (i.e. best) response into account. The customers are represented as a discrete set of points and their demands are assumed to be deterministic.

The difficulty in solving this model comes from two sources. To begin with, MCLP is an NP-hard problem (Megiddo et al., 1983). Therefore, no polynomial time algorithm is known to solve it. However, this is not really the biggest problem, as large instances of these problems can be dealt with efficiently. The major difficulty in solving this problem is that it has a bilevel structure, with conflicting objectives, and each of the levels can be viewed as an instance of MCLP, given the other.

Our approach aims to overcome this difficulty by replacing the follower's problem with a reasonable heuristic, embedding this heuristic into the leader's problem as constraints, and thus solving the initially bilevel problem as a single-level problem.

We provide an introductory review of the literature on facility location problems that deal with competition in Section 2.2. In Section 2.3, we introduce our problem and our approach to modeling and solving it. We present our model in Section 2.4 and discuss two alternate formulations. Section 2.5 briefly analyzes the worst-case performance of our approach, while Section 2.6 provides computational results on the solution performance and the quality of the solutions generated by the proposed models. Section 2.7 concludes.

2.2 Literature Review

In this section, we provide a concise summary of significant works on competitive facility location and draw a historical timeline introducing the predecessors of our model. Facility location problems are extensively studied and the literature on them is vast, and a general review of the facility location literature is outside the scope of this chapter. Interested readers may refer to the books by Daskin (1995), Drezner (1995) and Drezner and Hamacher (2002). Snyder (2010) and ReVelle and Eiselt (2005) provide broad overviews of facility location problems. ReVelle et al. (2008) provide a recent bibliography on the fundamental problem types in location science. Owen and Daskin (1998) discuss mainly dynamic and stochastic problems in facility location. Snyder (2006) discusses stochastic and robust facility location models. Klose and Drexl (2005) discuss mixed integer programming models in a wide range of facility location problems.

We can describe competitive facility location problems as any facility location problem that incorporates spatial competition among at least two firms. The decisions of these firms are interdependent and affect each other's market share. Competitive problems within the facility location field can be traced back to the seminal work by Hotelling (1929). Hotelling introduced a market that is represented by a line segment. Two firms, whose locations on this market are known, compete by setting their prices to maximize their profits. He also discussed a case in which the first firm's location is fixed. The result asserts that the second firm locates itself arbitrarily close to the first one, which extends to the sameness of the product features and the positions taken by political parties. Considering location as an attribute of the product (e.g., store) gave rise to a plethora of studies. Hotelling's model has been revisited several times with different settings for strategic variables, moving sequences, customer preferences and number of firms; including a correction by d'Aspremont et al. (1979). Reviews of these models can be found in Gabszewicz and Thisse (1992), Anderson et al. (1992), Eiselt (2011), and Younies and Eiselt (2011). These location models on a line are very effective in providing theoretical insights about spatial competition but their representative power fails to deal with the facility location problems that operations research mainly works on. Therefore, in the early 1980's several studies were introduced to incorporate competition among decision makers into the classical facility location problems.

One of the earliest attempts was by Hakimi (1983), who extended the linear decision space in the Hotelling model to a network. He studied locating a fixed number of new facilities on a network where there were some existing facilities. Customers are located on the vertices and they have associated weights. Each customer prefers the closest open facility. Finally, the decision maker's objective is to maximize the weighted sum of customers who prefer his facilities.

Eiselt et al. (1993) provide a five-criteria (space, number of players, pricing policy, number of players and behavior of customers) taxonomy of competitive facility location problems and classify more than one hundred articles accordingly. Several papers survey parts of this classification. Plastria (2001) presents a survey on static competition in facility location and a large section on classification. Eiselt and Laporte (1996) report on sequential models, mostly of the leader–follower type of facility location problems. Serra and ReVelle (1995) provide a discussion of competitive location in discrete space, which also brings together several of the models that will be explained below. Most recently, Kress and Pesch (2012) surveyed recent developments in sequential competitive location problems.

The problem we address originates from the maximal covering location problem (MCLP) introduced by Church and ReVelle (1974). In the classical MCLP, customers are represented as demand points and they can be served only by a facility within a pre-specified service distance. The decision maker locates a fixed number of facilities in order to maximize the total demand covered by the facilities. The MCLP is one of the pillars of facility location and numerous variants have been studied. Snyder (2011) reviews covering problems including a detailed discussion of the MCLP.

Note that this original problem does not consider competition. ReVelle (1986) extends the MCLP to the competitive domain and introduces the maximum capture (MAXCAP) problem. This is an example of static competition. The decision maker locates a fixed number of facilities in a market where competing facilities already exist, just as the follower in our problem does. Each customer prefers the closest open facility and the demand is equally shared if the closest open facilities of each firm are at equal distance from the customer. This problem is, in fact, equivalent to the p-median problem as covering, capturing and sharing are expressable as the weight of an edge between a facility and a customer. In a subsequent paper Serra et al. (1996) consider demand uncertainty for the MAXCAP problem.

Serra and ReVelle (1994) reformulate the MAXCAP as a Stackelberg-type problem (competition with foresight). The game described in the MAXCAP stays the same but now the leader's problem is solved. This model is named the preemptive capture problem. For reasons we explain in the coming sections, this problem is quite hard to formulate and solve as an IP. Serra and ReVelle describe a simple heuristic procedure to solve the problem. The leader locates her facilities first. The follower responds by solving the MAXCAP problem. Then the leader follows a one-opt procedure, exchanging a single open facility with a closed one at each iteration and re-solving for the follower's response. The new solution is accepted if the leader's demand increases and the procedure is continued until no more improvements are achieved. An interesting variant of the problem is described by Serra and ReVelle (1995), in which the number of facilities to be opened by the opponent is defined probabilistically, and the probability monotonically decreases as the number of leader facilities increases.

Finally, Plastria and Vanhaverbeke (2008) revisit the problem and formulate three MIP models, robust models MaxMin, MinRegret, and Stackelberg, where they restrict the follower decision to a single facility. This assumption allows them to formulate the problem without needing a bilevel formulation. In the models we present below, we extend their Stackelberg approach to allow multiple follower facilities. However, we do not solve for an exact Stackelberg-Nash equilibrium; rather, we describe the follower response heuristically.

From a mathematical programming perspective, our problem is a 0-1 *integer bilevel linear programming problem* (IBLP), where the upper and lower levels correspond to the leader's and the follower's decision stages, respectively. (See Colson et al. (2007) for a recent review of bilevel programming.) There are a few solution approaches, including classical branch-and-bound (Bard and Moore, 1992; Moore and Bard, 1990), cutting planes, and branch-and-cut variants (DeNegre and Ralphs, 2009), that have been proposed to solve problems of this type. These algorithms typically struggle when applied to problems with more than a few variables and constraints. More efficient algorithms exist either for problems with continuous variables (which is still an NP-hard optimization problem (Jeroslow, 1985)) and problems with special exploitable structures. In general, the existing practical needs for IBLP significantly exceed current solution capabilities (Bard, 2009).

We propose to reformulate the problem as a single-level 0-1 integer linear problem, and represent the follower's problem by a heuristic that is formulated using additional variables and constraints, whose cardinality is polynomial in the original number of variables and constraints. As the heuristic does not guarantee optimal solutions for the follower, the result is not guaranteed to be optimal for the leader. On the other hand, computational experiments show that the model performs quite well in terms of computational effort and solution quality.

2.3 The Problem

Before discussing the model, we introduce the problem and related terminology. Then, we describe the mechanics of the underlying Stackelberg game and how we embed the follower's problem into the leader's constraints.

In our model, there are two competing firms, the leader (L) and the follower (F). Both firms have a number of potential locations where they can build and operate facilities to serve customers. We denote the leader's potential locations with $s \in S$ and the follower's potential locations with $t \in T$. We assume $S \cap T = \emptyset$ for the moment and in the numerical analysis in Section 2.6. However, this assumption can be altered without changing the model as we explain later in this chapter. The leader and the follower, respectively, open B and K facilities. Customer demand is assumed to accumulate at a number of discrete points and we denote these points with $i \in I$ and the associated demand with d_i. Both firms would like to maximize their profits, thus, the total demand they serve.

We define two relations among the components: *coverage* and *preference*. It is customary to use a distance analogy for these relations. An open facility *covers* all the customers that are at most a given service distance (i.e. radius) away from it. Each customer *prefers* the closest open facility that covers her. If there is no open facility that covers a particular customer then that customer is not served. Figure 2.1 illustrates a simple instance and the coverage relationship among problem entities on a grid assuming a radius of 5 units. We represent these relations as subsets. For a given customer $i \in I$ the leader [follower] locations that cover her are denoted as the subset $S_i \subseteq S$ [$T_i \subseteq T$]. We denote the set of leader facilities that customer i would prefer over the follower location t by S_{it}.

The model assumes that the leader is the first mover. She first makes her decision and locates

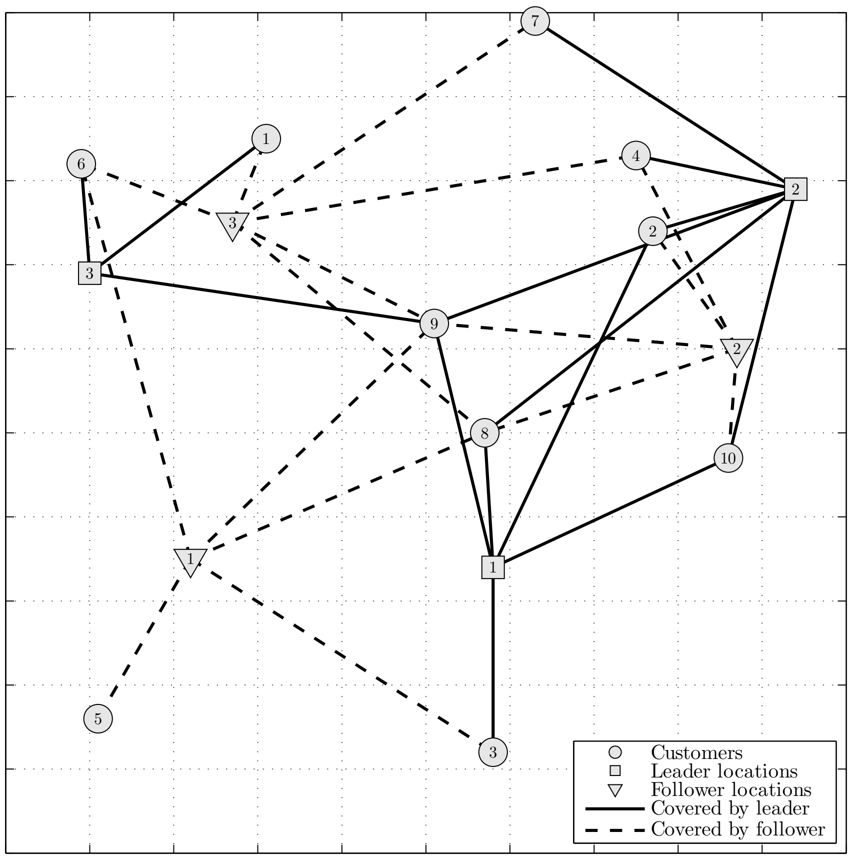


Figure 2.1: Customers, potential facilities and coverage relations

her facilities; then, after some time, the follower enters the market and locates his facilities. The follower's problem is to maximize the demand he serves, given the leader's solution. When both parties locate their facilities, the winner for each customer demand is decided according to the outcome matrix presented in Table 2.1. The winner firm is said to *capture* the customer. Figure 2.2 illustrates the optimal solution to the specified problem for $B = K = 2$, and the customers captured by each player.

Since the leader is not going to alter her solution after the follower acts, the follower does not need to take into account any strategic response and his problem is relatively easy. It reduces to the MCLP. On the other hand, the problem of the leader is a lot harder as she has to take into account her opponent's strategic response, too. This is difficult for two reasons. The follower, even though

Table 2.1: Outcome matrix for customer i :
Winner is Leader (L), Follower (F), none (X) or situation is impossible (I)

		Leader, s		
			$s \in S_i$	
		$s \notin S_i$	$s \in S_{it}$	$s \notin S_{it}$
Follower, t	$t \notin T_i$	1 X	2 L	3 I
	$t \in T_i$	4 F	5 L	6 F

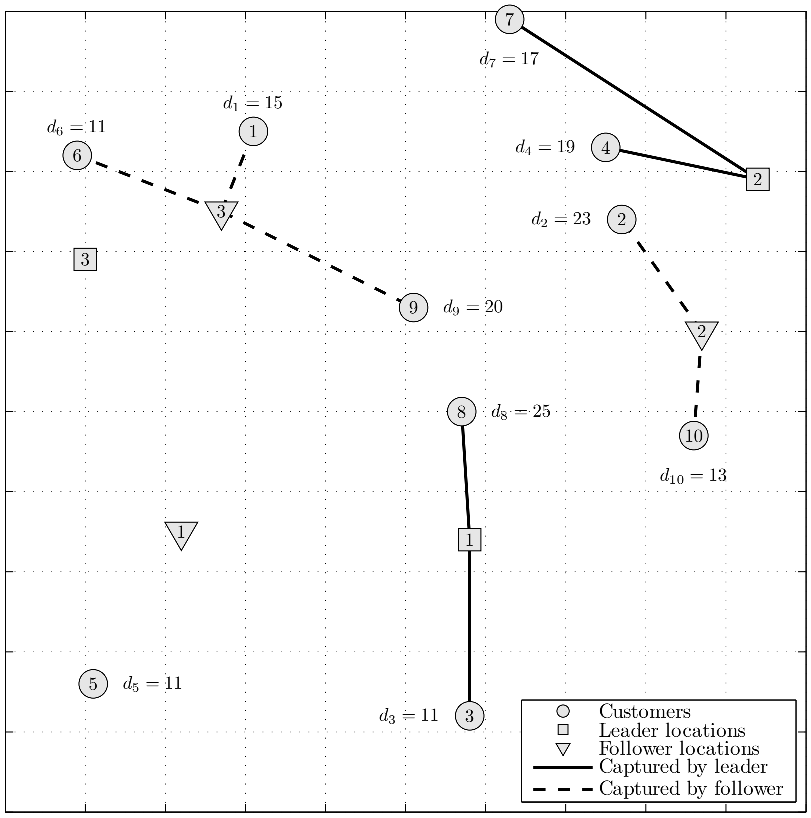


Figure 2.2: Solution: Opened facilities and captured customers

we restrict him to open exactly K facilities, needs to solve a combinatorial optimization problem which is known to be in the complexity class NP-hard. As almost all discrete facility location problems (including the MCLP) are in this category, this is not surprising. The bad news is that the follower's objective conflicts with the leader's, yet has to be optimized in the leader's problem. Since we are not able to put the follower's objective in the objective function next to the leader's, we ensure that the follower's problem is solved optimally through constraints, basically by making each suboptimal follower solution infeasible to the leader's problem, as follows:

$$\begin{aligned} \max \quad & LeaderCapture(L, F) && && (2.1) \\ \text{s.t.} \quad & \ldots && && (2.2) \\ & FollowerCapture(L, F) \geq FollowerCapture(L, F') && \forall F' && (2.3) \end{aligned}$$

Here, L and F denote the leader's and the follower's location vectors, respectively. The follower has $C(|T|, K)$ possible choices. This value is very large for large $|T|$ and gets larger as K approaches $|T|/2$. Therefore the number of required constraints (2.3) increases very fast, becoming hard to manage even for problems of moderate size. Plastria and Vanhaverbeke (2008) solve this problem by setting $K = 1$. They assume that the follower responds by opening a single facility, therefore keeping the number of constraints small.

Instead, we propose the following approach. We don't solve the follower's problem optimally, but replace it with a heuristic solution. We assume that the follower implements the *greedy add algorithm* given the locations of the leader facilities. The algorithm is described in Section 2.4.2. The leader views this heuristic as a proxy for her opponent's true optimal response, thus maximizes the total demand covered by her facilities assuming that the follower responds by applying his heuristic. We now introduce the rest of the notation and explain the proposed model and the accompanying heuristic.

2.4 The Model

2.4.1 Terminology and Notation

We revisit the sets and parameters, and introduce the variables of the model.

Sets

I**:** customers

S**:** potential leader facilities. Subset S_i denotes the set of facilities that can serve (*cover*) customer $i \in I$. Subset S_{it} denotes the set of leader facilities that customer i would prefer to the follower facility t.

T**:** potential follower facilities. Subset T_i denotes the set of follower facilities that cover customer $i \in I$.

Parameters

d_i**:** demand of customer $i \in I$

B**:** number of facilities that the leader will open

K**:** number of facilities that the follower will open, hence the number of greedy add algorithm iterations.

Decision Variables

L_s**:** binary variable that equals 1 if the leader opens facility s.

F_{tk}**:** binary variable that equals 1 if the follower opens facility t at the k^{th} iteration of the greedy add algorithm.

x_i^L**:** binary variable that equals 1 if customer i is captured by the leader.

x_{ik}^F**:** binary variable that equals 1 if customer i is captured by the follower facility opened at the k^{th} iteration of the greedy add algorithm.

y_{itk}: binary variable that equals 1 if customer i is still capturable by the follower facility $t \in T_i$ in the k^{th} iteration of the greedy add algorithm (definition of "capturable" to be made more precise later).

Although we define the last three variables as binary variables, we will show that if they are defined as continuous variables, the problem will still have an all-integer solution.

2.4.2 Follower's Response

Let us now analyze our assumption on the follower's response. The greedy add procedure (Kuehn and Hamburger, 1963) is as follows. At the beginning, all follower locations are closed. The follower receives the decision vector $L = [L_s]$ from the leader and updates the y variables. If a customer i can be covered by a particular follower facility t, then the customer is still *capturable* at iteration k ($y_{itk} = 1$) if there is no better open leader facility, otherwise it is not ($y_{itk} = 0$). Then for each facility that is still closed, the follower calculates the total demand that the facility could capture if it were open ($\sum_{i \in I} d_i y_{itk}$). The facility that can capture the largest demand (t') is opened. The customers who were capturable by this facility are captured ($x_{ik}^F \leftarrow 1$) and are removed from further consideration ($y_{itk'} \leftarrow 0$ for $k' > k$). This is repeated until the follower opens K facilities. This algorithm is included as a feasibility problem through constraints in the leader's model, as described in Section 2.4.3.

We selected this algorithm as the follower's response mainly for two reasons. First, the greedy add algorithm is a well-known procedure for facility location problems. The quality of the solution it provides depends heavily on the characteristics of the underlying graph for the follower's problem. It does not guarantee optimality in the majority of the cases, but worst case bounds exist. The second reason is that it enables us to embed the follower's response into the leader's problem using a polynomial number of new constraints. (In contrast, recall from Section 2.3 that a straightforward embedding of the follower's problem into the leader's would require an exponential number of constraints.) In the computational study, we discuss this and the effectiveness of the approach. Next we introduce and explain our mathematical model.

2.4.3 Leader's Problem

The leader aims to maximize the demand she captures. The objective function is therefore

$$\sum_{i \in I} d_i x_i^L, \tag{2.4}$$

the total demand captured by the leader. The leader is going to open at most B facilities, enforced with the constraint

$$\sum_{s \in S} L_s \leq B. \tag{2.5}$$

Note that B can also be interpreted as an investment budget if we introduce a fixed cost of investment for each facility. The greedy add algorithm requires exactly one follower facility to be opened at each iteration $k = 1, \dots, K$.

$$\sum_{t \in T} F_{tk} = 1 \quad \forall k = 1, \dots, K. \tag{2.6}$$

A particular follower facility $t \in T$ can be opened at most once throughout the algorithm.

$$\sum_{k=1}^{K} F_{tk} \leq 1 \quad \forall t \in T. \tag{2.7}$$

A particular customer $i \in I$ can be captured by at most one of the firms.

$$x_i^L + \sum_{k=1}^{K} x_{ik}^F \leq 1 \quad \forall i \in I. \tag{2.8}$$

The leader has to have an open and covering facility to capture customer i. That is, $x_i^L = 1 \Rightarrow \exists s \in S_i$ such that $L_s = 1$, or equivalently $L_s = 0 \; \forall s \in S_i \Rightarrow x_i^L = 0$.

$$x_i^L \leq \sum_{s \in S_i} L_s \quad \forall i \in I. \tag{2.9}$$

The above requirement is valid for the follower, too. $x_{ik}^F = 1 \Rightarrow \exists t \in T_i$ such that $F_{tk} = 1$ or equivalently $F_{tk} = 0 \; \forall t \in T_i \Rightarrow x_{ik}^F = 0$.

$$x_{ik}^F \leq \sum_{t \in T_i} F_{tk} \quad \forall i \in I, k = 1, \ldots, K. \tag{2.10}$$

After the leader's decision is finalized we need to determine whether a customer i is still capturable by a covering facility $t \in T_i$. This is the initialization step of the greedy add algorithm. The following two constraints determine if a customer is still capturable by a facility. Given a customer i and a covering potential follower facility t, if the leader has no better facility than t, then i can be capturable by the follower by opening facility t. We denote this situation as $y_{it1} = 1$.

$$1 - y_{it1} \leq \sum_{s \in S_{it}} L_s \quad \forall i \in I, t \in T_i. \tag{2.11}$$

On the other hand, if the leader opens a better facility $s \in S_{it}$ then the follower cannot capture customer i through facility t. We denote this situation as $y_{it1} = 0$.

$$1 - y_{it1} \geq L_s \quad \forall i \in I, t \in T_i, s \in S_{it}. \tag{2.12}$$

For the remaining iterations ($k \geq 2$) we update the value of y_{itk} as follows: if i is initially capturable ($y_{it1} = 1$) and it is not captured by the follower yet ($\sum_{k'<k} x_{ik'}^F = 0$) it is still capturable ($y_{itk} = 1$).

$$y_{itk} \geq y_{it1} - \sum_{k'=1}^{k-1} x_{ik'}^F \quad \forall i \in I, t \in T_i, k = 2, \ldots, K. \tag{2.13}$$

A customer becomes noncapturable once it is captured.

$$y_{itk} \leq 1 - \sum_{k'=1}^{k-1} x_{ik'}^F \quad \forall i \in I, t \in T_i, k = 2, \ldots, K. \tag{2.14}$$

Finally, a noncapturable customer stays noncapturable.

$$y_{itk} \leq y_{i,t,k-1} \quad \forall i \in I, t \in T_i, k = 2, \ldots, K. \tag{2.15}$$

The greedy add algorithm is implemented through the following constraints. At each iteration k, the follower captures a total demand that is equal to the demand capturable by any single follower facility.

$$\sum_{i \in I: t \in T_i} d_i y_{itk} \leq \sum_{i \in I} d_i x_{ik}^F \quad \forall t \in T, k = 1, \ldots, K. \tag{2.16}$$

At iteration k, customer i is captured if the follower opens a covering facility t and the customer is capturable by t.

$$x_{ik}^F \geq F_{tk} + y_{itk} - 1 \quad \forall i \in I, t \in T_i, k = 1, \ldots, K. \tag{2.17}$$

On the other hand, customer i is not captured if the follower opens a covering facility t but the customer is not capturable by t.

$$x_{ik}^F \leq 1 - F_{tk} + y_{itk} \quad \forall i \in I, t \in T_i, k = 1, \ldots, K. \tag{2.18}$$

Taken together, (2.10), (2.17), and (2.18) ensure that $x_{ik}^F = 1$ if and only if i was captured by the follower facility opened at iteration k. Finally, the variables have the following restrictions.

$$L_s, F_{tk} \in \{0, 1\} \quad \forall s \in S; t \in T, k = 1, \ldots, K, \tag{2.19}$$

$$x_i^L, x_{ik}^F, y_{itk} \geq 0 \quad \forall i \in I; t \in T_i, k = 1, \ldots, K. \tag{2.20}$$

Theorem 1. *The formulation above ensures the outcomes of the game depicted in Table 2.1.*

Proof. Each of the six cases (cells) in the outcome matrix are satisfied as follows.

1. If $L_s = 0$ for all $s \in S_i$ and $F_{tk} = 0$ for all $t \in T_i$, then $x_i^L = 0$ and $x_{ik}^F = 0$ by constraints (2.9) and (2.10), respectively.

2. If $F_{tk} = 0$ for all $t \in T_i$ and $k = 1, \ldots, K$ and there is at least one $s \in S_i$ such that $L_s = 1$, then $x_{ik}^F = 0$ for all k by constraint (2.10), and $x_i^L \leq 1$ by (2.9) and (2.8). Then, $x_i^L = 1$ because the leader maximizes total captured demand.

3. The case is impossible because if $s \in S_i$ but $t \notin T_i$, customer prefers s to t, thus $s \in S_{it}$.

4. If $L_s = 0$ for all $s \in S_i$ and there is at least one $t \in T_i$ such that $F_{tk} = 1$, then $x_{ik}^F = 1$ by constraints (2.17) and (2.8).

5. If there is an $s \in S_{it}$ such that $L_s = 1$, for each $t \in T_i$ with $F_{tk} = 1$, then $y_{itk} = 0$ for such t by constraints (2.12) and (2.15). Then $x_{ik}^F = 0$ by (2.18). $x_i^L = 1$ as in case 2.

6. If there is a $t \in T_i$ such that $F_{tk} = 1$ but $L_s = 0$ for all $s \in S_{it}$ then $y_{it1} = 1$, and $y_{itk} = 0$ if $x_{ik'} = 1$ for some $k' < k$ by (2.14), or $y_{itk} = 1$ and $x_{i,k} = 1$ by (2.17). □

Theorem 2. *Variables x_i^L, x_{ik}^F and y_{itk} assume binary values in the optimal solution.*

Proof. We have $y_{it1} \in [\max\{0, 1 - \sum_{s \in S_{it}} L_s\}, 1 - \max_{s \in S_{it}}\{L_s\}]$ by constraints (2.11) and (2.12), and nonnegativity constraint (2.20). Thus, given binary L_s, each y_{it1} is assigned a binary value.

At iteration k', assume we have binary L_s, $F_{tk'}$ and $y_{itk'}$. If there is no $t \in T_i$ such that $F_{tk'} = 1$, $x_{ik'}^F = 0$ by constraint (2.10) and $y_{itk'+1} = y_{itk'}$. If there is at least one such t, then given binary $y_{itk'}$, constraints (2.17) and (2.18) imply that $x_{ik'}^F = y_{itk'} = 1$, thus binary. Therefore $y_{itk'+1}$ is also binary. Then, by induction, all x_{ik}^F and y_{itk} are binary.

If there is a k such that $x_{ik}^F = 1$, by constraint (2.8) $x_i^L = 0$. If there is no $s \in S_i$ such that $L_s = 1$, by constraint (2.9) $x_i^L = 0$. If neither of these conditions are true, $x_i^L = 1$ as in case 5 of Theorem 1. □

Theorem 3. *Any follower response enforced by the above model corresponds to a greedy add algorithm result.*

Proof. At each iteration k, the follower facility which can capture the largest remaining demand is opened. That is, $F_{tk} = 1 \Rightarrow t = \arg\max_{t'}\{\sum_{i\in I} d_i y_{itk}\}$. The converse is also true if the maximizer is unique. Otherwise, there would be some $t' \neq t$ such that $\sum_{i\in I} d_i y_{it'k} > \sum_{i\in I} d_i y_{itk}$. Then $\sum_{i\in I} d_i x^F_{ik} > \sum_{i\in I} d_i y_{itk}$ by (2.16). However, if $F_{tk} = 1$, by (2.17) and (2.18), $x^F_{ik} = y_{itk}$. These constraints are only for (i,t) pairs such that $t \in T_i$, yet constraints (2.6) and (2.10) imply that for all i such that $t \notin T_i$, $x^F_{ik} = 0$. Therefore, $\sum_{i\in I} d_i x^F_{ik} = \sum_{i\in I} d_i y_{itk}$, which is a contradiction.

Also it follows that all of the demand capturable by t is captured and no noncapturable demand is captured. A total of K facilities are opened in this manner and the result is a greedy add algorithm solution. $\square$

Note that the greedy add algorithm may return different solutions and total captured demands for the follower if one does not use a stable selection procedure between the multiple maximizers of an iteration. In the model stated above, if there are multiple maximizers at a step of the greedy add algorithm, then the optimal solution selects the one that would result in the greatest objective value for the leader. To avoid this optimistic outcome, we can introduce a stable selection procedure for the multiple maximizers of an iteration. An example procedure could be defining a precedence order for the follower (e.g., index of the facility) such that the ties are resolved according to that order (e.g., select the one with the least index). We can enforce such a selection rule by introducing the following constraint.

$$\sum_{i\in I} d_i y_{itk} - \sum_{i\in I} d_i x^F_{ik} + \alpha_t F_{tk} \leq \sum_{t'\in T} \alpha_{t'} F_{t'k} \quad \forall t \in T, k = 1, \ldots, K. \tag{2.21}$$

where $\alpha_t \in (0, \min_i\{d_i\})$ and $\alpha_t > \alpha_{t'}$ if and only if t precedes t' on the selection rule. This way the ties would be solved in favor of the first alternative according to the precedence order.

The leader solves Problem 4. We refer to the following formulation as CMCLP1.

Problem 4 (Competitive MCLP with Heuristic Follower Response)**.**

$$\max \quad \sum_{i\in I} d_i x_i^L \tag{2.4}$$

$$s.t. \quad \sum_{s\in S} L_s \le B \tag{2.5}$$

$$\sum_{t\in T} F_{tk} = 1 \qquad \forall k = 1,\dots,K \tag{2.6}$$

$$\sum_{k=1}^{K} F_{tk} \le 1 \qquad \forall t \in T \tag{2.7}$$

$$x_i^L + \sum_{k=1}^{K} x_{ik}^F \le 1 \qquad \forall i \in I \tag{2.8}$$

$$x_i^L \le \sum_{s\in S_i} L_s \qquad \forall i \in I \tag{2.9}$$

$$x_{ik}^F \le \sum_{t\in T_i} F_{tk} \qquad \forall i \in I, k = 1,\dots,K \tag{2.10}$$

$$1 - y_{it1} \le \sum_{s\in S_{it}} L_s \qquad \forall i \in I, t \in T_i \tag{2.11}$$

$$1 - y_{it1} \ge L_s \qquad \forall i \in I, t \in T_i, s \in S_{it} \tag{2.12}$$

$$y_{itk} \ge y_{itk-1} - x_{ik-1}^F \qquad \forall i \in I, t \in T_i, k = 2,\dots,K \tag{2.13}$$

$$y_{itk} \le 1 - \sum_{k'=1}^{k-1} x_{ik'}^F \qquad \forall i \in I, t \in T_i, k = 2,\dots,K \tag{2.14}$$

$$y_{itk} \le y_{itk-1} \qquad \forall i \in I, t \in T_i, k = 2,\dots,K \tag{2.15}$$

$$\sum_{i\in I} d_i y_{itk} \le \sum_{i\in I} d_i x_{ik}^F \qquad \forall t \in T, k = 1,\dots,K \tag{2.16}$$

$$x_{ik}^F \ge F_{tk} + y_{itk} - 1 \qquad \forall i \in I, t \in T_i, k = 1,\dots,K \tag{2.17}$$

$$x_{ik}^F \le 1 - F_{tk} + y_{itk} \qquad \forall i \in I, t \in T_i, k = 1,\dots,K \tag{2.18}$$

$$L, F \in \{0,1\} \tag{2.19}$$

$$x^L, x^F, y \ge 0 \tag{2.20}$$

2.4.4 Alternative Formulations

CMCLP1 has the following two features that we might alter to obtain better formulations. First, it does not recognize which follower facility captures the customer at a given iteration. Second, it does not assume transitivity in customer preference. It has a follower point of view in defining the capturability, which we can define alternatively from the leader's point of view using transitivity.

CMCLP2: Identifying The Capturing Follower Facility

In CMCLP1, there is no variable that identifies which follower facility captures a given customer at a given iteration. This information is retrievable using x_{ik}^F and F_{tk} once the problem is solved. We introduce the variable x_{itk}^F, which equals 1 if customer i is captured by facility t at iteration k. This alters the previous formulation slightly. For constraints, (2.8), (2.13), (2.14) and (2.16), we simply replace x_{ik}^F with $\sum_{t\in T_i} x_{itk}^F$ because only one facility can capture a given customer in a given iteration.

On the other hand, the new variable allows us to define stronger relationships between y, F and x^F. Facility t captures a customer only if it is open and the customer is capturable in the given iteration.

$$x_{itk}^F \geq F_{tk} + y_{itk} - 1 \quad \forall i \in I, t \in T_i, k = 1, \ldots, K \tag{2.22}$$

If any of these conditions is not satisfied, the customer cannot be captured by facility t.

$$x_{itk}^F \leq F_{tk} \quad \forall i \in I, t \in T_i, k = 1, \ldots, K \tag{2.23}$$

$$x_{itk}^F \leq y_{itk} \quad \forall i \in I, t \in T_i, k = 1, \ldots, K \tag{2.24}$$

Therefore, we can replace constraints (2.10), (2.17) and (2.18) with (2.23), (2.22) and (2.24), respectively. Finally, we can obtain an improvement in constraint (2.15), which can be altered as

$y_{itk} \leq y_{itk-1} - x^F_{itk-1}$. Introducing the new index increases the number of variables and constraints, but the tighter constraints lead to improved performance; see Section 2.6 for more details.

CMCLP3: Capturability From The Leader's Point Of View

CMCLP1 takes the follower's point of view in formulating the capturability of a customer—variable y. It relies on the set S_{it} that is defined for each (i, t) pair. Note that this set definition does not assume transitive customer preference among members of $S \cup T$, even though consumer preferences are generally transitive. If we assume transitivity, we can use an alternative and tighter formulation. Therefore, let $\succeq_i$ be a strict total preference relation (complete, reflexive, transitive and antisymmetric) over the choice set $S \cup T$. Let ties be broken (e.g. in favor of the smaller-indexed facility) before we apply the preference relation. We introduce a new variable a_{isk} which equals 1 if customer i patronizes the leader facility s before iteration k—in a sense the game is played in $1 + K$ stages, which represent the leader's move and the follower's K consecutive moves, respectively. Then we make the following changes in CMCLP1.

At the beginning of each iteration k, customer i is either not covered by any of the opened facilities, or patronizes a covering leader facility or has been captured by the follower at some earlier iteration.

$$\sum_{s \in S_i} a_{isk} + \sum_{k'=1}^{k-1} x^F_{ik'} \leq 1 \quad \forall i \in I, k = 1, \dots, K \tag{2.25}$$

At the leader's decision stage, a customer can only patronize an open leader facility.

$$a_{is1} \leq L_s \quad \forall i \in I, s \in S_i \tag{2.26}$$

When leader facility s is opened, the following two conditions hold. A covered customer cannot be assigned to a less preferred facility (2.27), but it can be assigned to s or a more preferred facility

(2.28).

$$L_s \leq 1 - \sum_{s' \in S_i : s \succ_i s'} a_{is'1} \quad \forall i \in I, s \in S_i \tag{2.27}$$

$$L_s \leq \sum_{s' \in S_i : s' \succeq_i s} a_{is'1} \quad \forall i \in I, s \in S_i \tag{2.28}$$

A customer is considered capturable by a given follower facility t if it is not currently held by a better leader facility and has not already been captured by the follower. At each step of the greedy add algorithm, the left-hand side of the inequality is the total demand capturable by each follower facility and the right-hand side equals the demand captured by the follower in that iteration. This guarantees that the demand captured at each iteration is the largest capturable demand by a single open facility.

$$\sum_{i \in I : t \in T_i} d_i \left(1 - \sum_{s \in S_i : s \succ_i t} a_{isk} - \sum_{k'=1}^{k-1} x_{ik'}^F \right) \leq \sum_{i \in I} d_i x_{ik}^F \quad \forall t \in T, k = 1, \ldots, K \tag{2.29}$$

Then, we have the following three counterparts of the previous model, ensuring that (2.29) holds at equality for the opened facility. At a given iteration k, if customer i is captured, a covering follower facility t should be open.

$$x_{ik}^F \leq \sum_{t \in T_i} F_{tk} \quad \forall i \in I, k = 1, \ldots, K \tag{2.30}$$

Thus, if customer i is capturable by t, and t or a better follower facility is open, i is captured (2.31). However, if it is not capturable by t, yet the opened follower facility is t or worse, then it cannot be captured (2.32). (This can be incorporated in the same way into CMCLP1 (but not to CMCLP2) by replacing F_{tk} in (2.17) and (2.18) with $\sum_{t' \in T_i : t' \succeq_i t} F_{t'k}$ and $\sum_{t' \in T_i : t \succeq_i t'} F_{t'k}$, respectively.)

$$x_{ik}^F \geq \sum_{t' \in T_i : t' \succeq_i t} F_{t'k} - \sum_{s \in S_i : s \succ_i t} a_{isk} - \sum_{k'=1}^{k-1} x_{ik'}^F \quad \forall i \in I, t \in T_i, k = 1, \ldots, K \tag{2.31}$$

$$x_{ik}^F \leq 2 - \sum_{t' \in T_i : t \succeq_i t'} F_{t'k} - \sum_{s \in S_i : s \succ_i t} a_{isk} - \sum_{k'=1}^{k-1} x_{ik'}^F \quad \forall i \in I, t \in T_i, k = 1, \ldots, K \tag{2.32}$$

The assignment status of customer i to a particular leader facility s, after the initial assignment in (2.27)–(2.28) is updated in one of the following three ways. If i is not assigned to (held by) s at a given iteration, it would not be in the subsequent iterations (2.33). If the leader was holding i and a better follower facility was not opened, the leader continues to hold it (2.34).

$$a_{isk} \leq a_{isk-1} \quad \forall i \in I, s \in S_i, k = 2, \ldots, K \tag{2.33}$$

$$a_{isk} \geq a_{isk-1} - \sum_{t \in T_i : t \succ s} F_{tk-1} \quad \forall i \in I, s \in S_i, k = 2, \ldots, K \tag{2.34}$$

The last assignment status is converted to the leader's updated covering relation as follows. If the leader does not hold the customer before the last iteration (2.35), or the follower has captured it (2.36), the leader cannot capture it. However, if both are false, the leader captures it (2.37).

$$x_i^L \leq \sum_{s \in S_i} a_{isK} \quad \forall i \in I \tag{2.35}$$

$$x_i^L \leq 1 - \sum_{k=1}^{K} x_{ik}^F \quad \forall i \in I \tag{2.36}$$

$$x_i^L \geq \sum_{s \in S_i} a_{isK} - x_{iK}^F \quad \forall i \in I \tag{2.37}$$

As a final note, $L, F \in \{0, 1\}$ whereas $a, x \in [0, 1]$.

2.5 Theoretical Performance

How accurate is the greedy add algorithm as an approximation of the follower's optimal response, and what impact does the inaccuracy have on the leader's decision? Let L and F be the leader's and follower's location vectors, respectively, and let $LC(L, F)$ and $FC(L, F)$ be the demand captured by the leader and the follower, respectively, for given values of L and F. Let $F^*(L)$ be the follower's optimal response to the leader locations L, and let $F^G(L)$ be the follower's response if he uses the greedy add algorithm. Finally, let L^* be the leader's optimal solution if she assumes that the follower acts optimally, and let L^G be the leader's optimal solution if she assumes that the follower uses the greedy add algorithm. We start with an ideal, and extreme, case, in which the model solves the bilevel program optimally.

Proposition 5. *If the customers covered by the follower facilities do not overlap,* $F^*(L) = F^G(L)$, $L^* = L^G$.

Proof. In this case, the follower's coverage network (e.g., Figure 2.1) is a collection of stars which have a follower facility as the internal node and customers covered by that facility as the leaves. The leader's decision can remove edges from this graph, but cannot add edges to it. Opening a follower facility does not affect the stars associated with other facilities. Since there is no overlap, summing the demand captured by individual facilities gives the objective. The objective is maximized by selecting the best K facilities in order, which is equivalent to the greedy add algorithm. Since the greedy add algorithm identifies the follower's best response, the model solves the bilevel program optimally. $\square$

This result is tight in the sense that the algorithm can fail to find the optimal solution even when each facility shares at most one covered customer, as Figure 2.3 shows.

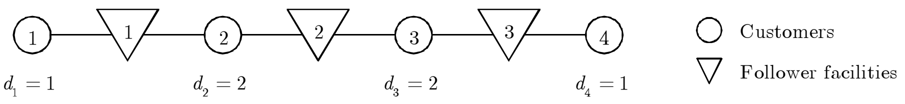


Figure 2.3: Greedy algorithm fails: $F^*(L) = [1, 0, 1], F^G(L) = [1, 1, 0]$

The following theorem states an upper bound on the worst-case performance of the greedy add algorithm as a heuristic for solving the follower's problem. It is an application of a result whose proof can be found in Hochbaum (1997).

Theorem 6. *If the follower opens K facilities, the following worst-case bound holds:*

$$\frac{FC(L,F^*(L)) - FC(L,F^G(L))}{FC(L,F^*(L))} \leq \left(1 - \frac{1}{K}\right)^K. \tag{2.38}$$

Moreover, $\lim_{K\to\infty}\left(1-\frac{1}{K}\right)^K = \frac{1}{e} \approx 0.37$, *and it approaches from below.*

Theorem 7. *The worst-case bound in Theorem 6 is tight.*

Proof. Worst-case examples can be generated by manipulating the the worst-case examples in Cornuejols et al. (1977). One such rule is as follows. Given K, let $|I| = K^2$, $|S| = K$, $|T| = 2K-1$ and $B = K-1$. Let each leader facility s cover customer $i = K(K-1)+s$, respectively. Let each follower facility $t < K$ cover customers $i \in \{(t-1)K+1, ..., (t-1)K+K\}$, and each $t \geq K$ cover customers $i \in \{t+1-K, t+1, t+1+K, ..., t+1+(K-2)K\}$. Assume the follower's facilities are all better than the leader's. Finally, let the customers' demand be $d_i = K^{K-2}(\frac{K-1}{K})^{\lceil \frac{i}{K} \rceil - 1}$ for $i \leq K(K-1)$ and $d_i = (K-1)^{K-1} - \epsilon$ otherwise, where ϵ is an arbitrarily small positive number. An illustration is presented in Figure 2.4.

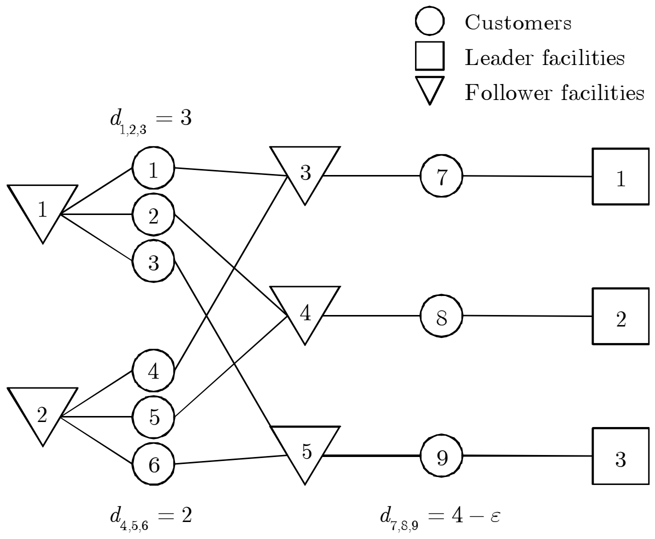


Figure 2.4: Worst case of greedy add algorithm, $K = 3$

The greedy add algorithm would select locations $t \in \{1, \ldots, K\}$ for each possible leader decision. However, the optimal follower decision is to open $t \in \{K, ..., 2K-1\}$ and capture all customer demand. This would give the following expressions for greedy and optimal objectives:

$$\begin{aligned} FC(L, F^G(L)) &= \sum_{i=1}^{K(K-1)} K^{K-2}(\frac{K-1}{K})^{\lceil \frac{i}{K} \rceil - 1} + (K-1)^{K-1} - \epsilon \\ &= \sum_{n=0}^{K-2} K^{K-1}(\frac{K-1}{K})^n + (K-1)^{K-1} - \epsilon \\ &= \sum_{n=0}^{K-1} K^{K-1-n}(K-1)^n - \epsilon = K^K - (K-1)^K - \epsilon \end{aligned}$$

$$\begin{aligned} FC(L, F^*(L)) = FC(L, F^G(L)) + \sum_{i=K(K-1)+2}^{K^2} \left[(K-1)^{K-1} - \epsilon\right] \\ = K^K - (K-1)^K + (K-1)^K - K\epsilon = K^K - K\epsilon \end{aligned}$$

Therefore, as $\epsilon \to 0$,

$$\frac{FC(L, F^*(L)) - FC(L, F^G(L))}{FC(L, F^*(L))} \to \frac{(K-1)^K}{K^K} = \left(1 - \frac{1}{K}\right)^K \tag{2.39}$$

$\square$

We can use this result to assess the worst-case performance of the greedy assumption from the leader's perspective. Our first such result is the following bound on the leader's loss as a percentage of demand that the follower greedily captures.

Theorem 8. *By assuming a greedy follower response and selecting facilities L, the leader loses at most $\frac{1}{1-\alpha} FC(L, F^G(L))$, where $\alpha = \left(1 - \frac{1}{K}\right)^K$.*

Proof. Let A be the subset of customers that the leader covers by selecting L, and B $[C]$ be the subset of customers that the follower captures if he responds greedily [optimally]. See Figure 2.5 for an illustration. Therefore, the leader expects to capture $A \setminus B$, but ends up capturing $A \setminus C$. This corresponds to losing $(A \cap C) \setminus B$ and gaining $(A \cap B) \setminus C$. The net loss is maximized if all [none] of the customers that the optimizing [greedy] follower captures are stolen from the leader; that is, if $(A \cap C) \setminus B = C$ and $(A \cap B) \setminus C = \emptyset$. The total demand of set C is $FC(L, F^*(L))$. Then, from Theorem 6, we obtain $FC(L, F^*(L)) < \frac{1}{1-\alpha} FC(L, F^G(L))$. □

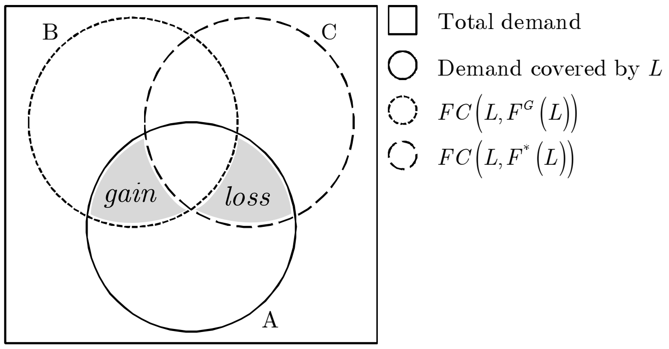


Figure 2.5: Set diagram for game outcome

On the other hand, no fixed worst-case bound exists for the leader's percentage error when she assumes a greedy response from the follower but the follower responds optimally. Similarly, there is no such bound for the leader's percentage error between the heuristic model solution and the optimal solution to the problem.

Theorem 9. *No fixed worst-case bounds exist for:*

(a) $\frac{LC(L,F^G(L)) - LC(L,F^*(L))}{LC(L,F^*(L))}$, *and*

(b) $\frac{LC(L^G,F^G(L^G)) - LC(L^*,F^*(L^*))}{LC(L^*,F^*(L))}$

Proof. For case (a), consider the worst-case example in Theorem 6 with $K = 3$, and let $L = [0, 1, 1]$. The greedy response of the follower is $F^G(L) = [1, 1, 1, 0, 0]$ and it yields $LC(L, F^G(L)) = 8$. If the follower responded optimally, he would pick $F^*(L) = [0, 0, 1, 1, 1]$ and it would yield $LC(L, F^*(L)) = 0$ since the follower facilities are better than the leader facilities for customers 7, 8 and 9. Thus, the leader expects to capture a positive demand whereas she cannot capture any demand in the end. This is illustrated in Figure 2.6.

$$\frac{LC(L, F^G(L)) - LC(L, F^*(L))}{LC(L, F^*(L))} = \infty.$$

For case (b), consider the same example, but now introduce two new leader locations $s = 4, 5$, and two new customers $i = 10, 11$ as in Figure 2.6. Let $S_{10} = 4, S_{11} = 5, T_{10} = T_{11} = \emptyset$, and each customer have a demand of $\gamma < 4 - \epsilon$. The solution is $L^G = [0, 1, 1, 0, 0]$, $F^G(L^G) = [1, 1, 1, 0, 0]$, $L^* = [0, 0, 0, 1, 1]$ and $F^*(L^*) = [0, 0, 1, 1, 1]$. These yield $LC(L^G, F^G(L^G)) = 8 - 2\epsilon$ and $LC(L^*, F^*(L^*)) = 2\gamma$. When γ approaches 0:

$$\frac{LC(L^G, F^G(L^G)) - LC(L^*, F^*(L^*))}{LC(L^*, F^*(L))} \to \infty.$$

$\square$

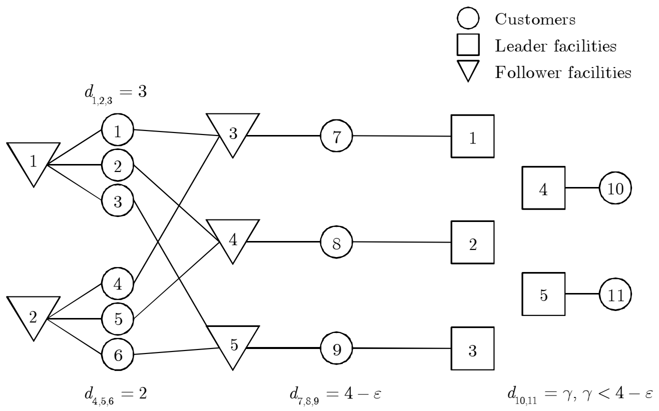


Figure 2.6: Worst case of greedy add algorithm, $K = 3$

These results are for the limited-coverage case. However, they still hold if we assume an infinite coverage radius, too. The follower always solves a limited-coverage problem, because the follower's coverage is updated (and becomes limited) once the leader makes his decision. The same worst-case examples can be generated for the infinite coverage case by carefully constructing the preference relations.

Finally, we present a modification of the proposed models. We will show that solving the models with this modification yields an upper bound for the demand that the leader captures. Such an upper bound is not difficult to obtain. For instance, if we solve the classical MCLP (i.e., no follower) for the leader, the result is clearly an upper bound on the leader's capture. Consequently, if we restrict the follower to open K facilities and capture customers according to the game rules, but do not attempt to optimize the follower's response, we obtain a tighter upper bound. In this case the leader can pick any follower response, but incur the associated demand loss.

Our modification is done as follows. First, we introduce variables $\hat{F}_t$ and $\hat{x}_i^F$. They indicate, respectively, whether a facility is opened and if a customer is captured by the follower. Constraint (2.40) requires the follower to open K facilities. Constraints (2.41)–(2.44) ensure that the follower takes all the facilities that it can capture with the open facilities and no more. Finally, (2.45) ensures that the opened facilities capture at least as much as the greedy follower response. The modification is done by replacing (2.8) in Model 4 with (2.44) below and adding the remaining constraints to the model. We show that the modified model is an upper bound on the leader's capture in Theorem 10.

$$\sum_{t \in T} \hat{F}_t = K \tag{2.40}$$

$$\hat{x}_i^F \le \sum_{t \in T_i} \hat{F}_t \qquad \forall i \in I \tag{2.41}$$

$$\hat{x}_i^F \ge \hat{F}_t + y_{it,1} - 1 \qquad \forall i \in I,\, t \in T_i \tag{2.42}$$

$$x_i^L \ge L_s - \sum_{t \in T_i : s \notin S_{it}} \hat{F}_t \qquad \forall i \in I,\, s \in S_i \tag{2.43}$$

$$\hat{x}_i^F + x_i^L \le 1 \qquad \forall i \in I \tag{2.44}$$

$$\sum_{i \in T} \sum_{t \in T_i} \sum_{k=1}^{K} d_i x_{itk}^F \le \sum_{i \in I} d_i \hat{x}_i^F \tag{2.45}$$

Theorem 10. *The modified model is an upper bound on the leader's optimal capture* $LC(L^*, F^*(L^*))$.

Proof. The modification below allows the leader to pick any of the follower's K facilities as long as it is better than (or equal to) the greedy follower response given L. Since the follower's optimal response is always better than or equal to his greedy response, the solution to the modified problem is always greater than or equal to leader's optimal capture $LC(L^*, F^*(L^*))$. $\square$

2.6 Computational Study

2.6.1 Experimental Design

We tested our model using two data sets from Daskin (1995). The first set includes the geographical coordinates and populations of 88 cities. These are the 50 largest US cities according to the 1990 census and the 48 continental state capitals, less the double entries. The second set includes the geographical coordinates of the 150 largest US cities according to the 1990 census. The first set represents the customers, thus $|I| = 88$, and we used their populations as customer demands. We randomly selected the potential facilities from the second set. We selected 11 settings for the number of potential facilities, changing $|S|$ and $|T|$. We further decided on 20 problem types by changing the model size parameters (K, B) and radius (r). The radius (given in miles) is the key to generating the coverage and preference relationships. Each location is assumed to have a coverage radius given by r, and each customer prefers facilities in order of distance. For each problem type, we generated 50 instances by randomly picking potential leader and follower locations. We discarded duplicate locations, thus each instance consists of distinct locations for the leader and the follower. In total, we solved 1000 instances. The problem types and their properties are summarized in Table 2.2.

2.6.2 Solution Statistics

We coded the model using AMPL and solved the instances using CPLEX 12.2 on a Pentium Xeon 3.0 GHz (x2) 64 bit computer. We used a 1 hour (3600 second) time limit per problem. All of the instances were solved within the time limit. The statistics are tabulated in Tables 2.3 and 2.4. The leftmost column indicates the instance type. Under a given instance type, rows 1-3 correspond to the results for formulations CMCLP1-3, respectively. The values are the aggregated results from 50 instances of each instance type. We report the CPU time used to solve the problem (Solution

Table 2.2: Instance settings

Type	$\|I\|$	$\|S\|$	$\|T\|$	K	B	r
T1	88	10	10	4	4	300
T2	88	10	20	4	4	300
T3	88	20	10	4	4	200
T4	88	20	10	4	4	300
T5	88	20	10	4	4	400
T6	88	20	10	4	4	500
T7	88	20	10	1	5	300
T8	88	20	10	2	5	300
T9	88	20	10	3	5	300
T10	88	20	10	4	5	300
T11	88	20	10	5	5	300
T12	88	20	20	4	4	300
T13	88	20	20	4	4	400
T14	88	20	30	4	4	300
T15	88	30	10	4	4	300
T16	88	30	20	4	4	300
T17	88	30	30	4	4	300
T18	88	40	10	4	4	300
T19	88	40	20	4	4	300
T20	88	40	20	4	8	300

Time), number of simplex iterations (Simplex Iterations), and number of branch and bound nodes evaluated (Node Evaluations). We compare the LP relaxation at the root node and the optimal MIP solution and report the percent integrality gap (%LPGap) calculated as $\frac{(z_{LP}-z_{MIP})}{z_{MIP}}$. For each of these performance measures, we report the minimum, maximum and average of the results.

We isolate the effects of the problem parameters and illustrate them in Figures 2.7 and 2.8 below. Comparing T1, T4, T15, T18 with T2, T12, T16, T19, and T4, T12, T14 with T15, T16, T17, we see that an increase in the number of potential leader and follower facilities increases the average solution time. Their cross comparison shows that at higher levels of the other parameter, this increase is magnified. These comparisons are plotted in Figure 2.7. If we compare T7-T11 we notice also that the number of follower facilities to open affects the solution effort. An increase in the number of facilities multiplies the size of the instance and the solution time. Another important

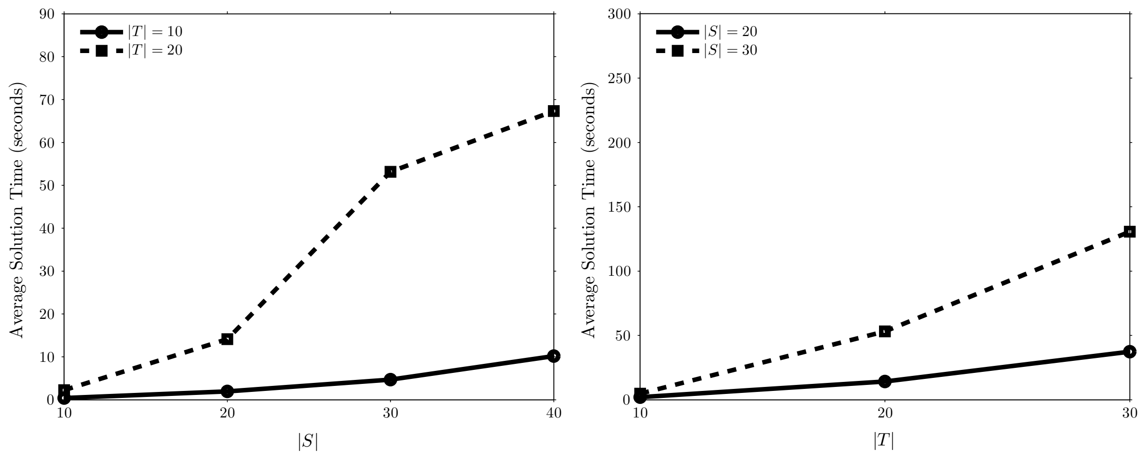
Table 2.3: Solution statistics: Comparison of formulations

		Solution Time			Simplex Iterations			Node Evaluations			%LPGap		
		Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
T1	1	0.41	0.04	2.10	728	0	5817	15	0	175	16.4	0.0	42.7
	2	0.39	0.04	2.79	686	0	8723	10	0	145	15.5	0.0	40.2
	3	0.39	0.04	1.44	778	0	5038	13	0	128	14.8	0.0	40.2
T2	1	2.94	0.19	38.58	4245	53	54582	90	0	908	38.6	11.9	77.3
	2	1.82	0.15	11.98	2681	35	19697	28	0	158	37.2	11.7	75.1
	3	1.96	0.24	9.24	2234	79	11724	42	0	208	35.0	7.2	70.8
T3	1	0.21	0.02	0.98	335	0	2106	6	0	58	9.6	0.0	32.9
	2	0.22	0.02	0.98	316	0	1898	4	0	27	8.5	0.0	32.9
	3	0.27	0.04	0.94	571	0	3461	14	0	116	7.8	0.0	32.9
T4	1	2.11	0.13	11.83	4589	254	25430	98	0	411	14.9	1.2	39.8
	2	1.70	0.18	7.20	3757	120	25484	51	0	245	14.3	1.2	36.0
	3	2.06	0.17	8.84	5522	175	35199	88	0	395	14.0	1.2	35.7
T5	1	17.15	0.25	103.52	30708	365	164064	369	0	1672	18.9	7.4	43.8
	2	11.23	0.28	66.61	20083	357	109892	153	0	820	18.5	7.4	43.4
	3	16.81	1.34	116.48	33436	2076	197455	288	5	1249	18.3	7.4	41.3
T6	1	70.90	4.86	349.41	99338	12259	544882	750	139	2264	20.7	7.3	40.2
	2	45.61	6.72	268.24	62933	8514	373206	318	68	1351	20.4	7.3	40.1
	3	62.64	8.74	316.26	97645	10874	502173	541	82	1901	20.1	7.3	40.1
T7	1	0.11	0.01	0.51	151	16	593	2	0	32	3.3	0.0	24.6
	2	0.08	0.01	0.42	144	0	748	2	0	31	2.9	0.0	20.7
	3	0.11	0.02	0.61	325	30	2183	8	0	92	3.3	0.0	23.1
T8	1	0.28	0.02	0.92	818	36	4416	26	0	153	7.6	0.0	17.6
	2	0.27	0.03	1.20	618	0	3350	14	0	90	6.9	0.0	17.5
	3	0.44	0.03	1.93	1276	0	4672	37	0	167	6.7	0.0	16.4
T9	1	0.74	0.06	3.75	2211	109	15442	61	0	539	10.9	0.2	36.9
	2	0.67	0.06	2.57	1595	0	9383	27	0	146	10.3	0.2	35.9
	3	0.94	0.08	3.04	2849	111	11079	57	0	213	9.9	0.2	33.0
T10	1	2.62	0.16	27.45	6888	170	70071	148	0	1086	12.8	3.8	33.5
	2	1.76	0.14	9.81	4595	100	41966	70	0	627	12.3	3.8	33.2
	3	2.28	0.15	7.92	6507	238	29673	121	0	562	12.0	3.7	33.2

Table 2.4: Solution statistics: Comparison of formulations *(continued)*

		Solution Time			Simplex Iterations			Node Evaluations			%LPGap		
		Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
T11	1	5.52	0.34	39.28	11986	473	71108	218	0	922	14.0	3.4	31.1
	2	4.03	0.41	18.12	9045	601	37972	116	0	474	13.6	3.4	30.5
	3	5.00	0.32	31.61	12758	666	78364	169	0	856	13.3	3.4	30.5
T12	1	18.90	0.73	82.16	31605	784	134944	539	3	2146	29.7	6.7	78.7
	2	11.95	0.57	55.30	21889	849	118361	217	0	1327	29.0	6.7	78.7
	3	11.52	1.15	61.31	24085	1654	131407	347	8	2591	28.5	6.7	78.6
T13	1	201.72	12.33	706.81	212994	11668	663059	1789	180	5911	32.6	12.5	68.2
	2	128.74	12.87	501.36	144339	10525	643126	682	57	3570	31.9	12.5	66.8
	3	117.99	9.78	431.41	175111	16026	648300	1255	156	3670	31.9	12.5	67.4
T14	1	50.32	1.42	284.75	63508	2475	386579	848	9	4019	39.6	10.4	74.0
	2	37.24	1.99	210.87	51574	3678	254825	368	27	1445	39.0	10.3	73.6
	3	24.54	1.46	119.57	41508	1880	150910	585	16	2168	38.0	10.3	68.3
T15	1	4.62	0.13	54.46	11152	211	129107	199	0	1489	13.3	3.2	27.8
	2	3.59	0.14	47.80	8963	221	108045	113	0	905	12.9	2.8	27.8
	3	5.88	0.40	45.21	15202	815	112504	205	0	1141	12.7	0.1	27.8
T16	1	69.10	0.94	640.31	100937	1408	596622	1354	31	8284	26.9	11.4	61.5
	2	50.90	1.01	557.96	81098	1540	655455	660	32	3361	26.3	11.4	59.9
	3	39.43	1.43	123.70	74529	2299	216763	909	60	3277	25.8	11.4	59.6
T17	1	160.04	10.32	725.09	180034	11939	655482	1913	150	6987	30.5	11.4	73.3
	2	139.64	8.83	522.29	150563	10405	528393	948	74	2979	30.1	11.4	72.0
	3	92.12	6.91	283.64	132119	8455	395635	1385	97	5053	29.6	11.3	72.7
T18	1	8.61	0.30	41.63	22097	475	85834	413	0	1432	13.6	3.2	40.0
	2	6.35	0.43	32.43	16636	600	91117	239	0	1494	13.3	3.2	38.2
	3	15.60	0.82	63.65	35319	982	140768	452	7	1729	13.2	3.2	35.7
T19	1	71.52	0.52	250.49	112918	754	420088	1449	0	6047	23.8	1.0	53.6
	2	61.64	0.53	217.65	101998	657	374626	842	0	3090	23.4	1.0	53.6
	3	68.78	0.61	194.02	112543	920	357041	1181	0	3689	23.1	1.0	53.6
T20	1	206.50	5.45	946.11	373380	13710	1694477	4717	265	31368	14.2	3.1	25.6
	2	168.39	4.46	1012.34	316145	10061	2032369	2868	129	17995	14.0	3.0	25.6
	3	250.67	15.20	1268.38	489369	21078	2691435	5578	339	35269	13.9	3.0	25.6

factor is the coverage radius of the facilities. Although not a model parameter, it is the main factor in determining the coverage relationship. Increasing the radius increases the number of covering facilities, thus supply alternatives, for each customer. This generates a denser graph and the problem size increases, as comparing T3-T6 shows. The effects of the number of follower facilities and the coverage radius are plotted in Figure 2.8.

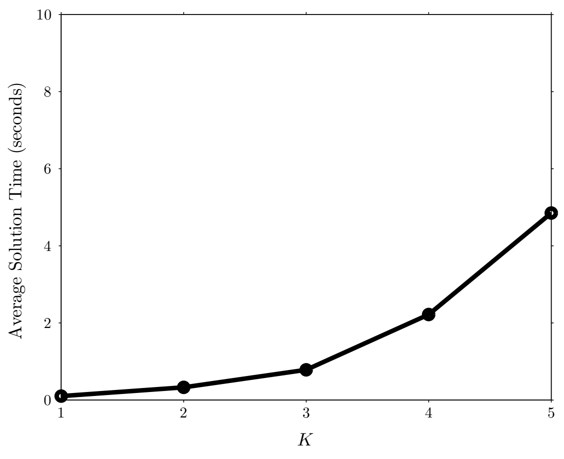


(a) Effect of $|S|$ (b) Effect of $|T|$

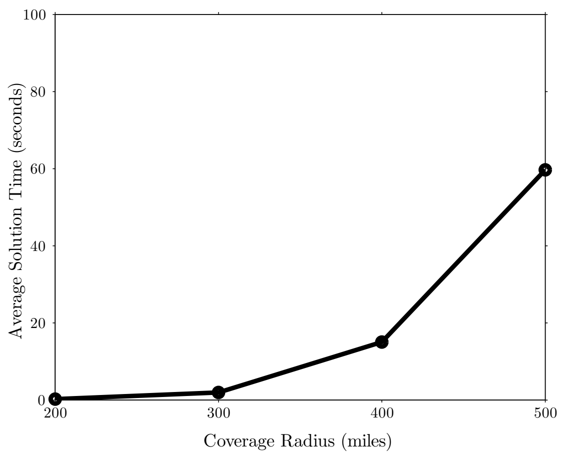
Figure 2.7: Effect of number of potential locations on solution times

One final factor is the number of leader facilities to open (i.e., budget). An increase in the budget does not increase the size of the instance but increases the number of possible facility combinations for the leader, increasing the feasible solution space. A comparison between T19 and T20 demonstrates this effect. When the budget is increased from 4 to 8, average solution time increases 3.1 times.

The same comparisons, when made for the %LPGap column, reveal different results. Increasing the number of follower facilities to open or the coverage radius increases the gap. However, when we increase the number of potential leader facilities and the investment budget, we see that the gap decreases. Comparing T1, T4, T15 and T18; and T2, T12, T16 and T19, we see that the LP gap decreases as $|S|$ increases. Similarly, comparing T4 and T10, and T19 and T20, we see that an



(a) Effect of K

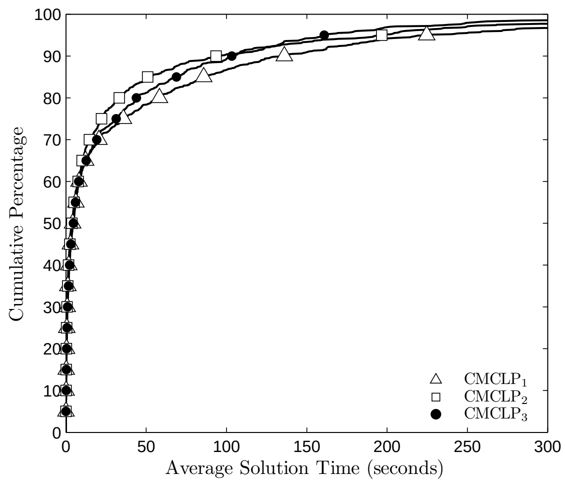


(b) Effect of coverage radius

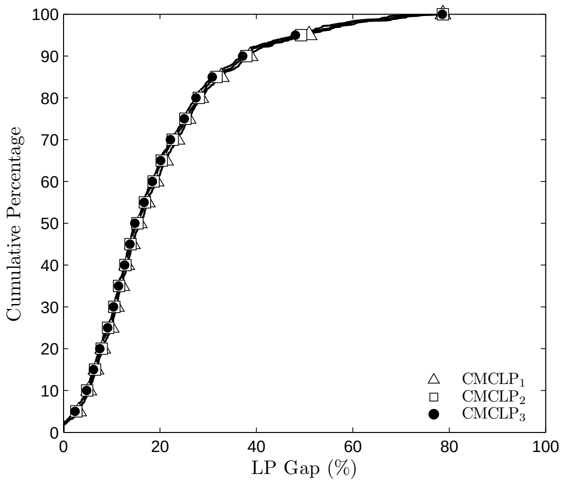
Figure 2.8: Effects of the number of greedy iterations and coverage radius on solution times

increase in budget decreases the gap. These are also expected as the former would make it possible to have more good locations and the latter would make it possible to select more facilities from the same set, keeping the opponent's response capacity the same. The results for the number of follower facilities have a opposite effect, as gap increases with increasing number of facilities.

The tables and the accompanying figures above provide a comparison among instance types, which represent different difficulty levels. The cumulative histograms below illustrate the solution performance of each formulation over all instances. Figure 2.9a shows that around 75% of the instances were solved in under 30 seconds. The performance of the three formulations are close but begin to differ as more difficult instances appear. Then, CMCLP2 and 3 dominate CMCLP1. Interestingly, CMCLP2 performs the best for instances that were solved in 2 minutes, and after that CMCLP3 takes over. Figure 2.9b indicates that about 65% of the instances had an LP gap of less than 20% and about 2% of the instances had no integrality gap. It also shows that the tightness of the formulations increases from CMCLP1 to 3, which is observable on Tables 2.3 and 2.4 as well.



(a) Solution times



(b) LP Gap

Figure 2.9: Comparison of three formulations

2.6.3 Solution Quality

Replacing the follower's optimization problem with the greedy add algorithm leaves us with potentially suboptimal solutions for the leader's problem. In general, the model is expected to perform well if the greedy add algorithm performs well for the restructured MCLP problem of the follower. By restructuring, we refer to the loss of coverage over previously covered customers as some customers would prefer open leader facilities. We would like to see how meaningful the solutions of the model are. To do so, we perform the following two procedures.

First, upon solving Problem 4, we fix the leader's open facilities. Then we restructure the follower's coverage relations and solve to optimality for the follower's problem alone. This yields the *best response* of the follower to the decision that the leader makes solving Problem 4. The best response is important because we expect that this would be the follower's true response, and the customers that the follower serves in the solution of this problem would be the ones that the follower actually would serve. Finally, we find the customers that are covered by the leader's open facilities but are not served by the follower in the subsequent solution and obtain the expected captured demand for the leader. We then compare this expected result with the suggested result of Problem 4 which yields the "error" in the leader's objective by assuming a greedy follower response. A zero error indicates that the suggested and expected objectives are the same.

Next, we solve the modified model and obtain an upper bound on the leader's optimal capture. Then we compare it again with the expected captured demand of the leader. Note that we solve Problem 4 to obtain the leader's strategy, but solve the modified model to obtain an upper bound only. This comparison yields the "optimality gap" of the leader. If this gap is zero, then the model has found the overall optimal solution. The error and the optimality gap is calculated using the formulas below, where superscript M denotes the modified model results.

$$\begin{aligned} \% \text{ Error} &= \frac{|LC(L^G, F^G(L^G)) - LC(L^G, F^*(L^G))|}{LC(L^G, F^G(L^G))} \\ \% \text{ Optimality Gap} &= \frac{LC(L^M, F^M(L^M)) - LC(L^G, F^*(L^G))}{LC(L^G, F^*(L^G))} \end{aligned}$$

In Tables 2.5–2.6 we summarize the error and optimality gap results for each of the 20 instance types. Table 2.5 is organized as follows. "Avg |Error| (%)" shows the average error (in absolute value) between the suggested and expected objectives and "Max |Error| (%)" shows the maximum of these errors, both as a percentage of the suggested objective. "% Correct" shows in what percent of the results the model objective was correct. The experiments show that for these instances the model performed quite well, the returned objective was at least 74 percent of the time correct and 93.4 percent on the average. The maximum percent error was 15.84 percent, whereas on average the percent error was 0.28 percent.

Table 2.6 is organized similarly. "Avg Opt Gap (%)" shows the average optimality gap and "Max |Error| (%)" shows the maximum of these gaps, both as a percentage of the suggested objective as a percentage of the suggested objective. The last column, "% Optimum," shows what percent of the instances were solved to optimality by the model. On the average, the optimality gap was 0.51 percent whereas the maximum optimality gap was 18.81 percent. Overall, 87.5 percent of all

Table 2.5: Solution quality: Leader's error in the objective

Type	Avg |Error| (%)	Max |Error| (%)	% Correct
T1	0.19	9.38	98
T2	0.55	14.0	92
T3	0.00	0.0	100
T4	0.01	0.30	98
T5	0.15	5.19	94
T6	0.44	6.53	88
T7	0.00	0.0	100
T8	0.00	0.0	100
T9	0.01	0.28	98
T10	0.13	4.19	94
T11	0.23	6.11	92
T12	0.10	5.24	98
T13	1.56	13.80	74
T14	0.76	15.84	92
T15	0.13	6.37	98
T16	0.22	7.43	94
T17	0.25	8.36	94
T18	0.21	3.83	88
T19	0.46	7.33	86
T20	0.28	5.88	90
Summary	0.28	15.84	93.4

instances were solved to optimality.

In light of Table 2.2, we see that the performance deteriorates with increasing numbers of leader and follower facilities, coverage radius, and number of greedy iterations. The performance, on the other hand, is enhanced by an increasing investment budget of the leader. Note that when $K = 1$, the error as well as the optimality gap is always 0 as the follower responds optimally. Figure 2.10 shows a joint histogram of error and optimality gap. This result does not differ between alternative formulations. All models give the same result since the generated instances have all transitive consumer choices (shortest distance).

Table 2.6: Solution quality: Leader's optimality gap

Type	Avg Opt Gap (%)	Max Opt Gap (%)	% Optimum
T1	0.28	10.34	94
T2	1.04	18.26	86
T3	0.00	0.0	100
T4	0.02	0.68	96
T5	0.31	5.47	88
T6	0.65	8.16	80
T7	0.00	0.0	100
T8	0.00	0.0	100
T9	0.04	1.77	96
T10	0.13	4.37	94
T11	0.28	6.50	88
T12	0.58	10.10	88
T13	2.61	17.70	58
T14	1.28	18.81	82
T15	0.28	13.32	96
T16	0.48	8.2	84
T17	0.55	12.37	90
T18	0.21	2.41	80
T19	0.98	10.50	74
T20	0.56	6.45	76
Summary	0.51	18.81	87.5

2.7 Conclusions

In this chapter, we introduced a mathematical model in order to devise a strategy for the leader firm in a leader–follower version of the maximal covering problem. Our primary contribution is to embed the follower's response into the constraints of the leader's optimization problem so that the leader's problem is reduced from a bilevel program to a single-level one. This keeps the leader's problem *tractable* and the problem size at a *manageable* level. Our model of the follower employs the greedy add algorithm rather than solving his problem exactly, but we demonstrate that this assumption comes at the expense of very little loss of accuracy.

Our computational study demonstrates that the model is able to:

- Generate near optimal (often optimal) solutions for the leader

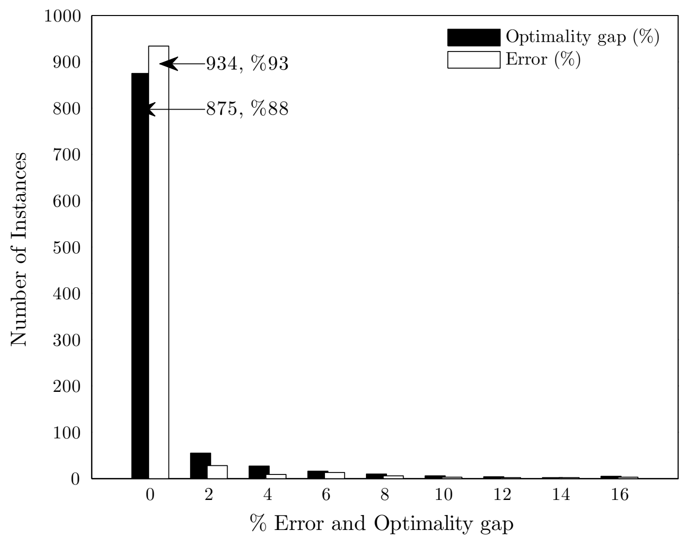


Figure 2.10: Histogram of objective error and optimality gap

- Provide a reasonable assessment of the follower's response that can be embedded into the leader's optimization model.

Our numerical studies also indicated that the model size and the solution effort increase quickly with increasing problem components—potential sites, customers and number facilities to open. One interesting avenue for future research would be the introduction of demand uncertainty, but the resulting scenarios would further increase the computational burden. Therefore investigation of solution methods that do not require MIP solvers such as Lagrangian relaxation and decomposition (e.g., Benders) based approaches is in our future research agenda.

Finally, the problem was stated with the most common features for competitive location problems. Variants of the problem with different patronizing rules and competition over potential locations (i.e. from a single set of potential locations) as well as customers would be important to study. It would also be worthwhile to study the applicability of other simple algorithms as proxies for the follower's response.

Chapter 3

A Competitive Network Design Problem

3.1 Introduction

Networks exist everywhere. Power networks transmit the electricity for our lighting and home appliances. Telecommunication networks enable us to communicate regardless of the distances. Production and distribution networks make the goods we need available. The efficiency of these networks is crucial for us; therefore, the design of these networks has been a fundamental area in operations research. Earlier studies about network design have a centralized point of view which aims to find the optimal design from a single designer's perspective.

However, networks are often formed without a central decision maker. The decentralized decision arising from competition among decision makers leads to network structures that have different structures and deviate in reliability, cost, etc. compared to the central (i.e., optimal) solutions. Studying competitively formed network structures has two objectives. A stakeholder (such as a local government) can devise incentives to let decision makers voluntarily align their decisions with her desired design. A competitor can develop design strategies that take rivals' responses into account.

In this study, we introduce the network design game played on a graph $G(N, E)$ with nodes N and edges E, respectively, by independent decision makers —players— $p \in P$. Players are located at some nodes of this graph. Each node has a revenue potential and players connect these nodes to their origins by building arcs, which is costly. Each player, thus, wants to maximize her profit, which is the net of earned revenue minus investment costs. Each individual player's problem is a special type of the *Prize-Collecting Steiner Tree* problem (PCST).

First, we consider a simultaneous-move setting and characterize the equilibrium actions of the players. We propose a mixed integer programming (MIP) formulation that finds a pure strategy Nash-equilibrium. Then, we discuss a Stackelberg version among two players. This problem is a bilevel integer programming problem and we propose a heuristic reformulation of the lower level (i.e., follower, second mover) problem that allows us to solve the problem as a single-level MIP providing near optimal design strategies for the leader in the game. Furthermore, this reformulation strategy is immediately applicable to other bilevel problems, whose second-level problems can be posed as a prize-collecting Steiner tree problem.

We start with a brief review of key studies and concepts regarding competitive formation of networks in Section 3.2.1. Then, we introduce the problem in Section 3.2.2, along with a review of studies. In Section 3.4 we introduce the mathematical models for the noncompetitive case (i.e., single-player, central decision maker). Then, in Section 3.4 we introduce the two models we developed for the competitive case, respectively, for a simultaneous-move game and a sequential game. Section 3.6 provides our computational study on the models and Section 3.7 concludes.

3.2 Literature Review

3.2.1 Competitive Network Formation

Network formation has been comprehensively studied from the game theory perspective. The motivation came mostly from the study of social networks and the formation of economic relationships between firms/countries. Some of the earliest works that define a simple model for competitive network formation are Aumann and Myerson (1988) and Myerson (1977) (also see (Myerson, 1991)). Authors define an extensive form and a simultaneous move form for the network creation game, where the nodes are the players and they wish to build arcs to other players. The idea being the same, the assumed arc type, payment structure and objective functions of the players lead to a wide range of games. For instance Fabrikant et al. (2003) and Albers et al. (2006) study the *network connection game* in which each node is a player and each player picks a subset of the other players to build arcs to. The arcs are undirected, they are paid for by the builder and once built they can be used by everyone. Each player minimizes the cost of building arcs and the total distance to all other players. This approach is quite meaningful in explaining the spread of networks like the internet where cost and proximity is of value. Halevi and Mansour (2007) study the same network but only consider the building cost and proximity to a subset of players which are named as friends, hence bring the emphasis on a neighborhood of players for each player. Piliouras and Vigfusson (2009) extend the idea where there is no connectivity requirement but there is an associated penalty for unreachable nodes. Some interesting examples of similar games like free-trade treaties, market sharing and academic cooperation (co-authoring) are introduced by Jackson (2003), who provides a comprehensive survey of network creation models.

Connection games are the most basic of these games. A more sophisticated game is the *path player game* (Puerto et al., 2008). In this game setting, the paths in a given network are players and they wish to maximize the flow through them. Although this is not a network creation game, it is important as it replaces the node player with a path player, which might be more appropriate to represent (with a complexity trade-off) the service providers such as railroads. A direct generalization of this is a sub-network player of a given network wishing to maximize the flow through it. 3PL's and passenger transportation firms are examples. The non-cooperative *tree creation game* (Hoefer, 2006) is an extension of the connection game where players are nodes and they wish to connect themselves to their terminal(s) forming a tree. Finally, *spanning tree games* (Gourvés and Monnot, 2008) are another type of sophisticated network formation games where a set of nodes are given, of which one is designated as a root node. The remaining nodes are players and they decide on their parent node (thus create a link to it) to ultimately connect themselves to the root node. Variations consider reliability (max), cost (min) and flow (bottleneck) objectives.

The model we suggest resembles spanning tree games the most. However, in our game, the root nodes are the players (so there are multiple root nodes) and the remaining nodes, even in the final network, are not necessarily spanned due to the investment budget. Even if the investment budget is infinite, the resulting network need not be a spanning tree. We explain our game in the next section.

In network formation games, two values are of significant importance. The first one is the *price of anarchy*, introduced by Papadimitriou (2001), which is the ratio of the optimal (centralized) network and the worst case Nash-equilibrium. This ratio, assuming that the game will end at some Nash-equilibrium, gives the worst case deviation from the optimal result, and thus points out the maximal cost of strategic behavior. On the other hand, as multiple Nash-equilibria are quite common, the *price of stability* (Anshelevich et al., 2004) refers to the ratio where the most likely — presumably the most profitable — Nash-equilibrium is considered instead of the worst case, reflecting the minimal cost of strategic behavior. Perakis and Roels (2007) studies the price of anarchy in supply chains under different given network topologies and contracting schemes. Since the networks are given, there exists no network creation addressed in it.

These values provide metrics to evaluate the quality of the networks formed. The study of improving the network quality (profitability, reliability) is a recent step taken in studying network formation games. Creating better networks and mechanisms to induce such network formation will also be an important part of our study. For instance, for certain games the resulting network need not be connected. Brandes et al. (2008) study games where disconnected network formation is not prohibited but penalized. Anshelevich et al. (2003) discuss near optimal network design maintaining competition. Chen et al. (2008) and Christodoulou et al. (2009) discuss coordination mechanisms and cost sharing in cooperative games, respectively.

3.2.2 Prize-collecting Steiner Tree

The Steiner tree problem is one of the most well-known problems in combinatorial optimization. Given a set of nodes, a Steiner tree is the graph that connects these vertices at minimum cost. It is also defined on an undirected graph $G(N, E)$ with associated weights for the edges E. A minimum weight connected subgraph that includes a given subset S of nodes N is sought. This is an NP-Hard problem; in fact, its decision problem is one of the original 21 NP-Complete problems of Karp (1972). For a general discussion of this problem, readers may refer to the reviews by Cieslik (1998) and Hwang et al. (1992).

The problem we consider is a variant of the Steiner tree problem. In the "prize-collecting" case, revenues are associated with the nodes of the graph and the node inclusion requirement is dropped. The problem is introduced by Segev (1987) with the name *node-weighted Steiner tree problem*. Segev studied a variant where the node inclusion requirement is dropped for all but one node (origin, root) and provided MIP formulations. This is the problem variant we are studying and we will refer to these formulations in the subsequent discussion. We are going to refer to the rooted variant as PCST, as well.

Bienstock et al. (1993) introduced the name PCST and a 3-approximation (guaranteeing an objective within 3 times the optimum) for it. Goemans and Williamson (1995) pose the problem as an equivalent minimization problem, where the sum of the costs of built edges and the revenue of unselected nodes is minimized. They propose a $(2 - \frac{1}{|N|-1})$-approximation (for the rooted variant we study) that runs in $O(|N|^2 \log |N|)$ time. Johnson et al. (2000) proposed an alteration over this — a subroutine called *strong pruning* — which we will also use in our formulations. An algorithm with a better approximation ratio $(2 - \frac{2}{|N|})$ and running time $(O(|N|^2 \log |N|))$ for the original prize-collecting problem is proposed by Feofiloff et al. (2007). For polyhedral studies regarding these problems readers may refer to Fischetti (1991) and Goemans (1994).

Some solution approaches other than Segev's work are as follows. Engevall et al. (1998) propose an integer programming formulation and a Lagrangian relaxation algorithm. Lucena and Resende (2004) propose a cutting-plane algorithm. Ljubić et al. (2006) implemented a branch-and-cut algorithm bringing together several earlier methods. Recently, Haouari et al. (2008) report on Lagrangian relaxation using several subgradient strategies.

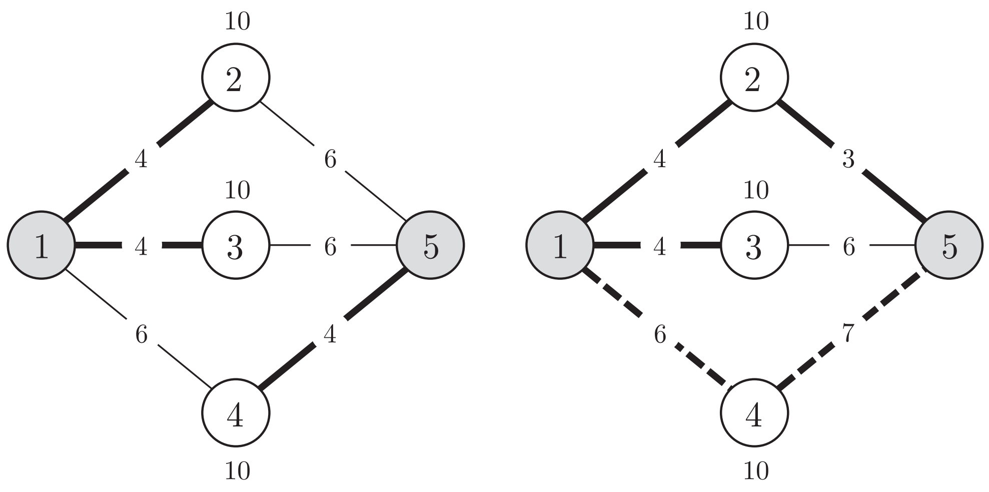
Finally, the problem has some immediate extensions such as imposing a quota (a lower limit on collected revenue) or a budget (an upper bound on incurred costs). We do not consider them in this study. Hereafter, we refer to the rooted variant of the prize-collecting Steiner tree problem that we study as PCST. In the next section, we are going to introduce how to incorporate competition into the PCST.

3.3 The Problem

Let $G(N, E)$ be an undirected graph with associated potential revenues r_i for each node $i \in N$ and costs $c_{i,j}$ for each edge $(i, j) \in E$. There are P players. Each player p is initially located at her individual origin node $o_p \in O \subseteq N$ and selects a subset of edges forming a connected subgraph that includes the initial location of the player. These parameters are the same for each player. The strategy set of a player is denoted as $S_p = \{G_p(N_p, E_p) : G_p \text{ is connected}, E_p \subseteq E, N_p \subseteq N, o_p \in N_p\}$. A strategy can also be equivalently formulated as the set of edges E_p selected by player p as edges would imply inclusion of the end nodes.

We consider a *non-cooperative game* setting in which each player is going to pay the full cost of the arc that it builds, yet the potential revenue is shared among players who build an arc to that node. For simplicity, we consider equal sharing of potential revenue, but other sharing rules can be incorporated into the models we are going to introduce. The players maximize their profits, which is the net of total earned revenue minus the construction cost of the individual network. Consequently, a Nash-equilibrium is defined as a result of the above game from which no individual player is willing to unilaterally deviate, hence find a better feasible solution (with higher profit) by only altering her own decision.

Before proceeding with the model, we briefly discuss the equilibria, central solution and preferable network structures. Consider Figure 3.1 which illustrates a network with 5 nodes and 6 arcs and associated potential node revenues and edge costs. Players 1 and 2 are respectively located at nodes 1 and 5. In Figure 3.1a, the centralized solution (shown with bold edges) consists of the edges {(1,2), (1,3), (5,4)} and has a profit of 18. In this case, when the two (non-cooperative) players make their decisions simultaneously, the equilibrium would be identical to the centralized solution. This is an ideal situation where competition does not cause inefficiency.



(a) Equilibrium, optimal

(b) Equilibrium, suboptimal

Figure 3.1: An example game

However, this solution is an unconnected network. Connectivity could be an important criterion for the reliability of the network. For instance, if this was a power network, connectivity could serve as a protection against generation failure at nodes 1 or 5. For a third party (e.g., government regulatory authority) that is interested in the connectivity of the network, this property has value. Therefore, this investigation can yield information for third parties in shaping their incentive policies. For instance, introducing a 2 unit penalty for unconnected networks the competitive solution moves towards connected networks introducing either of {(5,2), (5,3), (1,4)} to the equilibrium/optimal solution.

The identical result above cannot be obtained for all revenue/cost settings. Figure 3.1b illustrates a different example. The optimal network is {(1,2), (5,2), (5,4)}, yielding a profit of 17. However, when competition is brought into play, the equilibrium decisions are {(1,2), (1,3), (5,2)} (bold edges) and one of {(1,4), (5,4)} (dashed edges). This not only illustrates a deviation from central design, but also multiple equilibria. As a result, efficiency (i.e. profit) loss from competition does not only exist but also change depending on the obtained equilibrium. Here, the equilibria including (5,4) and (1,4) exemplify the *worst* and *best* equilibria, respectively, thus indicate *price of anarchy* of $\frac{17}{12} = 1.42$ and *price of stability* of $\frac{17}{13} = 1.31$. On the other hand the resulting network is connected compared to the unconnected optimal network, which does not require any incentives, unlike the other. As a final remark, it is possible that the firms cooperate in the design process. For instance, a contract, in which player 2 pays player 1 a sum of 2 units in exchange for not building arc (1,2), will result in an improved competitive solution. In the models below, however, we do not consider cooperation between players.

3.4 MIP Models for the Noncompetitive Problem

3.4.1 Individual Player

When we disregard competition, the problem of a given player is a Prize-Collecting Steiner Tree (PCST) problem. There are a few variants of this problem. Here, we consider the profit maximization variant. Given an undirected graph $G(N, E)$ with associated a nonnegative cost for each edge and a nonnegative revenue for each node, a player p selects (builds) a connected subgraph $G(N, E)$ that maximizes the net of collected revenues minus the incurred costs. The only restriction other than the connectivity requirement is that the player's origin node o_p should be in the built subgraph. For the sake of formulation, we replace the undirected graph with the equivalent directed graph $G(N, A)$ that has arcs $(i, j) \in A$ and $(j, i) \in A$ with symmetric cost parameters. We do not consider origin nodes as occupiable, hence let $r_i = 0$ for $i \in O$ and let $N' = N - O$ be the nodes that are occupiable, $A' = (i, j) \in A$ such that $i \in N' \cup o_p$ and $j \in N'$ be the arcs that are selectable.

Indices:

$i, j, k \in N$: Vertices, nodes

$p \in P$: Players

$o_p \in O \subseteq N$: Origin node of player p

Parameters:

c_{ij} : Cost of building arc (i, j), $c_{ij} = c_{ji} > 0$

r_j : Potential revenue of node j, $r_j \geq 0$

M : A sufficiently large positive number (better, arc specific M_{ij})

Variables:

y_{ij} : If arc (i,j) is selected $y_{ij} = 1$ and 0 otherwise

f_{ij} : Flow through arc (i,j)

The optimal solution to this problem is a tree; more specifically, given the selected subset of nodes, it is a minimum cost spanning tree (MCST) that spans only the selected nodes and is rooted at the origin node of the player. Ensuring this, hence, avoiding cycles, is key in the formulation. We can introduce subtour elimination constraints for that purpose. Instead, in Model 1, we prefer to use a *network flow* formulation and eliminate cycles through flow balance equations. It is a modification of Segev's (1987) tree-type formulation. Basically, we assume that the flow only passes through selected arcs and is reduced at the visited nodes by an amount that is equal to the revenue earned at that node. The flow originates from the source node and is consumed at the selected nodes. This resembles the approach of Miller, Tucker, and Zemlin (1960) for the traveling salesman problem.

Model 1 PCST Problem: Tree formulation

$$
\begin{aligned}
\max \quad & \sum_{j \in N'} f_{o_p,j} - \sum_{(i,j)\in A'} c_{ij} y_{ij} && && (3.1)\\
\text{s.t.} \quad & \sum_{j \in N' \cup o_p} f_{ji} - \sum_{j \in N'} f_{ij} = r_i \sum_{j \in N'} y_{ji} && \forall i \in N' && (3.2)\\
& x_{ij} \leq M y_{ij} && \forall (i,j) \in A' && (3.3)\\
& \sum_{i \in N} y_{ij} \leq 1 && \forall j \in N' && (3.4)\\
& f_{ij} \geq 0 && \forall (i,j) \in A' &&\\
& y_{ij} \in \{0,1\} && \forall (i,j) \in A' &&
\end{aligned}
$$

The objective (3.1) states that maximizing profit is equivalent to maximizing the net of total flow originating from the source node minus the total fixed arc costs. Flow balance constraint (3.2) requires that flow into a node is equal to the revenue of the node plus the flow out from the node.

Constraint (3.3) states that a positive flow through an arc, implies a built arc. Constraint (3.4) ensures that only one arc can enter a selected node. The resulting network is a tree, as directed cycles are eliminated by (3.4) and undirected cycles by (3.2). Finally, the flow variables f_{ij} are nonnegative and the arc selection variables y_{ij} are binary.

Model 1 is not very strong because of the M in (3.3). Setting $M = \sum_{j \in N'} r_i$ is sufficient, but an immediate improvement to this is to assign separate M_{ij} for each arc. We can calculate them as follows. Let SP_j denote the shortest path from o_p to node $j \in N'$ on graph $G(N, A)$ such that the distance d_{ij} between two nodes is defined as the revenue of the destination node. That is, $d_{ij} = r_j$. Then, we set $M_{ij} = \sum_{j \in N'} -SP_i$.

This modification helps, but we can come up with a tighter formulation, eliminating M completely. As mentioned earlier, whenever we select a set of nodes, the arcs to be selected is just a MCST problem. Therefore, we can also model PCST as a node selection problem, where the subsequent arc selection problem is modeled and solved as a linear program, for instance, using a multi-commodity flow formulation. We can define a separate flow from the origin to each node with positive revenue as if they were different commodities with unit demand at the destination nodes. This would result in Model 2, which is the multi-commodity flow formulation in Segev (1987). We are only interested in destinations with positive revenue, hence let N^+ be the set of $i \in N'$ such that $r_i > 0$.

Variables:

x_j : If node j is selected $x_j = 1$ and 0 otherwise

f_{ijk} : Flow through arc (i, j) with destination k

The objective (3.5) is the total revenue from selected nodes with positive revenue minus the total cost of selected arcs. Flow balance equations (3.6) ensure that for a selected (i.e. $x_k = 1$) node a unit flow is supplied by the origin, carried through intermediary nodes untouched and is consumed at the destination node. Similarly, according to (3.7), positive flow implies a selected arc. Since the flows are at most 1, we can set $M = 1$ and eliminate it from the formulation. Finally, flows f_{ijk} are nonnegative, and selection variables x_j and y_{ij} are binary.

The best thing about this formulation is that it is very tight, often returning an integral solution at the root node. The downside is, of course, the size of the instances it creates as it has about $|N^+|$ times more variables and constraints than the previous model, which exhausts memory rapidly. Despite this fact, we are going to use this formulation as the base for the coming models and related computational experiments.

Model 2 PCST Problem: Multi-commodity formulation

$$\max \quad \sum_{k \in N^+} r_k x_k - \sum_{(i,j) \in A'} c_{ij} y_{ij} \tag{3.5}$$

$$\text{s.t.} \quad \sum_{j \in N' \cup o_p} f_{jik} - \sum_{j \in N'} f_{ijk} = \begin{cases} -x_k & \text{if } i = o_p \\ x_k & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in N' \cup o_p,\ k \in N^+ \tag{3.6}$$

$$f_{ijk} \leq y_{ij} \qquad \forall (i,j) \in A',\ k \in N^+ \tag{3.7}$$

$$f_{ijk} \geq 0 \qquad \forall (i,j) \in A',\ k \in N^+$$

$$x_k \in \{0,1\} \qquad \forall k \in N^+$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A'$$

3.4.2 Central Decision Maker

We can easily extend this model to include multiple origins (players) when there is a central decision maker. Note that, with the inclusion of multiple origins the optimal subgraph becomes a forest with trees rooted at the origins. We can overcome this by picking one of the origins, say o_p, and defining dummy arcs, $(o_p, o_{p'})$, with 0 cost from this origin to all other origins. This turns the problem into an instance of PCST with origin node o_p. We simply update update $A' = A' \cup \bigcup_{p' \in P}(o_p, o_{p'})$ in the constraints. Note that this case is essentially the same as a single player with multiple origin nodes (e.g., a company with multiple power plants); hence, the trick would work for that case, too.

3.5 MIP Models for the Competitive Problem

As mentioned earlier, we are interested in the equilibrium solutions that are obtained under competition. We consider two game settings — *simultaneous* and *sequential*. The equilibria for simultaneous game can be found by mimicking the unilateral moves of each player on the $|P|$ dimensional matrix of player strategies until no utility-increasing (i.e., profit-increasing) move can be made by the players. Each move is an instance of PCST, once the revenues are updated according to the last occupancy information. This method, therefore, requires solving an indeterminate number of PCST instances. Whether there exists such an equilibrium is also a question. Using our second model, we show the existence of a pure-strategy equilibrium and as we propose an MIP model that solves for an equilibrium solution, returning a pure-strategy equilibrium for the players.

3.5.1 Simultaneous Game: Iterative Method

A simple way to find an equilibrium is to mimic the unilateral move of a player on the $|P|$ dimensional strategy matrix, given the rest of the players' strategies. Algorithm 3.1 performs this. It solves for a single player's decision optimally at each (inner) iteration, and updates the revenue structure accordingly. If two consecutive passes through all players (two consecutive outer iterations) generate the same network structure, then the algorithm terminates, returning an equilibrium solution. Let $G_{p,t}(N_{p,t}, A_{p,t}) = \{N_{p,t}; A_{p,t}\}$ be the graph built by player p at (outer) iteration t, $n_{i,p}$ be the number of occupants other than p in node i at a given time, and r_i^0 and $c_{i,j}^0$ be the initial values of the parameters for node revenue and arc cost.

3.5.2 Simultaneous Game: MIP Model

There are $|P|$ players in the game that is played on the undirected graph $G(N, E)$ (and the corresponding directed graph $G(V, A)$). There are $|A| = 2|E|$ items to select from for each player. A decision made by player p on arc (i, j) is denoted with a decision variable y_{ijp} where $y = 1$ if the arc is selected and 0 otherwise. Then the decision vector $y_p = (y_{ijp} : (i, j) \in A)$ is called a strategy

Algorithm 3.1 Iterative Method to Find Equilibria

```
let $t = 0, r_i^0 = r_i, c_{i,j}^0 = c_{i,j}$
let $G_{p,t}(N_{p,t}, A_{p,t}) = (o_p, \emptyset)$ for all $p \in P$
repeat
   $t \leftarrow t + 1$
   for $p = 1$ to $|P|$ do
      compute $n_{i,p}$ for all $i \in N'$
      $r_i \leftarrow \frac{r_i^0}{n_{i,p}+1}$ for all $i \in N'$
      solve $PCST(p)$ with updated parameters.
      $G_{p,t}(N_{p,t}, A_{p,t}) \leftarrow PCST^*(p)$
   end for
until $G_{p,t}(N_{p,t}, A_{p,t}) = G_{p,t-1}(N_{p,t-1}, A_{p,t-1})$ for all $p \in P$
return $G_{p,t}(N_{p,t}, A_{p,t})$ for each $p \in P$ as equilibrium solution
```

for player $p \in P$. The number of strategies of a player is finite (in fact, $2^{|A|}$) and this strategy set is the same for all players. However, the feasible strategy set is less as the arcs need to form a tree and they are different for different players as the selection of the player's origin node is mandatory in building a feasible player network.

This model is based on the idea by Rosenthal (1973) who described a class of games which possess pure strategy equilibria. Before introducing the mathematical model and going on with the formal proof that this model finds an equilibrium solution, let us discuss the rationale and give some examples. The model is quite similar to the earlier models in the definition of the feasible network structure (i.e., tree). Since the optimal network built by each player is a tree, we are going to eliminate non-tree solutions from players' strategy sets. Each player selects a subset of arcs (similar to *primary factors* in Rosenthal (1973)) and the head nodes pointed by these arcs. The revenue that a player would expect to earn when she selects an arc (i, j) when node j is already occupied by $n - 1$ players is $\frac{r_j}{n}$ and and she pays c_{ij}, yielding a marginal profit of $\frac{r_j}{n} - c_{ij}$. Recall that arc costs are fully paid by the builder while the node revenues are shared equally. Remembering the iterative procedure, this resembles how the decision is made by a player when her turn comes. The player would build the most profitable tree given the number of occupants in, hence the updated revenue of, each node. Roughly, our claim is that if we maximize the sum of this marginal profit — revenues over all nodes and number of players occupying them and costs over all selected arcs — and only allow trees to be built by individual players, this would result in an equilibrium solution.

Let us revisit the example from Section 3.4. Considering Figure 3.1b would give us an idea of how this method works. Two players — left and right — make their decisions on feasible strategy sets {(1,2),(1,3),(1,4)} and {(5,2),(5,3),(5,4)}, respectively. Both of them select node 2 because given whether the other selects, selecting node 2 still increases the sum — by $5-4=1$ and $5-3=2$, respectively — for both. If both players select node 3, then the sum would be $10+5-4-6=5$. If only left selects, $10-4=6$ or only right selects, $10-6=4$. Thus the sum of marginal profits would be maximized if left player selects. We already know from our previous analysis of these examples that these were the selected arcs in an equilibrium. However, we could select either of the dashed arcs and each would be an equilibrium solution. This method would return an equilibrium that selects $(1,4)$ because it has a higher marginal profit — $10-6=4$ vs. $10-7=3$ — and sharing would decrease — $10+5-6-7=2$ — the sum. As a result, the method promotes sharing when profitable by both (or more) parties, and prefers lower cost (higher margin) alternatives otherwise.

We introduce the new additions and modifications to the notation below. Note the slight difference in the definitions of parameter r and variables x, y and f. Since the model includes all players, we define $A' = (i,j)$ such that $i \in N$ and $j \in N'$.

Indices:

$n \in P$: Number of occupants at a selected node (in the solution).

Parameters:

r_{jn} : Revenue earned from node j by a player given that there are $\beta \geq 1$ occupants in the node. $r_{jn} = \frac{r_j}{n}$

Variables:

x_{kp} : If player p selects node k, $x_{kp} = 1$, and 0 otherwise

y_{ijp} : If player p selects arc (i,j), $y_{ijp} = 1$, and 0 otherwise

f_{ijpk} : Flow originating at o_p, passing through arc (i,j) with destination k

t_{jn} : Variable equals to 1 if n players occupy node j.

Model 3 Competitive PCST: Simultaneous Move Game

$$\max \quad \sum_{k \in N^+} \sum_{n \in P} r_{kn} t_{kn} - \sum_{(i,j) \in A'} \sum_{p \in P} c_{ij} y_{ijp} \tag{3.8}$$

$$\text{s.t.} \quad \sum_{j \in N} f_{jipk} - \sum_{j \in N'} f_{ijpk} = \begin{cases} -x_{kp} & \text{if } i = o_p \\ x_{kp} & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \qquad i \in N,\, p \in P,\, k \in N^+ \tag{3.9}$$

$$f_{ijpk} \le y_{ijp} \qquad (i,j) \in A',\, k \in N^+ \tag{3.10}$$

$$\sum_{p \in P} x_{kp} = \sum_{n \in P} t_{kn} \qquad k \in N^+ \tag{3.11}$$

$$f_{ijpk} \ge 0 \qquad (i,j) \in A',\, k \in N^+$$

$$y_{ijp} \in \{0,1\} \qquad (i,j) \in A'$$

$$x_{kp} \in \{0,1\} \qquad k \in N^+,\, p \in P$$

$$t_{kn} \in \{0,1\} \qquad k \in N^+,\, n \in P$$

Objective (3.8) is the sum of all nodes' marginal revenues over each possible number of occupants (if none occupies the node, this is 0) minus the total cost of arcs built by all players. (3.9) is the flow balance constraint at node i, for each destination node (i.e. commodity) k. If node i is an origin node of player p it supplies a flow of x_{kp}. If $i = k$, it consumes the flow x_{kp} supplied by player p. Otherwise, inflow equals outflow, netting 0. A positive flow through (i, j) with origin o_p and destination k indicates that arc (i, j) is selected by player p. Constraint (3.11) ensures that the number of occupants in nodes (with positive revenue) is accounted for correctly. The right hand side is the number of occupants. Assume there are m occupants in the optimal solution. Since r_{kn} is decreasing in n and the problem is a maximization problem, we obtain $t_{k,1} = \ldots = t_{k,m} = 1$ and $t_{k,m+1} = \ldots = t_{k,|P|} = 0$ at optimality. Finally, the flow variables are continuous and the arc/node selection variables are binary.

The above problem is feasible (each player can simply select her source vertex as a feasible strategy) and the number of feasible strategies is finite with finite profits, so the solution exists and is bounded. The second question is whether this solution is an equilibrium or not. In order to qualify as an equilibrium, a solution should consist of *best responses* of the players given the others' responses. Thus for a player p, denoting her decision vector as y_p and the corresponding individual profit given other players' decisions as $\Pi_p(y_1, ..., y_p, ..., y_{|P|})$, a solution $(y_1^*, ..., y_p^*, ..., y_{|P|}^*)$ is an equilibrium solution if and only if $\Pi_p(y_1^*, , y_p^*, , y_{|P|}^*) \geq \Pi_p(y_1^*, , y_p, , y_{|P|}^*)$ for all y_p, for all $p \in \{1, ..., |P|\}$.

Theorem 11. *The optimal solution of Model 3 is a Nash equilibrium for the simultaneous-move game.*

Proof. By contradiction. Let's assume that the solution is not an equilibrium solution, though it is still optimal to the model. Then there should exist a player who would *willingly* and *unilaterally* deviate from her current decision. The deviation is willing if the player would find a more profitable alternative, and it is unilateral if that player would alter only her decision, while other players' decisions are kept the same.

Thus, there exists a player $\hat{p}$ who drops some of her previously selected arcs and adds some of her previously unselected arcs. Let (*) denote the optimal model solution and (**) denote the solution obtained after $\hat{p}$ makes her decision. Let the set of dropped arcs be $A_0 = \{(i,j) : y_{ij,\hat{p}}^* = 1,\ y_{ij,\hat{p}}^{**} = 0\}$ and the set of added arcs be $A_1 = \{(i,j) : y_{ij,\hat{p}}^* = 0,\ y_{ij,\hat{p}}^{**} = 1\}$. Consequently, let the set of dropped head nodes be $N_0 = \{j : (i,j) \in A_0\}$ and the set of added head nodes be $N_1 = \{j : (i,j) \in A_1\}$.

Assume that in the solution, n_j players occupy the node j, thus $t_{j,1}^* =, \ldots, t_{j,n_j}^* = 1$. If a player drops arc (i,j), then $t_{j,n_j}^{**} = y_{ij,\hat{p}}^{**} = 0$ so her profit decreases by $r_{j,n_j} - c_{ij}$. On the contrary, if she adds arc (i,j), then $t_{j,n_j+1}^{**} = y_{ij,\hat{p}}^{**} = 1$ and her profit increases by $r_{j,n_j+1} - c_{ij}$. Since the alteration is profitable,

$$\sum_{(i,j)\in A_1}(r_{j,n_j+1} - c_{ij}) - \sum_{(i,j)\in A_0}(r_{j,n_j} - c_{ij}) = \sum_{j\in N_1} r_{j,n_j+1} - \sum_{(i,j)\in A_1} c_{ij} - \sum_{j\in N_0} r_{j,n_j} + \sum_{(i,j)\in A_0} c_{ij} > 0.$$

These steps are summarized below.

$$\begin{aligned}
z^* &= \sum_{j\in N'}\sum_{n\in P} r_{jn}t^*_{jn} - \sum_{(i,j)\in A'}\sum_{p\in P} c_{ij}y^*_{ijp} \\
&= \sum_{j\in N'}\sum_{n=1}^{n_j} r_{jn}t^*_{jn} - \sum_{(i,j)\in A'}\sum_{p\in P} c_{ij}y^*_{ijp} \\
&< \sum_{j\in N'}\sum_{n=1}^{n_j} r_{jn}t^*_{jn} - \sum_{(i,j)\in A'}\sum_{p\in P} c_{ij}y^*_{ijp} + \left[\sum_{j\in N_1} r_{j,n_j+1} - \sum_{(i,j)\in A_1} c_{ij}\right] - \left[\sum_{j\in N_0} r_{j,n_j} - \sum_{(i,j)\in A_0} c_{ij}\right] \\
&= \sum_{j\in N'}\sum_{n\in P} r_{jn}t^{**}_{jn} - \sum_{(i,j)\in A'}\sum_{p\in P} c_{ij}y^{**}_{ijp} \\
&= z^{**}
\end{aligned} \tag{3.12}$$

This contradicts our initial assumption that the solution was optimal (i.e., $z^* \geq z^{**}$), yet not an equilibrium. Thus, every solution to this problem is an equilibrium as well. □

Note that since the problem is always feasible and bounded and the solution is an equilibrium solution, a pure strategy equilibrium for the simultaneous-move game exists.

Corollary 12. *There exists a pure-strategy equilibrium for this game.*

This model does not necessarily give the most or the least profitable equilibrium solution. We can show this using the simple counterexample in Figures 3.2, which illustrates the graph and its parameters, and 3.3, which illustrates the four possible equilibria on this graph. The equilibria in Figures 3.4a–3.3d have objectives of 9, 8, 9, 10 but total profits of 4, 8, 9, 5, respectively. Therefore, Figures 3.3c and 3.4a show the most and least profitable equilibria, while the model returns the equilibria in Figure 3.3d.

Model 3 can be made smaller in size by a few modifications. We can redefine the variables f and y and obtain the compact formulation as follows:

Variables:

f_{ijk} : Flow passing through arc (i, j) with destination k

y_{ij} : Number of players that selected arc (i, j)

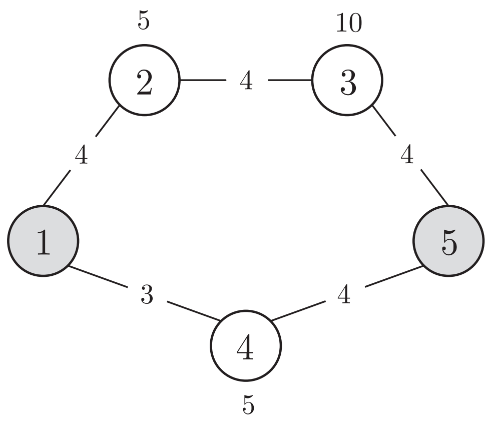


Figure 3.2: A counterexample: Setting

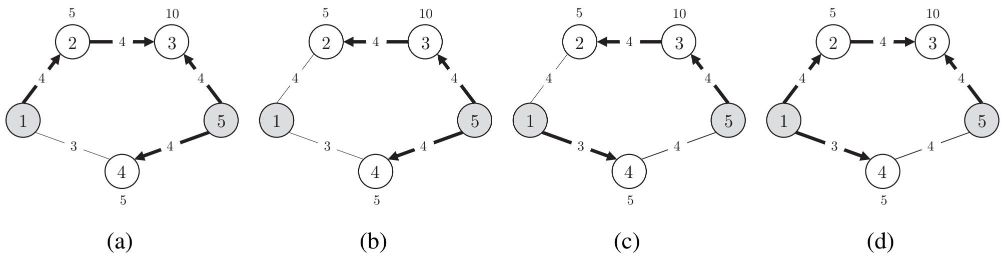


Figure 3.3: A counterexample: Equilibria

The two programs are almost alike, but now note that the flow through an arc can be larger than 1 and y is no longer binary.

Lemma 13. *Models 3 and 4 return the same optimal solution and objective function value.*

Proof. Assume that we solve both models optimally. Constraint (3.16) and variables t and x are identical in both models. In constraint (3.14), flows f_{ijk} originate at individual origin nodes and are consumed at destination nodes. Hence for each k the optimal network is a tree rooted at k. This tree can be traced back to the origin nodes of the players who select k and the corresponding f_{ijpk}

Model 4 Competitive PCST: Simultaneous Move Game, Compact

$$\max \quad \sum_{k\in N^+}\sum_{n\in P} r_{kn}t_{kn} - \sum_{(i,j)\in A'} c_{ij}y_{ij} \tag{3.13}$$

$$\text{s.t.} \quad \sum_{j\in N} f_{jik} - \sum_{j\in N'} f_{ijk} = \begin{cases} -x_{kp} & \text{if } i = o_p \\ \sum_{p\in P} x_{kp} & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \qquad i\in N,\ k\in N^+ \tag{3.14}$$

$$f_{ijk} \le y_{ij} \qquad (i,j)\in A',\ k\in N^+ \tag{3.15}$$

$$\sum_{p\in P} x_{kp} = \sum_{n\in P} t_{kn} \qquad k\in N^+ \tag{3.16}$$

$$f_{ijk} \ge 0 \qquad (i,j)\in A',\ k\in N^+$$

$$y_{ij} \ge 0 \qquad (i,j)\in A'$$

$$x_{k,p} \in \{0,1\} \qquad k\in N^+,\ p\in P$$

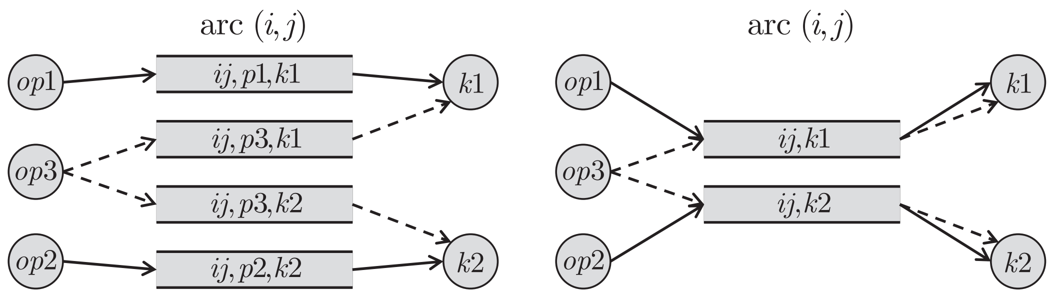
$$t_{kn} \in \{0,1\} \qquad k\in N^+,\ n\in P$$

can be retrieved. In constraint (3.9), flows are already defined for each player, hence the optimal network for each selected k and selecting p would be a path from o_p to k. The union of these paths would indeed give the optimal tree enforced by (3.14) and we can retrieve $f_{ijk} = \sum_p f_{ijpk}$.

Since constraint (3.10) returns if a given (i,j) is built by player p, we can obtain the number of players building the arc by computing $\sum_p y_{ijp}$. On the other hand, y_{ij} is not necessarily equal to the number of players building arc (i,j) but is equal to that value at optimality. See Figure 3.4 for an example.

Considering that we have a maximization problem, constraint (3.15) implies $y_{ij} = \max_k f_{ijk} = \max_k\{\sum_p f_{ijpk}\}$. Similarly, (3.10) implies $y_{ijp} = \max_k f_{ijpk}$. Hence our claim is that, at optimality, $\max_k\{\sum_p f_{ijpk}\} = \sum_p(\max_k f_{ijpk})$. As already defined, $\sum_p(\max_k f_{ijpk})$ is the number of players that use arc (i,j). And it is easy to show that $\sum_p(\max_k f_{ijpk}) \ge \max_k\{\sum_p f_{ijpk}\}$. For $\max_k\{\sum_p f_{ijpk}\}$ to correspond to the same value, there should be some k such that all distinct players using arc (i,j) should select k.

For a contradiction, assume otherwise. Let k_1 be the node that is selected by the largest number of players that use arc (i,j) to send flow to it. Let the set of those players be P_1, and note that



(a) $f_{ij,p_1,k_1} = f_{ij,p_3,k_1} = f_{ij,p_3,k_2} = f_{ij,p_2,k_2} = 1$, $y_{ij,p_1} = y_{ij,p_2} = y_{ij,p_3} = 1$

(b) $f_{ij,k_1} = f_{ij,k_2} = 2$, $y_{ij} = 2$

Figure 3.4: A suboptimal solution: $\sum_p y_{ijp} = 3 > 2 = y_{ij}$

$|P_1| = \max_k\{\sum_p f_{ijpk}\}$. The claim is that P_1 does not include all the players using arc (i,j). Then there should be some destination node k_2 and a corresponding set of players P_2 who use arc (i,j) to send flow to k_2 such that $P_2 \setminus P_1 \neq \emptyset$. Then there exist players $p_1 \in P_1 \setminus P_2$ and $p_2 \in P_2 \setminus P_1$. In other words, $f_{ij,p_1,k_1} = 1$, $f_{ij,p_1,k_2} = 0$, $f_{ij,p_2,k_1} = 0$ and $f_{ij,p_2,k_2} = 1$. This result is feasible, but is suboptimal as we can take the path from j to k_2 that belongs to player p_2 and give it to player p_1. By doing this we do not change the number of occupants at node k_2, yet (potentially) prevent p_2 from building (i,j) which increases the objective. Therefore at optimality $P_2 \setminus P_1 = \emptyset$ and $y_{ij} = \sum_p y_{ijp}$. □

3.5.3 Stackelberg Game: MIP Model

Frequently, we are not just interested in finding an equilibrium solution to the game but we also wish to assume the role of one of the players. There are different such game settings. Here, as in Chapter 2, we consider the *Stackelberg* type competition, a game setting in which one of the players has the chance to make the first move, thus is the *leader* (L) in the game, and the other player is her *follower* (F). Furthermore, we assume the role of the leader, and the role of the follower is simply optimizing his own objective after the leader's strategy is revealed. On the contrary, the leader has to account for her opponent's decision and has to optimize her objective expecting that the follower's objective (reaction) would be optimal given hers. This type of model (for two players) is referred to as *bilevel programming* in the mathematical programming literature. The integer programming version is referred to as *integer bilevel linear programming* (IBLP).

Letting $G(N_p, E_p)$ denote the subgraphs built by players $p \in \{L, F\}$, a bilevel program for the leader's problem is sketched below in Model 5. The leader maximizes her profit (3.17), which is dependent on the nodes she and the follower select and arcs she builds. The resulting network should be a connected (3.19) subgraph of $G(N, E)$, include leader origin o_L but not follower origin o_F as (3.18) suggests. Moreover, the subset of nodes that the follower selects should solve the optimization problem — the same problem from follower's point of view — as in (3.20).

Model 5 Competitive PCST, Bilevel Program Sketch

$$\max_{N_L, E_L} \quad \mathit{revenue}(N_L, N_F) - \mathit{cost}(E_L) \tag{3.17}$$

$$\text{s.t.} \quad E_L \subset E,\ N_L \subset N,\ o_L \in N_L,\ o_F \notin N_L \tag{3.18}$$

$$G(N_L, E_L) \text{ is connected} \tag{3.19}$$

$$\begin{aligned} N_F \in \arg \max_{N_F, E_F} \quad & \mathit{revenue}(N_F, N_L) - \mathit{cost}(E_F) \\ \text{s.t.} \quad & E_F \subset E,\ N_F \subset N,\ o_F \in N_F,\ o_L \notin N_F \\ & G(N_F, E_F) \text{ is connected} \end{aligned} \tag{3.20}$$

Solving the leader's problem optimally for combinatorial optimization problems like PCST (as follower's problem) seems impractical using current algorithmic technology. Instead, we are going to focus on developing a heuristic reformulation of the follower's problem to obtain a reasonable *single-level* MIP model for the leader's problem. In Chapter 2, we used the greedy algorithm to determine the follower's response but kept his decision space intact. He could open any facility but the facilities were picked using a heuristic that potentially yielded suboptimal solutions. Now, we would like to look at the problem from a different perspective. We will restrict the decision space of the follower, but are going to solve his optimization problem exactly over this restricted decision space.

The idea is as follows. Given graph $G(N, E)$, the follower's decision space is restricted to a directed tree $T(N_T, A_T)$ that is rooted at o_F. It consists of nodes $N_T \subset N \setminus o_F$ and arcs $A_T \subset A$ whose direction is from the root to the leaves. Then the optimization problem in (3.20) becomes Model 6.

Model 6 Competitive PCST, Follower's Restricted Response

$$\max_{N_F, A_F} \quad revenue(N_F, N_L) - cost(A_F) \tag{3.21}$$

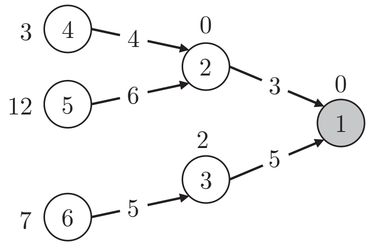
$$\text{s.t.} \quad A_F \subset A_T,\ N_F \subset N_T,\ o_F \in N_F \tag{3.22}$$

$$G(N_F, A_F) \text{ is connected} \tag{3.23}$$

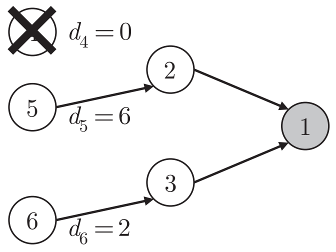
The good thing about Model 6 is that we can solve this problem optimally using a heuristic that runs in $\mathcal{O}(|T|)$ (with given ordering) time. Next, we will describe the heuristic and show that it finds the optimal solution. Then, we will describe a feasibility problem — a set of linear constraints without an objective function and show that this problem is equivalent to the heuristic and hence to Model 6.

The heuristic (see Algorithm 3.2) relies on the observation that the optimal PCST does not have any unprofitable subtrees. That is, starting with the origin and continuing through children, every subtree is profitable. The heuristic is modified from the *strong pruning* algorithm by Johnson et al. (2000), who devised it as an improvement step for the primal-dual algorithm for PCST proposed in Goemans and Williamson (1995). Recall that T is defined to be rooted at o_F. We also define the subtree of T rooted at $i \in N_T$ as T_i. Hence $T = T_{o_F}$. Furthermore, let S denote the candidate solution with current knowledge and R denote the currently unchecked portion of T. Both S and R will be initialized as T and are going to stay as trees throughout the iterations. Finally, let S_i be the subtree of S which is rooted at i and d_i be the profit from adding it.

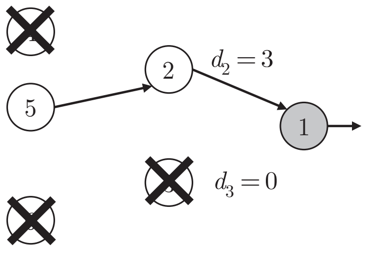
The algorithm proceeds as follows. At each iteration, a leaf i of the remaining tree R is checked. If it is profitable to select $S_i \cup (parent(i), i)$, S remains intact and the profit is carried to the parent of i. Otherwise, the entire $S_i \cup (parent(i), i)$ is pruned from S. At the end of the iteration node i and arc $(parent(i), i)$ is removed from R, hence will not be considered in the subsequent iterations. See Figure 3.5 for an example



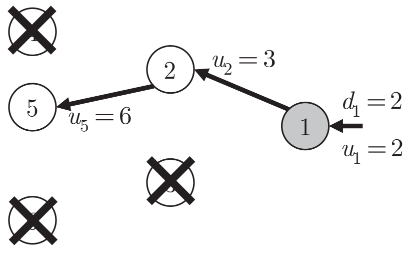
(a) Setting



(b) $S_4 = \emptyset$, $S_5 = 5$, $S_6 = 6$,
$R = \{1, 2, 3;\ (1, 2), (1, 3)\}$



(c) $S_3 = \emptyset$, $S_2 = \{2, 5;\ (2, 5)\}$,
$R = 1$, Terminate



(d) Return $S_1 = \{1, 2, 5;\ (1, 2), (2, 5)\}$,
Actual subtree profits: u

Figure 3.5: Strong pruning

Algorithm 3.2 Strong Pruning for Follower Problem

```
given leader solution $x_i^L$ for each $i \in N_T$
update revenue: let $r_i \leftarrow (1 - 0.5x_i^L)r_i$
let $S = T(N_T, A_T)$, $R = T(N_T, A_T)$, and $d_i = 0$ for all $i \in N_T$
while $R \neq o_F$ do
    let $i$ be a leaf of $R$
    let $d_i = \max\{d_i + r_i - c_{parent(i),i}, 0\}$
    if $d_i > 0$ then
        let $d_{parent(i)} = d_{parent(i)} + d_i$
    else
        remove $S_i$ and arc $(parent(i), i)$ from $S$
    end if
    remove node $i$ and arc $(parent(i), i)$ from $R$
end while
return $S$ as optimal PCST on $T$ and with objective $d_{o_F}$
```

Before showing that the algorithm returns the optimal PCST on a tree, let us discuss some properties of this restricted problem. First, consider $T(N, A)$ rooted at some node o. Since each node $i \in N \setminus o$ has a single arc $(parent(i), i)$ entering it, selecting the node is equivalent to selecting the entering arc. Consider a subtree T_i of T. The profit of selecting any subset $N^{'}$ from the nodes in T_i is $\sum_{j \in N'}(r_j - c_{parent(j),j})$, hence the optimal solution for T_i is independent of the decisions made for any $j \notin T_i$. However, a feasible solution for the overall problem should be connected to the root node. Therefore, if $parent(i)$ or any node on the path from o to i is not selected, entire T_i would not be selected, either.

Theorem 14. *Algorithm 3.2 solves the PCST on a tree optimally.*

Proof. By induction on i from leaves to root. We start with the leaves of T. Let i be a leaf of T, so i has no children. Then, if $r_i - c_{parent(i),i} > 0$, hence adding node i is profitable, we carry the profit to the parent. Because if the parent of i is selected, it will be profitable to select i, too. If not, we remove i from further consideration. Any feasible solution that selects i can simply be improved by not selecting it it. At the end of the iteration S_i is either i or empty. This is exactly what the algorithm does for the leaves of T.

Assume we find the optimal solution for all the children j of a given node i. It is easy to see that the algorithm goes through all the children before considering a parent. Optimal solutions S_j are assumed to be found, so associated profits d_j are already carried to the parent from profitable children. The optimal decisions for the subtrees rooted at the children of i are independent of the decision for i as mentioned before. $d_i + r_i - c_{parent(i),i}$ equals the profit of selecting node i and all the optimal solutions of the children. Currently, $S_i = i \bigcup_j (S_j \cup (i,j))$. If the profit is positive, we select i and the optimal solutions of the children, hence S_i remains untouched. Otherwise, for any feasible solution that selects i, we can come up with a better solution that does not select i, hence $S_i = \emptyset$ after the removal.

Letting i be the origin, we obtain the optimal solution S. $\square$

Now, we represent this algorithm as a set of linear constraints. The leader's decision variables are the same in Model 2. We add the superscript L $[F]$ to x in order to distinguish the leader [follower]. We also define three new variables to represent the strong pruning algorithm in the model.

Variables:

x^L_j : If leader selects node j, $x^L_j = 1$ and 0 otherwise

y_{ij} : If leader selects arc (i,j), $y_{ij} = 1$ and 0 otherwise

f_{ijk} : Leader's flow through arc (i,j) with destination k

x^F_j : If follower selects node j, $x^F_j = 1$ and 0 otherwise

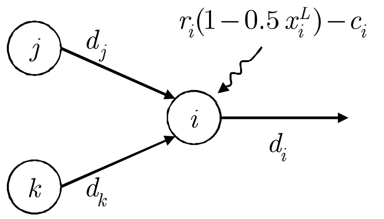
d_i : (Potential) follower profit transferred from node i to parent of i

u_i : (Actual) follower profit transferred from parent of i to node i

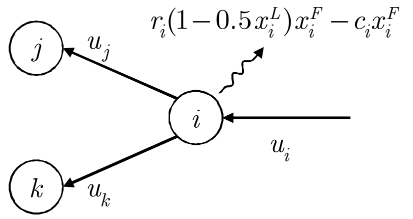
w_i : If both players select node i, $w_i = 1 = x^L_j x^F_j$ and 0 otherwise

The potential profit flows "downward" from the leaves to the root node as in Figure 3.6a. At each evaluated node i potential profit d_i is calculated as in (3.24). It is the maximum of 0 and the sum of the potential profit from the children plus its own profit. Note also that d is nonnegative.

$$d_i \geq \sum_{j:(i,j)\in T} d_j + r_i(1 - 0.5x^L_i) - c_{parent(i),i} \forall i \in T \tag{3.24}$$



(a) Potential profit d_i, "down"



(b) Actual profit u_i, "up"

Figure 3.6: Two way profit flow

(3.24) imposes no upper bound on the value of d_{o_L}. We would like to balance this with a reverse, hence "upward," flow from the root node to the leaves as in Figure 3.6b, which would eventually be equal to the actual profit. (3.25) says that the actual profit would be equal to the potential profit at the root node, but it would be less than the potential profit everywhere else. If i is selected, profit u_i is reduced by the i's own profit and is passed to its children conditional on that the parent is selected (3.26).

$$u_i \begin{cases} = d_i & \text{if } i = o_F \\ \le d_i & \text{otherwise} \end{cases} \qquad \forall i \in T \tag{3.25}$$

$$u_i = \sum_{j:(i,j)\in T} u_j + r_i(x_i^F - 0.5w_i) - c_{parent(i),i}x_i^F \qquad \forall i \in T \tag{3.26}$$

Since a feasible solution needs to be a tree with root o_F, if a node is selected, its parent should also be selected (3.27).

$$x_j^F \le x_i^F \forall (i,j) \in T \tag{3.27}$$

Note that $w_i = x_i^L x_i^F$ and (3.28) linearizes this constraint.

$$w_i \begin{cases} \le x_i^L \\ \le x_i^F & \qquad \forall i \in T \\ \ge x_i^L + x_i^F - 1 \end{cases} \tag{3.28}$$

Finally, the flows u and d are nonnegative and selection variables x^F and w are binary.

Model 7 Competitive PCST, Stackelberg Game

$$\max \quad \sum_{k \in N^+} r_k(x_k^L - 0.5w_k) - \sum_{(i,j)\in A'} c_{ij}y_{ij} \tag{3.29}$$

s.t. Leader's decision: (3.30)

$$\sum_{j \in N' \cup o_p} f_{jik} - \sum_{j\in N'} f_{ijk} = \begin{cases} -x_k^L & \text{if } i = o_p \\ x_k^L & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in N' \cup o_p,\, k \in N^+ \tag{3.31}$$

$$f_{ijk} \le y_{ij} \qquad \forall (i,j) \in A',\, k \in N^+ \tag{3.32}$$

Follower's response:
Constraints (3.24), (3.25), (3.26), (3.27), (3.28)

$$f_{ijk} \ge 0 \qquad \forall (i,j) \in A',\, k \in N^+$$
$$x_k^L \in \{0,1\} \qquad \forall k \in N^+$$
$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A'$$
$$u_i, d_i \ge 0 \qquad \forall i \in T$$
$$x_k^F, w_i \in \{0,1\} \qquad \forall i \in T$$

Theorem 15. *Constraints* (3.24), (3.25), (3.26), (3.27), (3.28) *are equivalent to strong pruning.*

Proof. Each selectable node has only one parent and we can treat revenue as a parameter after accounting for the leader decision. Therefore, to ease the notation, let c_i and r_i be used instead of $c_{parent(i),i}$ and $r_i(1 - 0.5x_i^L)$, respectively.

Nonnegativity $d \ge 0$ and (3.24) implies $d_i \ge \max\{d_i + r_i - c_i, 0\}$. Therefore, $d_i \ge d_i^*$ for all $i \in T$, where (*) denotes the strong pruning (i.e., optimal) result. By definition, $u_i = \sum_{i \in T_i}(r_i - c_i)x_i^F$, where T_i is the subtree rooted at node i. As $x_i^F \ge x_j^F$ for all $j \in T_i$, we have

$$x_i^F = 0 \Rightarrow u_i = 0 \forall i \in T \tag{3.33}$$

Now, consider a subtree T_i. Potential profit d_i is always nonnegative, hence we have the following two cases:

Case 1: Let $d_i > 0$ and the children of i be $j \in J$. Assume $d_i = u_i$, if $r_i \neq c_i$, then $x_i^F = 1$. Recall that $u_j \leq d_j$, for all j, so we obtain

$$\sum_{j \in J} u_j = u_i - (r_i - c_i)x_i^F = d_i - r_i + c_i \geq \sum_{j \in J} d_j \Rightarrow u_j = d_j \forall j \in J.$$

If $r_i = c_i$, then this implies there is at least one child j with $d_j > 0$ and it becomes the problem in this case again. Continuing in this fashion, there should be at least one node $k \in T_i$ with $r_k \neq c_k$, which would imply $x_k^F = 1$ and, due to constraint (3.27), all the way down to the origin the entire path from k is selected. Hence if $d_i = u_i > 0$, then $x_i^F = 1$.

Case 2: If $d_i = 0$, then $u_i = d_i = 0$ readily. Moreover, $r_i - c_i \leq 0$, otherwise we would have $d_i > 0$. Consequently,

$$r_i - c_i < 0 \Rightarrow x_i = 0 \Rightarrow x_j = 0 \forall j \in T_i. \tag{3.34}$$

Hence, if $d_i = 0$, then either $x_j = 0$ for all $j \in T_i$ and we prune the whole subtree T_i, or any subtree of T_i that is selected should have $r = c$, throughout. Hence it is optimal to prune T_i if $d_i = 0$. This indifference does not cause any problem on the leader side. We assume that the follower does not act "malevolently" and if it would not bring benefit to him (i.e., multiple optima), leaves the decision to the leader.

Now, we start from the root node. By constraint (3.25), $d_{o_F} = u_{o_F}$ is readily given. If $d_{o_F} > 0$, we have an example of Case 1, and $u_j = d_j$ for all j that are the children of o_F. After this, for any node i, either $d_i = 0$ and Case 2 applies, or $d_i > 0$ and Case 1 applies. If $d_{o_F} = 0$, we have Case 2 and it is optimal for the follower not to move.

Note that this implies $u_i = d_i$ for all selected nodes i, and eventually, $u_j = d_j = r_j - c_j$ for j denoting the leaves (nodes without children) of the selected subtree. Therefore, it is not feasible to falsely set some $d_i > \max\{d_i + r_i - c_i, 0\}$ that would be transferred to the origin.

To sum up, the feasible solution to this set of constraints and the declared domains for variables has correctly calculated potential profits d (if they are carried to the origin), which brings correct actual profits u and any nonprofitable subtree is pruned. This is equivalent to strong pruning in effect. $\square$

This approach brings $\mathcal{O}(m)$ additional constraints and variables to the leader's PCST problem, where m is the number of nodes in T. This is not many, considering the original number of constraints the problem has. This model can easily be extended to a case where the follower evaluates multiple trees and responds optimally picking a subtree in one of them. Letting $\mathcal{T}$ be the set of those trees, follower variables are going to have an additional index $t \in \mathcal{T}$. Then the response can be restricted to the best of these trees by simply setting $\sum_{t \in \mathcal{T}} x^F_{o,t} = 1$. Similarly, multiple origins can be included into the model using the dummy arcs trick as in Section 3.4.2.

3.6 Computational Study

We tested our models using 80 randomly generated data sets. The data sets can be divided in three classes. Classes `D`, `R` and `A` stand for "Delaunay", "random" and "additional", respectively. Each class is further divided into subsets consisting 5 instances that have the same properties. Class `D` consists of 25 instances. They are *Euclidean graphs* which have 100 to 500 nodes that are uniformly distributed on a 100-by-100 square, and edges that are formed using *Delaunay triangulation*. Leader and follower origins — one for each — are picked randomly. The number of nodes with positive revenue is set to 25% of all nodes. They are randomly picked excluding origins and their values are generated uniformly in $[\mu, 3\mu]$ where μ is the average edge cost (i.e., length). Class `R` are again Euclidean graphs, but no triangulation is used in their generation. They consist of 25 instances. Nodes are generated in the same manner as class `D`, but the edges are generated randomly keeping the edge-to-vertex ratio constant at 3. These graphs are also connected. We first generate a random spanning tree and then pick the remaining edges. Finally, class `A` consists of 30 additional instances that we use for comparison. We generated them over the 200-node instances in class `R` in order to test the two constants we used in the data generation. In the first 15 of the additional instances, we

Table 3.1: Instance settings

Class	Subset	Nodes	Revenue nodes	Edges	Edge/node ratio	Revenue node percentage
D	1	100	25	284	2.8	25
	2	200	50	582	2.9	25
	3	300	75	883	2.9	25
	4	400	100	1182	3.0	25
	5	500	125	1480	3.0	25
R	1	100	25	300	3	25
	2	200	50	600	3	25
	3	300	75	900	3	25
	4	400	100	1200	3	25
	5	500	125	1500	3	25
A	1	200	50	800	4	25
	2	200	50	1000	5	25
	3	200	50	1200	6	25
	4	200	60	600	3	30
	5	200	70	600	3	35
	6	200	80	600	3	40

varied the edge-to-node ratio in $[3, 5]$ and added additional randomly picked edges. In the second 15, we changed the percentage of revenue nodes in $[25, 40]$ by generating revenue data for nodes with 0 revenue in the original setting. Properties of these data sets (averaged over subsets) are summarized in Table 3.1.

3.6.1 Solution Statistics

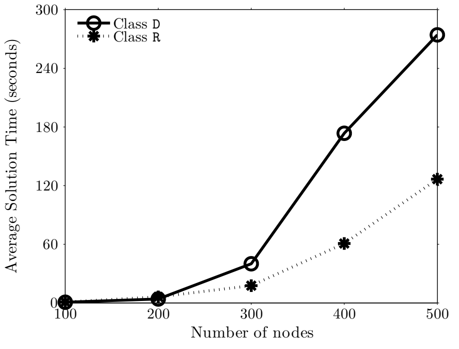
We coded the model using GAMS and solved the instances using CPLEX 11.1 on a Intel T7500 2.2 GHz 2GB 32-bit computer. Model size and solution statistics of each subset (averaged over 5 instances) of classes `D` and `R` are tabulated in Tables 3.2 and 3.3.

We first solve the two-player model (Model 7). We use two trees as the follower's response basis. The first one is generated by finding the minimum cost spanning tree for the follower and then pruning unprofitable subtrees using strong pruning. The second one is generated by finding the PCST for the follower, hence solving Model 2 for the follower. The values that are reported under "Stackelberg" correspond to the average of these. After solving the Stackelberg model, we fix the leader's variables at their suggested levels, and solve PCST for the follower in order to compute the error generated by the restriction of the follower's response. In order to compare the leader's loss from competition, we solve the leader's single-player problem (Model 2), too. Since the leader and follower problems are identical with only a few parameter changes, we report them together. "PCST (L & F)" correspond to the average of these two runs. Then, we solve the equilibrium finding model (Model 4) and report the results under "Simultaneous". Finally, we solve the central model and reported under "central".

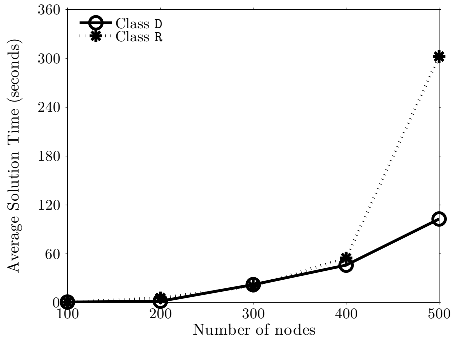
The sizes of the two classes are similar to each other. However, if we look at the solution statistics in Table 3.3, we see that the instances we obtained through triangulation take longer to solve than the random instances. An interesting result is that the instances were almost always solved at the root node, without any LP gap. The solution time with respect to instance type for each problem type is plotted in Figure 3.7. As expected, solution time increases rapidly with increasing problem size. The simultaneous-move game model is the most time consuming one, whereas the rest are similar to each other. Among all instances two (both Stackelberg) were solved in longer than 1 hour. The twist in Figure 3.7d is caused due to the removal of those instances while plotting the figure.

The same process is followed for the `A` instances. These instances were generated from `R2` by adding arcs or adding revenue nodes. The solution statistics for class `A` is provided in Table 3.4 and solution times of each type and for each control parameter is illustrated in Figure 3.8. The change in the number of arcs (hence the edge-to-node ratio) affects the solution time more than the percentage (hence number) of revenue nodes.

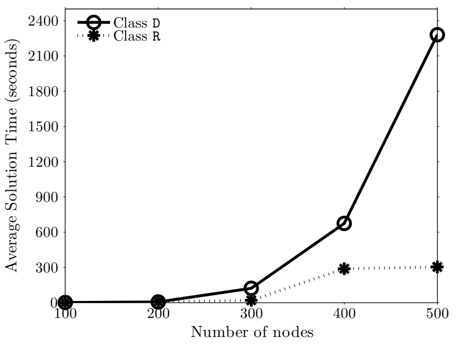
Finally, we compare the profits obtained in the equilibrium and central designs. According to Figure 3.9a, the loss is less (percentage is high) for the triangulation graph. It increases with increasing number of arcs, but stays stable for increased number of nodes. Figure 3.9b illustrates leader's error obtained by calculating $\frac{|\text{Expected}-\text{Suggested}|}{\text{Suggested}}$, where "suggested" is the leader's profit as a result of solving the Stackelberg model and "expected" is the leader's profit after follower responds optimally. This error is significantly less for the random instances. The two lines represent



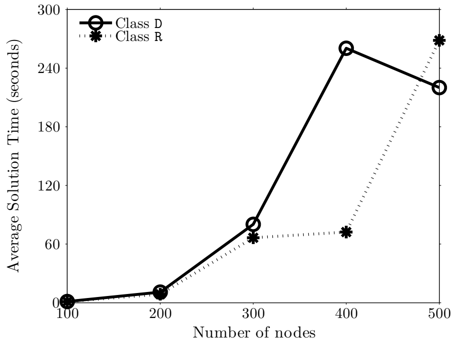
(a) PCST



(b) Central

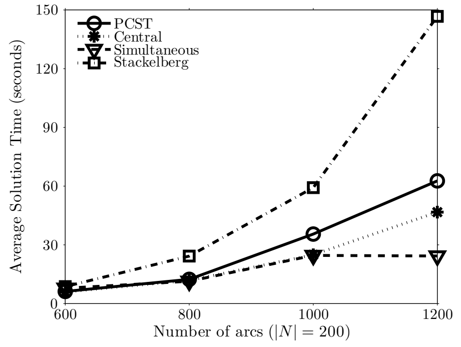


(c) Simultaneous-move

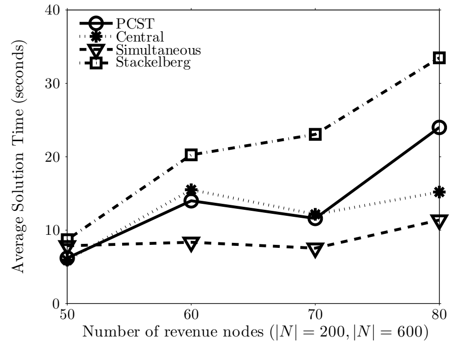


(d) Stackelberg

Figure 3.7: Solution time change

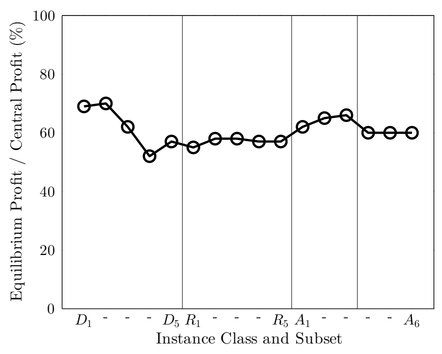


(a) Edge-to-node ratio

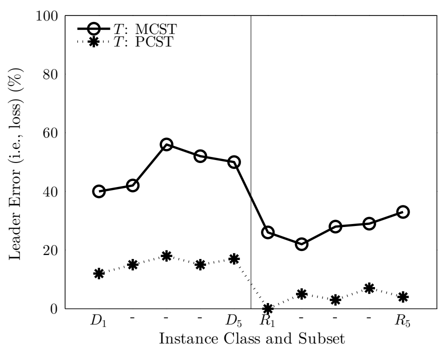


(b) Revenue node percentage

Figure 3.8: Sensitivity analysis



(a) Loss from competition



(b) Leader's error

Figure 3.9: Sensitivity analysis

two different trees, T, for follower response. Using the PCST of the follower improves the result significantly.

3.7 Conclusions

We discussed network formation and design under competition. We introduced a network formation game, characterized optimal strategies for individual players and proposed an iterative method and a mathematical programming formulation that finds a pure strategy equilibrium solution in order to anticipate the resulting network. The equilibrium is not unique and the resulting equilibrium solution, however, is not an extreme equilibrium—the least or most profitable. Therefore, the problem of measuring the best/worst case deviation from a centralized solution remains open.

Next, we introduced a leader-follower version of the model. This problem is inherently a more difficult problem as discussed in Chapter 1. In order to obtain near optimal results within short solution times, we proposed a reformulation approach. We restricted the follower's response space to a tree (or set of trees) and demonstrated that strong pruning solves the follower's problem given this restriction. Then we expressed the follower's restricted optimization problem as a feasibility problem by converting the strong pruning algorithm to a group of linear constraints. The heuristic reformulation has a single-level problem and is more tractable and can be solved by standard MIP methods.

We considered the same game for both players, however this reformulation, and in general similar reformulation strategies to describe the follower problem, can be implemented in various scenarios. We are currently developing an implementation for network interdiction using the ideas presented in this study. In the future we would like to develop an exact optimization method that employs this solution technique iteratively in an algorithm.

Table 3.2: Model statistics: Classes D and R

Class	D					R				
Subset	1	2	3	4	5	1	2	3	4	5
PCST (L & F)										
Constraints	16261	46844	153796	274741	430276	16966	69121	155881	277661	434801
Variables	14387	40872	133273	237389	371094	15121	60454	135385	240339	375655
Binary variables	601	988	1902	2548	3193	630	1283	1929	2578	3229
Central										
Constraints	16426	26636	154216	275401	431126	17156	69431	156436	278581	435901
Variables	14534	23372	133623	237956	371826	15293	60721	135873	241168	376639
Binary variables	583	706	1832	2455	3075	612	1240	1862	2487	3113
Simultaneous										
Constraints	16451	26666	154291	275501	431251	17181	69481	156511	278681	436026
Variables	14583	23431	133772	238155	372075	15342	60820	136022	241367	376888
Binary variables	50	60	150	200	250	50	100	150	200	250
Stackelberg										
Constraints	16548	47291	154760	276068	431941	17272	69726	156768	278803	436278
Variables	14531	41093	133757	238056	371930	15272	60753	135822	240898	376380
Binary variables	663	1082	2110	2836	3553	693	1409	2112	2810	3532

Table 3.3: Solution statistics: Classes D and R

Class	D					R				
Subset	1	2	3	4	5	1	2	3	4	5
PCST (L & F)										
Solution time	0.9	4.0	40.1	173.6	274.2	0.9	6.2	17.7	60.8	126.6
Simplex iterations	4633	9810	34766	66805	97518	4930	18505	41140	91350	133592
B&B nodes	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Central										
Solution time	0.7	1.8	22.2	46.1	102.8	0.7	5.9	20.2	54.8	302.3
Simplex iterations	3492	5440	27473	45250	64155	3765	16452	33247	79707	138647
B&B nodes	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Simultaneous										
Solution time	1.4	6.3	121.8	674.8	2280.8	1.0	7.9	19.9	288.3	303.8
Simplex iterations	5771	11304	63960	126534	220913	5850	24774	54917	119681	183345
B&B nodes	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Stackelberg										
Solution time	1.2	10.9	80.2	656.3	1880.2	1.1	8.7	66.5	72.3	268.4
Simplex iterations	5302	16444	49684	90312	118595	5888	24863	67717	103221	174072
B&B nodes	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 3.4: Solution statistics: Class A

Class			A			
Subset	1	2	3	4	5	6
PCST (L & F)						
Solution time	12.4	35.5	62.7	14.0	11.6	24.0
Simplex iterations	23515	39033	49209	22571	22869	27430
B&B nodes	0.0	0.0	0.0	0.0	0.0	0.0
Central						
Solution time	11.4	24.6	24.3	8.3	7.5	11.4
Simplex iterations	24739	36207	29175	21074	20848	25977
B&B nodes	0.0	0.0	0.0	0.0	0.0	0.0
Simultaneous						
Solution time	11.9	24.9	46.7	15.5	12.2	15.2
Simplex iterations	31297	53700	74689	31142	33537	39251
B&B nodes	0.0	0.0	0.0	0.2	0.0	0.0
Stackelberg						
Solution time	24.2	59.2	146.7	20.3	23.1	33.5
Simplex iterations	30623	47120	62985	27949	31376	34812
B&B nodes	0.0	0.0	0.4	0.0	0.0	0.0

Chapter 4

A Portfolio Choice Problem: Simultaneous Signaling

4.1 Introduction

The college application and search process and the job application and search process involve not only a candidate's choice of the set of schools or institutions to apply to, but also the set of schools or institutions to signal interest to. For example, in the job market for economics Ph.D.s organized by the American Economic Association, a job market candidate not only chooses the set of schools to which he or she applies, but also the set of (two) schools to which he or she can send, through the Association, signals of interest. Similarly, in the college admissions process, applicants have the option to send costly signals to schools, again to increase the probabilities of admission. Examples of such signals are attending campus tours and information sessions, making phone calls and sending e-mail inquiries, all of which require time and effort from the applicants. Some schools track this information to evaluate the candidates' interest in the school.

The Chronicle of Higher Education, 2010, asks the following questions, while investigating American University's admission figures: "How many applicants would turn down a super-selective, big-name college to attend a somewhat less-selective, less-famous one? How do you know whether a student considers your college a top choice or a ‘safety school’? How does an applicant’s sense of ‘fit’ with a college relate not only to matriculation, but also retention?” The article continues, “In recent years, such questions have prompted American’s admissions teams to look more closely at ‘demonstrated interest,’ the popular term for the contact students make with a college during the application process, such as by visiting the campus, participating in an interview, or e-mailing an admissions representative.” (Hoover, 2010)

In the market for Ph.D. economists, Coles et al. (2010a), with data from the American Economic Association, examine the effectiveness of the AEA signaling mechanism; and in the college admissions process, Dearden et al. (2011), in their empirical section, with data from a highly-selective medium-sized university, examine the effectiveness of “demonstrated interest.” In theoretical analyses of market signaling, Avery and Levin (2010), Coles et al. (2010b), Dearden et al. (2011), and Kushnir (2010) examine signaling and institutional offer game-theoretic models. While these models offer insights into signaling decisions, they are restrictive in that each applicant is permitted to send only one signal. Given the established empirical importance of signaling, an interesting issue is the characterization of optimal signaling decisions by a decision maker in a model in which the choice of the number of signals and which schools to signal is endogenous.

In this chapter, we consider a setting in which a decision maker – either a student applying to colleges and universities or a graduate applying for jobs – has submitted applications to various institutions. After submitting the applications, the decision maker has the option to send a costly signal to each institution. We analyze an individual’s simultaneous choice about the set of institutions to signal. In doing so, we generalize the results of Chade and Smith (2006). Chade and Smith analyze the problem of characterizing the set of institutions to which a decision maker optimally applies, assuming that not applying to an institution results in zero probability of an offer from that institution. Their model, however, could be used to describe the signaling choice problem. But, in the context of signaling, assuming a zero probability of receiving an offer following no signal is unrealistic. To consider the signaling decision problem, we generalize their model by permitting the probability of receiving an offer to be positive (or possibly zero) following no signal.

We demonstrate that the optimal algorithm to solve the decision maker's problem identified by Chade and Smith for their specific environment also is optimal in this more general environment. Incidentally, they generalize Stigler (1961). Moreover, we pose the same problem as a longest path problem and propose a dynamic programming algorithm.

The optimal decision rule is a "greedy algorithm", which Chade and Smith appropriately term for an economics audience a "marginal improvement algorithm." This algorithm has the advantage of being simple; that is, it is easy to characterize and to understand. Furthermore, for the N school case, the algorithm executes in polynomial time, amounting to $O(N^3)$ operations. Therefore, the solution to a student's decision about the set of schools to "demonstrate interest" or the new economics Ph.D.'s decision about the schools to signal through the AEA mechanism is simple and easy to characterize.

The key to their proof of the optimality of the marginal improvement algorithm is the "downward recursive" nature of the decision maker's gross payoff function. (In their model, as in ours, the decision maker's utility equals gross payoff less cost.) In our more general environment, however, the decision maker's gross payoff function is no longer downward recursive. But, as we demonstrate, a monotonic transformation of the gross payoff function does satisfy this property. We use the downward recursive nature of this monotonic transformation in the proof of the optimality of the marginal improvement algorithm.

Our sequence of lemmas and theorem for the greedy algorithm follows a similar path to that of Chade and Smith. However, we provide a simpler proof for the optimality of the greedy algorithm. The dynamic programming algorithm we propose relies on the observation that the problem may be posed as a nonconventional longest path problem. It offers a better time complexity ($O(N^2)$) using the same memory ($O(N)$).

4.2 The Model

4.2.1 The Signaling Problem

A student has applied to a finite set $N = \{1, 2, ..., N\}$ of schools. We avoid using set notation by writing, for example, $i = \{i\}$, $A + B = A \cup B$, $A - B = A \setminus B$, $A \wedge B = A \cap B$, and $(i, j) = \{k \in N \mid i < k < j\}$. We abuse notation by letting N represent both a natural number and a set N. The cardinality of the set of subsets of N is 2^N. The student's decision problem in this model is to choose a subset of schools to which he or she signals. Let $f : 2^N \longmapsto \mathbb{R}_+$ be a supermodular function. Interpret $f(S)$ as the expected gross payoff to the student of signaling a subset S of the schools. We let ρ_A^S denote the probability of being rejected by each of the schools in A, $A \subseteq N$, given that the student signals each of the schools in S, $S \subseteq N$. We let $\rho_A \equiv \rho_A^{\emptyset}$ denote the probability of being rejected by each of the schools in A, given the student has signaled none of the schools in A. We assume for each school $i \in N, \rho_i \leq \rho_i^i$. Furthermore, $\rho_A^S = (\prod_{i \in A \wedge S} \rho_i^i)(\prod_{i \in A - S} \rho_i)$. We label the schools by the student's ex post utility, $u_1 > u_2 > ... > u_N$. The student's gross payoff of signaling the schools in S is:

$$f(S) = \sum_{k=1}^{N} \rho_{[1,k)}^S (1 - \rho_k^S) u_k.$$

We assume that the cost of signaling a portfolio of schools, S, is a function of the cardinality of S, $c(|S|)$, where $|S|$ denotes the cardinality of S. We assume that c is increasing and convex and that $c(\emptyset) = 0$.

We examine the choice of S to maximize the student's net payoff, $v(S) = f(S) - c(|S|)$. We assume that $f(i) - c(1) > f(\emptyset)$ for at least one i. Let $\Sigma^*(N)$ solve

$$\max_{S \subseteq N} v(S), \tag{4.1}$$

and denote $\Sigma^* \equiv \Sigma^*(N)$.

In our analysis, we also make use of a special case in which the student signals exactly n schools from the set of alternatives N. For this special case, the cost function is $c(|S|) = 0$ for $|S| \leq n$ and $c(|S|) = \infty$ for $|S| > n$. For this case, we let $\Sigma_n(N)$ denote the solution to (4.1). Further, we define $\Sigma_n \equiv \Sigma_n(N)$.

From f, we define the function, $g : s^N \longmapsto \mathbb{R}_+$, which equals the decision makers's net expected payoff of signaling of signaling S less his or her or expected payoff of signaling the null set, $\emptyset$. That is, $g(S) \equiv f(S) - f(\emptyset)$. In our proof of the solution to (4.1), which involves the maximization of $f(S) - c(|S|)$, we analyze the function g.

4.2.2 Properties of the Payoff Functions, f and g

We define that U *is above* L, written $U \sqsupseteq L$, if the worst prize in U is better than the best prize in L. The function g is *downward recursive* (DR) as explained in the following property.

Property 16. *Given sets of signaled options U and L in N that satisfy $U \sqsupseteq L$, we have that g is DR:*

$$g(U + L) = g(U) + \frac{\rho_U^U}{\rho_U} g(L). \tag{4.2}$$

Proof. We begin with:

$$g(U + L) = f(U + L) - f(\emptyset) = \sum_{t=1}^{N} \rho_{[1,t)}^{U+L}(1 - \rho_t^{U+L})u_t - f(\emptyset). \tag{4.3}$$

We partition $\{1, ..., N\}$ into two sets $\{1, ..., l_1 - 1\}$ and $\{l_1, ..., N\}$, where l_1 is the best alternative in L, and rewrite (4.3) to obtain:

$$= \sum_{t=1}^{l_1-1} \rho_{[1,t)}^{U}(1 - \rho_t^{U})u_t + \sum_{t=l_1}^{N} \rho_{[1,t)}^{U+L}(1 - \rho_t^{U+L})u_t - f(\emptyset). \tag{4.4}$$

We add and subtract the same expression from (4.4) to obtain:

$$\begin{aligned}
&= \sum_{t=1}^{l_1-1} \rho^U_{[1,t)}(1-\rho^U_t)u_t + \sum_{t=l_1}^{N} \rho^{U+L}_{[1,t)}(1-\rho^{U+L}_t)u_t - f(\emptyset). \\
&\quad + \sum_{t=l_1}^{N} \rho^U_{[1,t)}(1-\rho_t)u_t - \sum_{t=l_1}^{N} \rho^U_{[1,t)}(1-\rho_t)u_t. \qquad (4.5)
\end{aligned}$$

We rearrange the terms in (4.5) and use these equalities: first, $\rho_t^{U+L} = \rho_t^L$ for any $t \in [l_1, N]$; second, $\rho_t = \rho_t^U$ for any $t \in [l_1, N]$; third, $\rho^U_U \rho^L_{[1,t)} = \rho_U \rho^{U+L}_{[1,t)}$; and fourth $\rho^U_U \rho_{[1,t)} = \rho_U \rho^U_{[1,t)}$. In doing so, we obtain:

$$\begin{aligned}
&= \sum_{t=1}^{N} \rho^U_{[1,t)}(1-\rho^U_t)u_t - f(\emptyset) \\
&\quad + \frac{\rho^U_U}{\rho_U}\left(\sum_{t=l_1}^{N} \rho^L_{[1,t)}(1-\rho^L_t)u_t - \sum_{t=l_1}^{N} \rho_{[1,t)}(1-\rho_t)u_t \right). \qquad (4.6)
\end{aligned}$$

We use the fact that the first $l_1 - 1$ terms of $f(L)$ and $f(\emptyset)$ are the same and thus cancel each other in $g(L) = f(L) - f(\emptyset)$ to find:

$$= g(U) + \frac{\rho^U_U}{\rho_U} g(L). \qquad (4.7)$$

which shows that Property (4.2) holds. □

We also have that g satisfies the following multiplicative property. For any $U \sqsupseteq M \sqsupseteq L$, we have:

$$\begin{aligned}
g(U+M+L) &= g(U+M) + \frac{\rho^{U+M}_{U+M}}{\rho_{U+M}} g(L) \\
&= \left(g(U) + \frac{\rho^U_U}{\rho_U} g(M) \right) + \frac{\rho^{U+M}_{U+M}}{\rho_{U+M}} g(L).
\end{aligned}$$

Because $\dfrac{\rho^S_S}{\rho_S} \leq 1$ and is multiplicative, this function is decreasing in S.

Even though the function g is a monotonic transformation of f, and g is DR, the function f is not DR. We demonstrate by means of a counterexample that the function f is not DR. In this counterexample, consider $N = 2$. To demonstrate that f is not DR, set $f(1+2) = f(1) + \alpha f(2)$ and solve for α. In doing so, we have

$$\alpha = \frac{\rho_1^1(\rho_2 - \rho_2^2)u_2}{(1-\rho_1)u_1 + \rho_1(1-\rho_2^2)u_2},$$

which indicates that f is not DR.

4.3 The Solution

4.3.1 Properties of the Optimal Set

Chade and Smith establish a key property of DR functions – *downward maximization*. We establish that this property extends to f, which is not DR in our environment. However, we do use the monotonic relationship between f and g, which is DR, in our proof of Lemma 17.

Lemma 17. *Let $\Sigma_n = U + L$, where $U \sqsupseteq L$ and L has k elements. Then $\Sigma_k(D) = L$, where D are those options in N that are not better ranked than the best in L.*

Proof. By contradiction. Assume $\Sigma_k(D) \neq L$. That is, there exists an $S \subseteq D$ such that $|S| = k$ and $f(S) > f(L)$. By definition of the set U, $U \sqsupseteq D$, which implies $U \sqsupseteq S$. Property 4.2 states that $g(U+S) = g(U) + \frac{\rho_U^U}{\rho_U} g(S)$. Then,

$$f(S) > f(L) \Leftrightarrow g(S) > g(L) \Leftrightarrow \frac{\rho_U^U}{\rho_U} g(S) > \frac{\rho_U^U}{\rho_U} g(L).$$

Therefore,

$$g(U+S) = g(U) + \frac{\rho_U^U}{\rho_U} g(S) > g(U) + \frac{\rho_U^U}{\rho_U} g(L) = g(U+L).$$

However, $\Sigma_n = U + L$ implies $f(U+L) \geq f(U+S)$ and thus $g(U+L) \geq g(U+S)$, which is a contradiction. Therefore $\Sigma_k(D) = L$. $\square$

We move on to Lemma 18, where we consider any two alternatives i and j for which the ex post preferences, $u_i > u_j$ (i.e., $i < j$), matches the ex ante preferences, $f(i) > f(j)$. For these pairs of alternatives, Lemma 18 states that the marginal values of adding these alternatives to a set $S \subset N - \{i, j\}$ has the same ranking, $MB_i(S) = f(S+i) - f(S) > f(S+j) - f(S) = MB_j(S)$.

Lemma 18. *Assume $f(i) > f(j)$ and $i < j$. Then the marginal benefits of i and j are ordered $MB_i(S) = f(S+i) - f(S) > f(S+j) - f(S) = MB_j(S)$ for any set $S \subseteq N - \{i, j\}$.*

We demonstrate by means of an example that we cannot apply the Chade and Smith proof of their Lemma 2 to establish our Lemma 18. In this example, consider $N = 3$ in which $f(1) > f(3)$. For $\rho_1 = \rho_2 = \rho_3 = 1$, Chade and Smith in the context of this example demonstrate in their Lemma 2 that $f(1+2) > f(2+3)$. To do so, they write the expected utility of signaling schools 1 and 2, and then choosing school 2 if accepted at 2, school 1 if accepted at 1 and rejected by 2, and finally school 3 if accepted by 3 and rejected by 1 and 2. They demonstrate that for the case in which $\rho_1 = \rho_2 = \rho_3 = 1$, this suboptimal policy yields an expected utility that is less than $f(1+2)$ but greater than $f(2+3)$. Specifically, we express the latter inequality in the context of this example for the case in which $\rho_1 = \rho_2 = \rho_3 = 1$:

$$\begin{aligned}&(1-\rho_2^{[1,2]})(f(2) + \rho_2^{[1,2]}((1-\rho_1^{[1,2]})u_1 + \rho_1^s f(3))) \\ &\qquad > (1-\rho_2^{[2,3]})(f(2) + \rho_2^{[2,3]}((1-\rho_3^{[2,3]})u_3 + \rho_3^{[2,3]} f(3))).\end{aligned} \tag{4.8}$$

Therefore, (4.8) does not hold for the case in which $(\rho_1, \rho_2, \rho_3) \neq (1, 1, 1)$.

In continuing with the example, let $u_1 = 2.4$, $u_2 = 2$, and $u_3 = 1.9$. Also let $\rho_1 = 0.9$, $\rho_2 = 0.8$, $\rho_3 = 0.7$, $\rho_1^s = 0.0.8$, $\rho_2^s = 0.7$, $\rho_3^s = 0.6$. For this numerical example, we have $f(1) = 1.1648 > f(3) = 1.1472$. In an attempt to use (4.8) to establish that if $f(1) > f(3)$, then

$f(1+2) > f(2+3)$, we have on the left-hand side of (4.8):

$$(1-\rho_2^{[1,2]})u_2 + \rho_2^{[1,2]}((1-\rho_1^{[1,2]})u_1 + \rho_1^{[1,2]}(1-\rho_3)u_3) = 1.2552.$$

We have on the right-hand side of (4.8):

$$(1-\rho_1)u_1 + \rho_1((1-\rho_2^{[2,3]})u_2 + \rho_2^{[2,3]}(1-\rho_3^{[2,3]})u_3) = 1.2588.$$

Hence, for this example (4.8) does not hold. Based on this example, we therefore develop a new proof of Lemma 18.

Proof. We start with $f(i) - f(j)$ and gradually build $f(S+i) - f(S+j)$. In our notation, we let $u_{N+1} = 0$, which is a dummy option that we include to preserve the integrity of our expression. We express $f(i)$, the expected utility from signaling school i, as:

$$\begin{aligned} f(i) &= \sum_{t=1}^{i-1} \rho_{[1,t)}\,(1-\rho_t)u_t + \rho_{[1,i)}\,(1-\rho_i^i)u_i + \sum_{t=i+1}^{N} \rho_{[1,t)-i}\,\rho_i^i(1-\rho_t)u_t \\ &= (1-\rho_1)u_1 + ... + \rho_{[1,i)}(1-\rho_i^i)u_i + ... + \rho_{[1,N-1]}^i(1-\rho_N)u_N \\ &= u_1 - \rho_1(u_1-u_2) - ... - \rho_{[1,i)}\,\rho_i^i(u_i - u_{i+1}) - ... - \rho_{[1,N]-i}\,\rho_i^i(u_N - u_{N+1}). \end{aligned}$$

We now construct $f(i) - f(j)$. In expressing this difference, we can derive from the above expression that the first $\min\{i,j\} - 1$ terms of $f(i)$ and $f(j)$ are identical. As $i < j$, the first $i-1$ terms of $f(i) - f(j)$ cancel. Note that in this proof, we partition the signaling set S into U, M, and L, where $U \subseteq [1,i)$, $M \subseteq (i,j)$, and $L \subseteq (j,N]$. We have that $f(i) - f(j)$ is expressed as follows:

$$\begin{aligned} &f(i) - f(j) \\ &= \rho_{[1,i)}\left(\sum_{t=i}^{j-1} \rho_{(i,t]}(\rho_i - \rho_i^i)(u_t - u_{t+1}) + \sum_{t=j}^{N} \rho_{(i,t]-j}\,(\rho_i\rho_j^j - \rho_i^i\rho_j)(u_t - u_{t+1})\right) > 0. \end{aligned} \tag{4.9}$$

Note that $\rho_i - \rho_i^i \geq 0$ and $u_t - u_{t+1} > 0$. Therefore, all terms of the summation on the left (that is, for $t = \{i, \ldots, j-1\}$) are positive. Also note that $\rho_i \rho_j^j - \rho_i^i \rho_j$ can be positive or negative. Therefore, all terms of the summation on the right (that is, $t = \{j, ..., N\}$) will either be positive or negative. Our assumption that $f(i) - f(j) > 0$ implies that the cumulative sum up to $n \in \{i, \ldots, N\}$ is always positive, because the negative terms in (4.9) occur only starting with term $t = j$.

We use the following property in the next step of the proof.

Property 19. *Let $x, y \in \mathbb{R}$ such that $x > 0$ and $x + y > 0$, and let $r \in [0, 1]$. Then we have*

$$x + ry > 0, \tag{4.10}$$

because if $y > 0$ then $x + ry > r(x + y) \geq 0$, and if $y \leq 0$ then $x + ry \geq x + y > 0$.

Next, for some $n \in \{i, ..., N\}$ let x be the cumulative sum up to $t = n - 1$, y be the sum over the remaining terms, and r be $\frac{\rho_n^n}{\rho_n} \in [0, 1]$. For $x > 0$, $x + y > 0$ and $r \in [0, 1]$, we can apply (4.10) in Property 19 to the following. We multiply each term in (4.9) for $t > n$ and $t \in S$ by $r = \frac{\rho_n^n}{\rho_n}$ iteratively until all n are accounted for:

$$\rho_{[1,i)} \left(\sum_{t=i}^{j-1} \rho_{(i,t]}^{M} (\rho_i - \rho_i^i)(u_t - u_{t+1}) + \sum_{t=j}^{N} \rho_{(i,t]}^{M+L} (\rho_i \rho_j^j - \rho_i^i \rho_j)(u_t - u_{t+1}) \right) > 0.$$

We rearrange the left-hand-side by splitting the factors $(\rho_i - \rho_i^i)$ and $(\rho_i \rho_j^j - \rho_i^i \rho_j)$, and also by taking the outer factors, $\rho_{[1,i)}$, inside. In doing so, we obtain:

$$\sum_{t=i}^{N} \rho_{[1,t]}^{M+L} \rho_j^j (u_t - u_{t+1}) - \sum_{t=i}^{N} \rho_{[1,t]}^{M+L} \rho_i^i (u_t - u_{t+1}) > 0. \tag{4.11}$$

Finally, we multiply it by $\dfrac{\rho_U^U}{\rho_U} \in [0, 1]$, then add and subtract $u_1 - \sum_{t=1}^{t=i-1} \rho_{[1,t]}^{U} (u_t - u_{t+1})$. (Note that this term is 0 if $U = \emptyset$.) In doing so, we see that the left-hand-side of (4.11) is equal to $f(S + i) - f(S + j)$. Therefore $f(S + i) - f(S + j) > 0$. □

Our next lemma yields a simple insight about Σ^*.

Lemma 20. *Assume $f(i) > f(j)$ and $i < j$. If $j \in \Sigma_n(N)$, then $i \in \Sigma_n(N)$.*

We provide a simple proof of this property.

Proof. By contradiction. Assume $j \in \Sigma_n(N)$, but $i \notin \Sigma_n(N)$. Let $S \equiv \Sigma_n(N) - \{j\}$. Then by Lemma 18, $f(S+i) > f(S+j)$. This implies $f(S+i) > f(\Sigma_n(N))$ which leads to a contradiction as $\Sigma_n(N)$ is the optimal solution. $\square$

4.3.2 The Optimal Algorithm

We move on to establishing that the following greedy algorithm, by an inductive procedure, identifies Σ^*.

Algorithm 4.1 MIA: Marginal Improvement Algorithm

```
let $Y_0 = \emptyset$ and $n = 0$
repeat
   set $n = n + 1$
   choose any $i_n \in \arg\max_{i \in N} f(Y_{n-1} + i)$
   set $Y_n = Y_{n-1} + i_n$
until $f(Y_n) - f(Y_{n-1}) < c(n) + c(n-1)$
return optimal set $Y_{n-1}$
```

The MIA works as follows. The decision maker begins by calculating $f(1)$ through $f(n)$, and includes the option, i, with the greatest value in S. The decision maker then recalculates, determining $f(i+j)$ for each $j \in N - i$. He or she adds the option j that brings the greatest $f(i+j)$. The decision maker continues until he or she hits the point where for each $k \notin S$, $f(S+k) - f(S) < c(|S+1|) - c(|S|)$. Our primary result is:

Theorem 21. *The MIA implements the optimal set Σ^* for problem* (4.1) *with $D = N$.*

We provide a simpler proof than Chade and Smith (2006) based on the following observation. Consider the optimal set with n signaled options, $\Sigma_n(N)$, and let $S \subset \Sigma_n(N)$. If we set $\rho_i = \rho_i^S$ for all $i \in S$ and solve for the best $n - |S|$ options in N with updated probabilities, we obtain

$\Sigma_{n-|S|}(N) = \Sigma_n(N) - S$. We can see that this is true because we simply fix the already known $|S|$ options in the optimal set (because signaling i when $\rho_i = \rho_i^i$ is clearly suboptimal) and solve for the remaining $n - |S|$ best options.

Proof. Assume we are given $\Sigma_n(N)$. Let j be the lowest ranked (i.e., largest indexed) option in $\Sigma_{n+1}(N) - \Sigma_n(N)$. Note that such a j must exist because the $\Sigma_{n+1}(N)$ has one more option even if the rest of the options are common. Let S be the subset of options in $\Sigma_{n+1}(N)$ that are lower ranked than j. Note that S is common to both optimal sets by definition of j, hence $S \subseteq \Sigma_{n+1}(N) \cap \Sigma_n(N)$. Let $\{i_1, \ldots, i_{n-|S|}\}$ be the options in $\Sigma_n(N)$ that are better ranked than j. We will iteratively show that j's being in the optimal set implies that each i_t is in the optimal set for $t \in \{1, \ldots, n - |S|\}$.

Let us start with $t = 1$. First, we update the rejection probabilities $\rho_k = \rho_k^S$ for all $k \in S$. Then, by Lemma 17, we have $\Sigma_1([1, i_1]) = i_1$. This implies $f(i_1) > f(j)$. Then, since $i_1 < j$ and $j \in \Sigma_{n+1-|S|}([1, i_1])$, we have $i_1 \in \Sigma_{n+1-|S|}([1, i_1])$ by Lemma 20.

For the remaining t, we update rejection probabilities $\rho_{i_{t-1}} = \rho_{i_{t-1}}^{i_{t-1}}$. We obtain $\Sigma_1([1, i_t]) = i_t$ by applying Lemma 17. This implies $f(i_t) > f(j)$. Then, since $i_t < j$ and $j \in \Sigma_{n+1-|S|-(t-1)}([1, i_t])$, we have $i_t \in \Sigma_{n+1-|S|-(t-1)}([1, i_t])$ by Lemma 20. Then, we increment t.

When $t = n - |S|$, we obtain $i_{n-|S|} \in \Sigma_2([1, i_{n-|S|}])$. As j is also in the optimal set, this implies $\Sigma_2([1, i_{n-|S|}]) = \{j, i_{n-|S|}\}$, which in turn, implies $\Sigma_{n+1}(N) = \Sigma_n(N) + \{j\}$.

Since $\Sigma_1(N) = \arg\max_{k \in N} f(k)$, Algorithm 4.1 returns $Y_n(N) = \Sigma_n(N)$ for each n. To complete the proof, the stopping rule in Algorithm 4.1 is optimal because $c(n)$ in convex in n and f by Lemma 22 (below) has diminishing returns — $f(A + k) - f(A)$ is decreasing in A for any $k \notin A \subseteq N$. Furthermore, because c is a function of only the cardinality of N, $\Sigma^* = \Sigma_n(N)$ $\square$

For the function f, the marginal benefit of adding j to choice set S is decreasing in S. Specifically, for $S \subset S'$, we have:

$$f(S + j) - f(S) = \rho_{[1,j)}^S \left(\sum_{t=j}^{N} \rho_{(j,t]}^S (\rho_j - \rho_j^j)(u_t - u_{t+1}) \right)$$

$$\geq \rho^{S'}_{[1,j)} \left(\sum_{t=j}^{N} \rho^{S'}_{(j,t]} (\rho_j - \rho^j_j)(u_t - u_{t+1}) \right)$$
$$= f(S' + j) - f(S')$$

We therefore have:

Lemma 22. *The function $f : 2^N \mapsto \mathbb{R}_+$ has diminishing returns.*

4.4 Comparative Statics

We analyze whether students tend to signal schools that are more selective. Specifically, if a school becomes more selective, will a student move from not signaling the school to signaling the school? Similarly, if a school becomes less selective, will a student move from signaling the school to not signaling the school? In answering these questions, which we do in Theorem 23, we change only the probabilities of admission, conditional on signaling and not signaling the school. Note that we do not change the student's utility of the school. Furthermore, to change school i's selectivity, we change the admissions probabilities of school i, ρ^S_i and ρ^{S+i}_i, so that $\rho^S_i - \rho^{S+i}_i$ does not change.

As we demonstrate in the proof to Theorem 23, changing ρ^S_i and ρ^{S+i}_i, without changing $\rho^S_i - \rho^{S+i}_i$, leaves the student's marginal benefit of signaling school i unchanged. However, the marginal benefit of signaling each other school does change. Therefore, while the changes in ρ^S_i and ρ^{S+i}_i may affect Σ^*, Theorem 23 answers the question about whether these changes in probabilities affect the decision to signal school i.

To examine the effect of the change in a school's selectivity, we introduce additional notation. Let f^ρ denote the expected utility and $\Sigma^{\rho *}$ denote the optimal set under the probability structure $\rho = (\rho_1, ..., \rho_N, \rho^1_1, ..., \rho^N_N)$. In Theorem 23, we compare the optimal choice sets under two different probability structures: $\rho = (\rho_1, ..., \rho_N, \rho^1_1, ..., \rho^N_N)$ and $\varrho = (\rho_1, ..., \varrho_i, ..., \rho_N, \rho^1_1, ..., \varrho^i_i, ..., \rho^N_N)$ for which $\varrho_i > \rho_i$, $\varrho^i_i > \rho^i_i$, and $\varrho_i - \varrho^i_i = \rho_i - \rho^i_i$.

Theorem 23. *Consider* $\rho = (\rho_1, ..., \rho_N, \rho_1^1, ..., \rho_N^N)$ *and* $\varrho = (\rho_1, ..., \varrho_i, ..., \rho_N, \rho_1^1, ..., \varrho_i^i, ..., \rho_N^N)$ *for which* $\varrho_i > \rho_i$, $\varrho_i^i > \rho_i^i$, *and* $\varrho_i - \varrho_i^i = \rho_i - \rho_i^i$.

$$(i) \text{ If } i \notin \Sigma^{\rho *}, \text{ then } i \notin \Sigma^{\varrho *}.$$

$$(ii) \text{ If } i \in \Sigma^{\varrho *}, \text{ then } i \in \Sigma^{\rho *}.$$

Proof. Statement (ii) is the contrapositive of statement (i), hence has the same truth value as (i). Therefore, we only prove (i). In our proof, we begin by establishing four properties about the relationship between f^ρ and f^ϱ. With these properties in place, we prove the claim in (i), namely if $i \notin \Sigma^{\rho *}$, then $i \notin \Sigma^{\varrho *}$.

For some $i \in N$, let $\varrho_i - \rho_i = \varrho_i^S - \rho_i^S = \delta$, and note that $\delta > 0$. We now establish the first of four relationships of f^ρ and f^ϱ.

Claim 1: If $f^\rho(S+j) > f^\rho(S+i)$ for some $j < i$ and $S \subseteq N - \{i, j\}$, then $f^\varrho(S+j) > f^\varrho(S+i)$.

We prove this claim using (4.9):

$$\begin{aligned}
& f^\varrho(S+j) - f^\varrho(S+i) \\
&= \rho^S_{[1,j)} \left(\sum_{t=j}^{i-1} \rho^S_{(j,t]} (\rho_j - \rho_j^j)(u_t - u_{t+1}) + \sum_{t=i}^{N} \rho^S_{(j,t]-i} (\rho_j \varrho_i^i - \rho_j^j \varrho_i)(u_t - u_{t+1}) \right) \\
&= \rho^S_{[1,j)} \left(\sum_{t=j}^{i-1} \rho^S_{(j,t]} (\rho_j - \rho_j^j)(u_t - u_{t+1}) + \sum_{t=i}^{N} \rho^S_{(j,t]-i} (\rho_j(\rho_i^i + \delta) - \rho_j^j(\rho_i + \delta))(u_t - u_{t+1}) \right) \\
&= f^\rho(S+j) - f^\rho(S+i) + \rho^S_{[1,j)} \left(\sum_{t=i}^{N} \rho^S_{(j,t]-i} \delta(\rho_j - \rho_j^j)(u_t - u_{t+1}) \right) \\
&> f^\rho(S+j) - f^\rho(S+i) > 0.
\end{aligned} \tag{4.12}$$

This completes the proof of Claim 1.

Next, we move on to our second relationship between f^ρ and f^ϱ. For this relationship, we consider three schools i, j and k, for which $j > i$ and also $k \neq j$. (Recall $\varrho_i > \rho_i$, $\varrho_i^S > \rho_i^S$, and

$\varrho_i - \beta_i^S = \varrho_i - \rho_i^S$.)

Claim 2: If $f^\rho(S+j) > f^\rho(S+k)$ for some $j > i$ and $S \subseteq N - \{j,k\}$, then $f^\varrho(S+j) > f^\varrho(S+k)$.

To prove Claim 2, we first consider $k < j$. Note that the calculations below depend on whether i is greater than, less than, or equal to k. Referring again to (4.9), we have:

$$\begin{aligned} &f^\rho(S+j) - f^\rho(S+k) \\ &= \rho^S_{[1,k)} \left(\sum_{t=k}^{j-1} \rho^S_{(k,t]} (\rho_k^k - \rho_k)(u_t - u_{t+1}) + \sum_{t=j}^{N} \rho^S_{(k,t]-j} \, (\rho_k^k \rho_j - \rho_k \rho_j^j)(u_t - u_{t+1}) \right). \end{aligned}$$

If $k > i$, then only the factor $\rho^S_{[1,k)}$ in this expression changes and we obtain:

$$f^\varrho(S+j) - f^\varrho(S+k) = (1+\delta)(f(S+j) - f(S+k)) > 0.$$

If $k = i$, only the last term in the parentheses changes (note that $\varrho_i - \varrho_i^S = \rho_i - \rho_i^S$), and we obtain

$$\begin{aligned} f^\varrho(S+j) - f^\varrho(S+k) &= f^\rho(S+j) - f^\rho(S+k) \\ &+ \rho^S_{[1,k)} \left(\sum_{t=j}^{N} \rho^S_{(k,t]-j} \, \delta(\rho_j - \rho_j^j)(u_t - u_{t+1}) \right) > 0. \end{aligned}$$

Finally, if $k < i$, both terms in the parentheses change. Note that the term in parentheses below is greater than 0, and we obtain:

$$\begin{aligned} &f^\varrho(S+j) - f^\varrho(S+k) = f^\rho(S+j) - f^\rho(S+k) \\ &+ \rho^S_{[1,i)-k} \delta \left(\sum_{t=i}^{j-1} \rho^S_{(i,t]} (\rho_k^k - \rho_k)(u_t - u_{t+1}) + \sum_{t=j}^{N} \rho^S_{(i,t]-j} \, (\rho_k^k \rho_j - \rho_k \rho_j^j)(u_t - u_{t+1}) \right) > 0. \end{aligned}$$

For the case in which $k > j$, a change in ρ_i or ρ_i^s does not affect the sign of $f(S+j) - f(S+k)$. Therefore, Claim 2 holds for $k > j$. This completes the proof of Claim 2.

Claim 3: Consider a set $S, S \subset \Sigma^{\rho *}$. If $f(S+j) > f(S+k)$ for all $k \in N - S$, then $j \in \Sigma^{\rho *}$.

To prove Claim 3, observe that we can solve for the optimal choice set by the following two-step process. First, create a new probability structure $\hat{\rho}$ by setting ρ_k^s equal to ρ_k for each $k \in S$, where S is a set in $\Sigma^{\rho *}$. Second, for this new probability structure $\hat{\rho}$, solve for $\Sigma^{\hat{\rho}*}(N - |S|)$. Because we proved that MIA finds the optimal choice given any $0 \leq \rho_k \leq \rho_k^s \leq 1$, the MIA can be used to find $\Sigma^{\hat{\rho}*}(N - |S|)$. Then, finding the j such that $f(S+j) > f(S+k)$ for all k is the next step of MIA applied to the probability structure $\hat{\rho}$, which completes the proof of Claim 3.

Claim 4: The marginal benefit of adding i to choice set S is the same under both probability structures, ρ and ϱ.

To prove Claim 4, we have:

$$\begin{aligned} f^{\varrho}(S+i) - f^{\varrho}(S) &= \rho_{[1,i)}^{S} \left(\sum_{t=i}^{N} \rho_{(i,t]}^{S} (\rho_i + \cancel{\delta} - \rho_i^i - \cancel{\delta})(u_t - u_{t+1}) \right) \\ &= f^{\rho}(S+i) - f^{\rho}(S) \end{aligned}$$

We are finished with the four claims.

We now proceed to prove (i) by contradiction. Assume $i \notin \Sigma^{\rho *}$ and $i \in \Sigma^{\varrho *}$. Consider the MIA. We start with $\emptyset$, and step-by-step build our optimal set $\Sigma^{\varrho *}$ under the probability structure ϱ. In the optimal set, $\Sigma^{\rho *}$, let $j_m \in \arg\max_{j \in N} f(\Sigma_{m-1}^{\rho} + j)$ denote the school added at stage j of the MIA under the probability structure ρ.

We begin with step 1. If $j_1 < i$, by Claim 1 and Lemma 20, $j_1 \in \Sigma^{\varrho *}$. If $j_1 > i$, then by Claim 2, $j_1 \in \Sigma^{\varrho *}$. We now have that $j_1 \in \Sigma^{\varrho *}$.

We continue building the optimal set $\Sigma^{\varrho *}$ and arrive at step m, $1 < m < n$, where we attempt to add i to $\Sigma^{\varrho *}$. However, $j_m \in \Sigma^{\varrho *}$ by the same proof as in step 1 and Claim 3 and $j_m \neq i$. Therefore, we cannot add i before $n + 1^{\text{st}}$ iteration.

If we add i at an iteration $r > n$, then by Claim 4 we obtain:

$$f^{\varrho}(\Sigma^{\rho} + \{j_{n+1}, \ldots, j_{r-1}\} + i) - f^{\varrho}(\Sigma^{\rho} + \{j_{n+1}, \ldots, j_{r-1}\})$$

$$= f^{\rho}(\Sigma^{\rho} + \{j_{n+1}, \ldots, j_{r-1}\} + i) - f^{\rho}(\Sigma^{\rho} + \{j_{n+1}, \ldots, j_{r-1}\}). \tag{4.13}$$

By diminishing returns (i.e. Lemma 22), we obtain:

$$f^{\rho}(\Sigma^{\rho} + \{j_{n+1}, \ldots, j_{r-1}\} + i) - f^{\rho}(\Sigma^{\rho} + \{j_{n+1}, \ldots, j_{r-1}\}) \leq f^{\rho}(\Sigma^{\rho} + i) - f^{\rho}(\Sigma^{\rho}). \tag{4.14}$$

Finally, by the optimality of Σ^{ρ} under initial probability structure ρ, we obtain:

$$f^{\rho}(\Sigma^{\rho} + i) - f^{\rho}(\Sigma^{\rho}) \leq c(n+1) - c(n). \tag{4.15}$$

Putting together (4.13), (4.14), and (4.15), we have that i is not added to $\Sigma^{\varrho *}$ by the MIA in any iteration $r > n$.

We now have that i is not included in any iteration of the step-by-step building of $\Sigma^{\varrho *}$ by the MIA. Hence, we have a contradiction. $\square$

4.5 Alternative Formulation

4.5.1 Longest Path Problem

The problem that the student faces can be restated from a different point of view, as a *longest path problem.* We use Figure 4.1 to represent the decision problem. In this figure, the student starts the evaluation from the grey node (i.e., a dummy node $N+1$ with $u_{N+1} = 0, \rho_{N+1} = \rho_{N+1}^{N+1} = 1$). We describe horizontal points in the figure — $N+1, N, N-1, ..., 1$ — as schools and vertical points — $n = 0, 1, 2, ..., N$ — as the number of options that the student, at a stage, has chosen to signal. At every stage the student has the option to signal the next alternative (represented by a solid arrow) or not (represented by a dashed arrow). We use the following notation to refer to nodes on this network. Let (i, n) be the node depicting horizontal point i and vertical point n, i.e., school i and n signals. That is, at node (i, n), the student has previously chosen to signal n of the schools and must decide whether to signal school $i+1$.

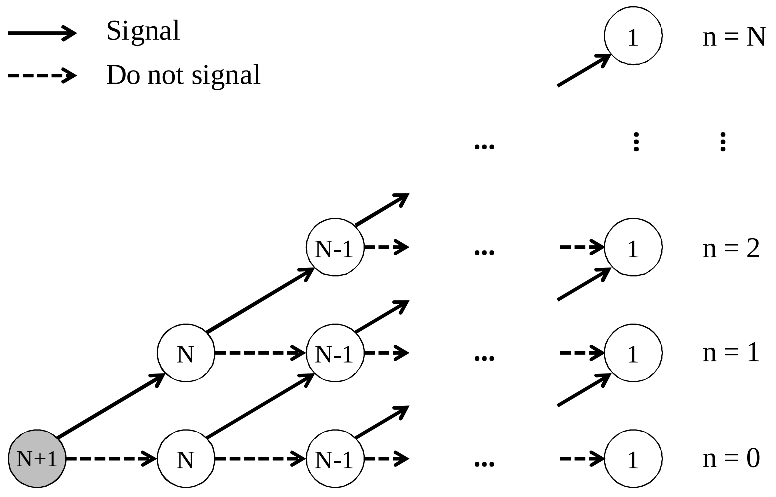


Figure 4.1: Network representation of the signaling problem

In Figure 4.1, any solution S to our original problem, (4.1), can be represented as a directed path from the original node $(N+1,0)$ to a terminal node $(1,|S|)$. We let $l(i,S)$ represent the length of the path from $(N+1,0)$ to $(i,|S|)$. We define $l(i,S)$ as

$$l(i,S) = \sum_{j=i}^{N} \rho^S_{(j,N]}(1-\rho^S_j)u_j.$$

We define the longest distance from node $(N+1,0)$ to node (i,n) as $d(i,n) = \max_{S:|S|=n} l(i,S)$.

Next, we characterize our original problem as maximizing $d(1,n) - c(n)$ over n, which is finding the longest path from node $(N+1,0)$ to node $(1,n)$ for some n. This longest path problem is not defined in the traditional way, in which the marginal utility of signaling a school (i,n) is independent of the path and the objective function is additive. Therefore, we name this problem the *nonadditive longest path problem* (NLPA).

The signaling problem in (4.1) has some properties which allow a more efficient approach to solve it. First, the objective function is downward recursive, hence Lemma 17 indicates that we can benefit from *dynamic programming* as the optimal selection among lower ranked options stays the same and we can benefit from considering options by decreasing values of their indices (i.e., increasing payoffs). Normally, with option specific costs the state space can be large. However, the cardinality dependent cost structure reduces the *state space* to $\{0, \ldots, N\}$ in general and if there is a budget of n to $\{0, \ldots, n\}$. Similarly, the order of schools indicates an acyclic graph and the assumption that we are given a selected set of options (i.e., applied schools) reduces our *stages* to $\{1, \ldots, N\}$. The signaling problem, so NLPA, can be solved by the dynamic programming algorithm described below in Algorithm 4.2, which has a better time complexity than the MIA as we will demonstrate.

Algorithm 4.2 NLPA: Nonadditive Longest Path Algorithm

```
let $d(N+1,0) = 0$,
$n = 0$ {signal no options}
for $i = N$ down to 1 do
    $d(i,n) = (1-\rho_i)u_i + \rho_i d(i+1,n) \Rightarrow p(i,n) = (i+1,n)$
end for
repeat
    $n = n+1$
    $i = N+1-n$ {signal all $n$ options in $\{i, \ldots, N\}$}
    $d(i,n) = (1-\rho_i^i)u_i + \rho_i^i d(i+1,n-1) \Rightarrow p(i,n) = (i+1,n-1)$
    for $i = N-n$ down to 1 do
        $d(i,n) = \max \begin{cases} (1-\rho_i^i)u_i + \rho_i^i d(i+1,n-1), & \Rightarrow p(i,n) = (i+1,n-1) \\ (1-\rho_i)u_i + \rho_i d(i+1,n), & \Rightarrow p(i,n) = (i+1,n) \end{cases}$
    end for
until $d(1,n) - d(1,n-1) < c(n) + c(n-1)$
return optimal number of options to signal $n^* = n-1$, and parents $p$
```

Note that the first `for` loop calculates the "signal no options" case, which does not require any comparison. Similarly, calculations about nodes $(N+1-n, n)$ can be performed without comparison because at most n schools can be signaled among n schools, hence all are signaled. The second and third assignment statements in the `repeat...until` loop performs this. The algorithm yields a longest path tree rooted at $(N+1, 0)$, hence each node (i, n) will have a parent node $p(i, n)$ depending on the decision to signal or not to signal option i. Then the optimal solution Y_{n^*} can be retrieved by Algorithm 4.3. Algorithms 4.2 and 4.3 returns the optimal solution as the following theorem states.

Theorem 24. *$d(i, n)$ is the maximal distance between nodes $(N+1, 0)$ and (i, n).*

Algorithm 4.3 NLPA: Retrieve optimal set

```
let $i = 1$, $Y_{n^*} = \emptyset$ and $k = n^*$
while $k > 0$ do
  if $p(i,k) = (i+1, k-1)$ then
    $Y_{n^*} = Y_{n^*} + i$
    $k = k - 1$
  end if
  $i = i + 1$
end while
return optimal set $Y_{n^*}$
```

Proof. By induction. It holds trivially for $(i, 0)$ and $(i, N+1-i)$ as these correspond to "no option signaled" and "all options signaled" cases, respectively. Assume it holds for $(i+1, n)$ and $(i+1, n-1)$, we claim that $d(i,n) \geq l(i,S)$ for each $S \sqsubseteq i$ such that $|S| = n$. For a contradiction, assume such an S exists but $d(i,n) < l(i,S)$. If $i \in S$, we have $\rho_i^S = \rho_i^i$, hence:

$$\begin{aligned} l(i,s) &= (1-\rho_i^i)u_i + \rho_i^i l(i+1, S-i), \\ &\leq (1-\rho_i^i)u_i + \rho_i^S d(i+1, n-1) \leq d(i,n). \end{aligned}$$

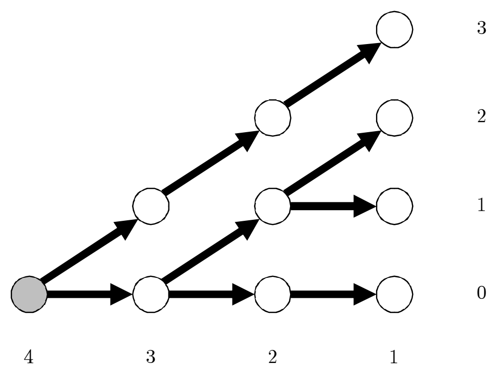
Otherwise, if $i \notin S$, we have $S - i = S$ and $\rho_i^S = \rho_i$, therefore:

$$\begin{aligned} l(i,s) &= (1-\rho_i)u_i + \rho_i l(i+1, S-i), \\ &\leq (1-\rho_i)u_i + \rho_i d(i+1, n) \leq d(i,n). \end{aligned}$$

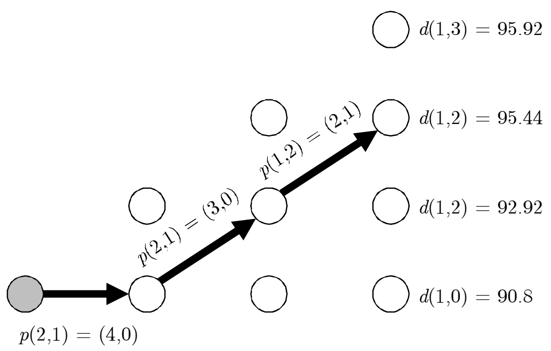
This is a contradiction as we assumed $d(i,n) < l(i,S)$. Therefore, $d(i,n)$ equals the length of the longest path to (i,n). □

Finally, letting $i = 1$, we obtain $d(i,n) = f(\Sigma_n)$ for each n. The equivalence between a solution S to the original problem and a path from $(N+1, 0)$ to $(1, n)$ implies that NLPA solves the problem optimally. The following example shows a sample run of the algorithm.

Consider a case with $N = 3$. Let $u = [100, 90, 80]$, $\rho = [0.4, 0.5, 0.2]$, $\rho^N = [0.3, 0.2, 0.1]$ and $c(n) = n$. For $n = 0$ the calculations are trivial and we obtain $d(1,0) = f(0) = 90.8$, when



(a) Longest paths for each n



(b) Optimal path

Figure 4.2: NLPA example

no options are signaled. For $n = 1$, $d(3,1) = 72$ with $p(3,1) = (4,0)$, $d(2,1) = 84.8$ with $p(2,1) = (3,0)$ and $d(1,1) = 93.92$ with $p(1,1) = (2,1)$. Since $d(1,1) - d(1,0) = 3.12 > 1$, we continue. For $n = 2$, $d(2,2) = 86.4$ with $p(2,2) = (3,1)$ and $d(1,2) = 95.44$ with $p(1,2) = (2,1)$. Now, $d(1,2) - d(1,1) = 1.52 > 1$, we continue. Finally, for $n = 3$, we have $d(1,3) = 95.92$ with $p(1,3) = (2,2)$. As $d(1,3) - d(1,2) = 0.48 < 1$, we stop, returning $n^* = 2$. The resulting solution graph is illustrated in Figure 4.2.

The optimal set is retrieved as follows. We start from node $(1,2)$ with $Y_2 = \emptyset$. The parent of that node is $p(1,2) = (2,1)$, hence $Y_2 = \{1\}$. Now, at node $(2,1)$, we check $p(2,1) = (3,0)$, hence $Y_2 = \{1,2\}$. Then, we stop. The optimal set is $\{1,2\}$.

In Theorem 25, we compare the computational complexities of the two algorithms.

Theorem 25. *MIA runs in $O(N^3)$ time and requires $O(N)$ memory. NLPA runs in $O(N^2)$ time and requires $O(N)$ memory.*

Proof. MIA runs in $O(N)$ steps. At each step n, it makes $O(N)$ comparisons in order to find $\arg\max_{i \in N} f(Y_{n-1} + i)$. Each of these comparisons requires calculating f. It takes $O(N)$ operations to calculate $f(S)$ for any $S \subseteq N$. Therefore, MIA runs in $O(N^3)$ time.

MIA requires $O(N)$ memory. It requires $O(1)$ memory for calculating f, which can be done sequentially starting from the lowest ranked options. And it requires $O(N)$ to store the optimal set.

NLPA, on the other hand, runs in $O(N)$ steps. At each step n, it makes $O(N)$ comparisons, but each comparison requires $O(1)$ operations. The advantage of NLPA over MIA is that, in each step n, NLPA calculates $f(S)$ only once, which is for $S = \Sigma_n$. Retrieving the solution requires $O(N)$ time. Hence, NLPA runs in $O(N^2)$ time.

As for memory, Algorithm 4.2 calculates longest path lengths d and requires parents p to be stored, hence it requires $O(N^2)$ memory. However, we can modify the algorithm to obtain a complexity as stated in the theorem. First, we see that once level n calculations are made, level $n-1$ becomes obsolete, hence we do not need to store the entire grid of values, but only the current and the previous. Therefore, the longest path length calculations can be performed using a $2 \times N$ vector.

Moreover, once an upward arc pointing to i is selected in iteration n, only upward arcs pointing i are selected in the subsequent steps. This is because once an option is in the optimal set, it stays in the optimal set; and an upward arc pointing i at iteration n implies $i \in \Sigma_n([1,i])$. Therefore, instead of storing the parents p, we can store the first iteration in which an upward arc pointing to i is selected. Let v denote this vector and note that $v_i \leq N - i + 1$ for all i because there are $N - i$ options that are lower ranked than i. Then, we can still retrieve the optimal solution in $O(N)$ time using Algorithm 4.4. Therefore, NLPA can be altered to require $O(N)$ memory. □

Algorithm 4.4 NLPA: Retrieve optimal set using v

```
let $i = 1$, $Y_{n^*} = \emptyset$ and $k = n^*$
while $k > 0$ do
  if $v_i \leq k$ then
    $Y_{n^*} = Y_{n^*} + i$
    $k = k - 1$
  end if
  $i = i + 1$
end while
return optimal set $Y_{n^*}$
```

4.5.2 Reformulation

Now we describe a set of linear constraints that imitates the NLPA algorithm, hence yields the optimal result for the signaling problem. This idea is similar to the heuristic reformulations of the follower problems in Chapters 2 and 3. The key difference is that, for the signaling problem, the applicant's problem can be solved "optimally". Although we do not consider the game version of this problem in this chapter, we present an example of the reformulation of the applicant's problem using ordinary linear constraints below. We use the following notation.

Variables:

d_{in} : Longest distance from node $(N+1,0)$ to node (i,n)

v: Net payoff, that is, longest path length minus signaling cost

x^s_{in} : Flow from node (i,n) to option $i+1$ when i is signaled $(s=1)$ or not $(s=0)$

y^s_{in} : Given that n options are signaled in $[i,N]$, if option i is signaled (i.e., arc connecting $(i+1,n-1)$ to (i,n) is on the longest path), $y^1_{in}=1$, else (i.e., arc connecting $(i+1,n)$ to (i,n) is on the longest path) $y^0_{in}=1$.

The constraints resemble the idea we used in Chapter 3 for the competitive prize-collecting Steiner tree problem. We first define the flow variable d that originates from node $(N+1,0)$ and goes to to every node until option 1. Then, we calculate the expected utilities at option 1, for each n. Finally, we define a flow x from option 1 to $N+1$, which yields the optimal decisions y.

The first flow $d_{i,n}$ represents the potential longest distance to each node (i,n). It is calculated using constraint (4.16), similar to Algorithm 4.2. The main difference is that, in constraints we make the calculation for each (i,n) pair unlike the algorithm, in which we iterate over n until finding the optimal one. Given that n options are signaled until (including) option i, there could be two possibilities: i is signaled (first inequality) or not (second inequality). Each node j supplies either $(1-\rho^j_j)u_j$ or $(1-\rho_j)u_j$ depending on the decision. Since we do not consider the decision yet, we name the variable "potential longest distance". In the optimal solution, the d_{in} that correspond to the longest path will have the longest distances.

$$d_{in} \geq \begin{cases} \rho_i^i d_{i+1,n-1} + (1-\rho_i^i)u_i & \forall i \in N,\, n \in \{1,\ldots,N+1-i\} \\ \rho_i d_{i+1,n} + (1-\rho_i)u_i & \forall i \in N,\, n \in \{0,\ldots,N-i\} \end{cases} \tag{4.16}$$

Once we arrive at option 1, we calculate the $d_{1,n}$ using (4.16), and then calculate the optimal net payoff $v = \max_n\{v(\Sigma_n)\}$ for each n using constraint (4.17) below.

$$v \geq d_{1,n} - c_n \quad \forall n \in \{0,\ldots,N\} \tag{4.17}$$

Next, we calculate the reverse flow x. We initiate the flow as in constraint (4.18). Now, the individual contributions of the options are associated with binary decision variables y. The payoff from the lower ranked options are transferred once the individual contribution is deducted as in constraint (4.19).

$$v + \sum_{n=0}^{N} c_n(y_{1,n}^0 + y_{1,n}^1) = \sum_{n=0}^{N} \left[(1-\rho_1)u_1 y_{1,n}^0 + (1-\rho_1^1)u_1 y_{1,n}^1 + \rho_1 x_{1,n}^0 + \rho_1^1 x_{1,n}^1\right]$$
$$\forall n \in \{0,\ldots,N\} \tag{4.18}$$

$$x_{i-1,n}^0 + x_{i-1,n+1}^1 = (1-\rho_i)u_i y_{i,n}^0 + (1-\rho_i^i)u_i y_{i,n}^1 + \rho_i x_{i,n}^0 + \rho_i^i x_{i,n}^1$$
$$\forall i \in \{2,\ldots,N+1\},\, n \in \{0,\ldots,N+1-i\} \tag{4.19}$$

Note that certain x and y that do not correspond to arcs in Figure 4.1 are equal to 0 by definition. Furthermore, in order to obtain a tree in the end, we add the following constraints. (4.20) states that there is an arc entering node (i,n) in the optimal solution if and only if there is an arc leaving it. (4.21) states that only one arc is selected at first, which combined with (4.20) ensures that the resulting structure is a tree.

$$y^0_{i,n} + y^1_{i,n} = y^0_{i-1,n} + y^1_{i-1,n+1} \quad \forall i \in \{1, \dots, N-1\},\, n \in \{0, \dots, N+1-i\} \tag{4.20}$$

$$1 = y^0_{N,0} + y^1_{N,1} \tag{4.21}$$

Finally, d, x and v are nonnegative and y is binary. It is not difficult to see that these constraints result in the optimal solution as the following theorem states.

Theorem 26. *Constraints* (4.16)-(4.21) *constitute a sufficient optimality condition for the signaling problem and return optimal solution y and optimal net payoff v.*

Proof. Constraints (4.16)-(4.17) ensure that the calculated longest distances are greater than or equal to the actual longest distances, hence v is greater than or equal to optimal net payoff. Now, let us look at constraint (4.19). At node $(N+1, 0)$, we have $x^0_{N,0} + x^1_{N,1} = 0$. Constraint (4.21) ensures that only one arc (connecting $(N+1,0)$ to $(N,0)$ or $(N,1)$) is selected, hence not only the flow through the unselected arc, but also the flow x into the unvisited node, becomes zero. That is,

$$y^0_{N,0} = 0 \Rightarrow x^0_{N-1,0} + x^1_{N-1,1} = 0$$
$$y^1_{N,1} = 0 \Rightarrow x^0_{N-1,1} + x^1_{N-1,2} = 0$$

Consequently, no flow passes through any x that corresponds to an unselected arc. Therefore, since constraints (4.18)-(4.19) are equalities, and the flow x only includes contributions from selected arcs, the calculated net payoff v in (4.18) is equal to the net payoff of the selected path. Since v is already greater than or equal to the net payoff of the maximum net payoff, it is equal to the optimal (maximum) net payoff and the associated y is the optimal solution to the signaling problem. □

We do not further investigate and refine these constraints and spare it for future studies.

4.6 Conclusions

We demonstrated that a greedy algorithm implements the optimal set for a simultaneous signaling problem, a problem that applies most directly to signaling college admissions and job search processes. In characterizing the optimal algorithm, we extended the analysis of Chade and Smith (2006) to an environment in which non-inclusion in a choice set results possibly in a non-zero probability of acceptance — either university admission or a job offer.

Moreover, we showed that the problem can be restated as a longest path problem, for which we proposed a dynamic programming algorithm, which is more efficient than the initial greedy algorithm we studied. Finally, using a similar idea as in Chapter 3, we derived a set of constraints based on a flow argument, which constitute a sufficient optimality condition for the signaling problem.

Our study is part of two larger research issues. The first is the determination of the optimal algorithm for a simultaneous search and signaling problem. As an example of this problem, a high school senior must decide not only the set of schools to which he or she applies, but also the subset (of those to which he or she sends applications) of schools to which the applicant sends a signal. Our preliminary results on this problem indicate that a straightforward greedy algorithm is not optimal. However, the problem can again be posed as a longest path problem and NLPA could solve it. Again using the insight from NLPA, we believe that MIA can be modified to solve this problem, too.

The second problem is the determination of the acceptance probabilities (with and without signals) in a college or job matching problem. Dearden et al. (2011) examine, both theoretically and empirically, whether signaling has a heterogeneous effect in a population of applicants. In particular, this model examines the determination of equilibrium acceptance probabilities as a function of applicant SAT scores. Their analysis demonstrates that the positive effect of a signal on acceptance probabilities is increasing in SAT scores. The results of their theoretical model are somewhat restrictive, however, because the model has only two selective schools and furthermore each applicant can send only one signal. The examination of a general model with N schools and a general signaling cost function could yield interesting results.

Chapter 5

Conclusions and Future Research

Most classical models for network design and operations research, in general, assume a single decision maker with a single objective. This assumption fails to represent problems adequately when there are multiple decision makers or a decision maker has multiple goals. In this research, we dealt with an extension of the former type and incorporated hierarchical competition in three network design problems. Incorporating competition, however, brings a trade-off between higher representation power and increased solution effort.

We introduced a heuristic reformulation idea in order to solve these problems near-optimally within reasonably short solution times. The essence of the idea was to find or derive a simple algorithm for the problem, represent it as a feasibility problem using linear constraints and embed them into the leader's problem in order to obtain a single-level MIP. We demonstrated three implementations on network design problems — maximal covering, Steiner tree and longest path.

We first studied the competitive maximal covering location problem. In this problem, two firms enter the market —locate their facilities— sequentially, each with the objective of maximizing the customer demand it captures in the end. The problem that the leader faces is significantly more difficult to solve, because the leader has to anticipate the best response of the follower, which, alone, is the solution of an NP-hard optimization problem. We overcame this difficulty by using a greedy algorithm as a proxy for the follower's decision. Not only is the computational performance very reasonable, but the solution quality is also very high. Even if the follower actually responds optimally, the method still provides an effective heuristic for the leader's problem.

Our model assumes that both firms have perfect information about the customer demand. If the leader is the first entrant to a market that is not yet mature and there is a time gap until the follower enters, it is reasonable to think that the customer demand can change between the two actions. We can introduce this demand uncertainty into our model, where the leader does not know but has a probabilistic belief about what the customer demands will be in the long run. On the other hand, the follower is assumed to observe the long run demand and enter the somewhat matured market. This uncertainty results in a Stackelberg game with imperfect and asymmetric information, where asymmetry favors the follower, who perfectly knows the state of demands. We can model the situation using a two-stage stochastic programming approach, where under each scenario the follower responds implementing a greedy response and the leader maximizes expected capture.

Our second study models a competitive distribution system design problem. We model firms that compete by building their individual distribution networks to serve customers. Each would like to maximize its network profit, which is the net of revenue at the demand nodes that are served and investment costs of the arcs that are built. Consequently, the decisions of each firm affect the revenue structure of the network. This corresponds to a competitive version of the prize-collecting Steiner tree problem. We consider both simultaneous-move and sequential games. We characterize an equilibrium solution for the former setting using a MIP model to determine the players' optimal solutions. For the sequential game, due to the same computational difficulty as before, we replace the follower's optimal response with the strong pruning heuristic. The heuristic relies on finding the best induced subtree from a population of trees, and subsequent pruning of branches given the leader's solution. The result is a tractable model that offers short solution times to produce near-optimal design strategies.

This method can easily be implemented for other bilevel problems whose second-level problem can be represented as a prize-collecting Steiner tree problem. An example is highway drug interdiction. In this problem we can consider the follower to be a criminal organization smuggling drugs into the country using roadways and the leader as a law enforcement institution who has a limited number of patrol units to deploy along these roads. Each patrol unit deployed increases the cost of using that particular road for the trafficker. Now instead of sharing nodal revenue, we model the increase in cost (in discrete steps) as units are deployed. Since we restrict the follower to a collection of trees, there is no significant distinction between selecting a node or selecting an arc. Therefore, we believe that the method can be applied to this problem with minimal additional effort.

Finally, in our last study, we investigated demonstrated interest in the job and university application process. We revisited the college admissions problem in a sequential game with imperfect information. In this game, there is a market formed by candidates (e.g., students, job applicants) and institutions (e.g., schools, companies). Each candidate applies to a set of institutions and subsequently demonstrates further interest in order to increase his or her acceptance probability by signaling a subset of these. In this dissertation, we study the applicants' side of this game and have developed an algorithm that characterizes the candidates' optimal choices. We also formulate the problem as a non-additive longest-path problem and introduce a dynamic programming algorithm, for which we showed an initial formulation using linear constraints. The appeal of this formulation was that, as the algorithm solved the problem optimally, this was an exact reformulation of the second-level problem.

The next step in this research is to establish the school's problem and analyze the resulting game. We would like to derive results on the selectivity of the schools and evaluate admission policies. The proposed algorithm can be considered in multiple dimensions. If we consider a budget for both the selection and signaling decisions, we can model the problem using a similar network structure. Now, there would be three decisions at every node: “do not select”, “select”, “select and signal”.

In conclusion, we believe our study opens new research directions in bilevel programming. There are several other application areas including traffic management, network interdiction and electricity demand response management, to name a few. In the near future, we would like to explore applications and extensions of these ideas and develop specific solution methods for the resulting models.

Bibliography

S. Albers, S. Eilts, E. Even-Dar, Y. Mansour, and L. Roditty. On nash equilibria for a network creation game. In *In Proc. of SODA*, pages 89–98, 2006.

M. A. Amouzegar and K. Moshirvaziri. Determining optimal pollution control policies: An application of bilevel programming. *European Journal of Operational Research*, 119(1):100–120, 1999.

S. P. Anderson, A. de Palma, and J.-F. Thisse. *Discrete Choice Theory of Product Differentiation*, chapter 8, pages 273–342. MIT Press, 1992.

E. Anshelevich, A. Dasgupta, E. Tardos, and T. Wexler. Near-optimal network design with selfish agents. In *STOC*, pages 511–520, 2003.

E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, and T. Wexler, T. Roughgarden. The price of stability for network design with fair cost allocation. In *FOCS'04*, pages 295–304, 2004.

S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

A. Aumann and R. Myerson. Endogenous formation of links between players and coalitions: An application of the shapley value. In A. Roth, editor, *The Shapley Value*, pages 175–191. Cambridge University Press, 1988.

C. Avery and J. D. Levin. Early admissions in selective colleges. *American Economic Review*, 100 (5):2125–2156, 2010.

J. F. Bard. Coordination of a multidivisional organization through two levels of management. *Omega*, 11(5):457–468, 1983.

J. F. Bard. *Practical Bilevel Optimization: Algorithms and Applications*. Kluwer Academic Publishers, 1998.

J. F. Bard. Bilevel programming in management. In C. A. Floudas and P. M. Pardalos, editors, *Encyclopedia of Optimization*, pages 269–274. Springer US, 2009.

J. F. Bard and J. T. Moore. An algorithm for the discrete bilevel programming problem. *Naval Research Logistics*, 39:419–435, 1992.

J. F. Bard, J. Plummer, and J. C. Sourie. A bilevel programming approach to determining tax credits for biofuel production. *European Journal of Operational Research*, 120:30–46, 2000.

O. Ben-Ayed and C. E. Blair. Computational difficulties of bilevel linear programming. *Operations Research*, 38(3):556–560, 1990.

O. Ben-Ayed, C. Blair, D. Boyce, and L. LeBlanc. Construction of a real-world bilevel linear programming model of the highway network design problem. *Annals of Operations Research*, 34(1):219–254, 1992.

D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. Williamson. A note on the prize collecting traveling salesman problem. *Mathematical Programming*, 59(1):413–420, 1993.

J. Bracken and J. T. McGill. Mathematical programs with optimization problems in the constraints. *Operations Research*, 21(1):37–44, 1973.

U. Brandes, M. Hoefer, and B. Nick. Network creation games with disconnected equilibria. In P. C. and Z. S., editors, *LNCS*, pages 394–401. Springer-Verlag, 2008.

A. P. Burgard, P. Pharkya, and C. D. Maranas. Optknock: A bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. *Biotechnology and Bioengineering*, 84:647–657, 2003.

W. Candler and R. Norton. Multilevel programming and development policy. Staff Working Paper SWP258, World Bank, 1977.

H. Chade and L. Smith. Simultaneous search. *Econometrica*, 74(5):1293–1307, 2006.

H. Chen, T. Roughgarden, and G. Valiant. Designing networks with good equilibria. In *SODA'08*, pages 854–863. SIAM, 2008.

G. Christodoulou, E. Koutsoupias, and A. Nanavati. Coordination mechanisms. *Theoretical Computer Science*, 410(36):3327–3336, 2009.

R. Church and C. S. ReVelle. The maximal covering location problem. *Papers in Regional Science*, 32(1):101–118, 1974.

D. Cieslik. *Steiner minimal trees*. Kluwer Academic Publishers, 1998.

P. A. Clark and A. W. Westerberg. Bilevel programming for steady-state chemical process designi. fundamentals and algorithms. *Computers & Chemical Engineering*, 14(1):87–97, 1990.

P. Coles, J. Cawley, P. B. Levine, M. Niederle, A. E. Roth, and J. J. Siegfried. The job market for new economists: A market design perspective. *Journal of Economic Perspectives*, 24(4): 187–206, 2010a.

P. Coles, A. Kushnir, and M. Niederle. Preference signaling in matching markets. Working Paper 16185, National Bureau of Economic Research, 2010b.

B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153:235–256, 2007.

G. Cornuejols, M. L. Fisher, and G. L. Nemhauser. Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management Science*, 23(8):789–810, 1977.

J.-P. Cote, P. Marcotte, and G. Savard. A bilevel modelling approach to pricing and fare optimisation in the airline industry. *Journal of Revenue and Pricing Management*, 2(1):23–36, 2003.

M. S. Daskin. *Network and Discrete Location: Models, Algorithms, and Applications*. John Wiley and Sons, New York, NY, 1995.

C. d'Aspremont, J. J. Gabszewicz, and J.-F. Thisse. On hotelling's "stability in competition". *Econometrica*, 47(5):1145–1150, 1979.

J. Dearden, S. Li, and C. Meyerhoefer. Demonstrated interest: Signaling behavior in college admissions. Working paper, Lehigh University, 2011.

S. Dempe. *Foundations of Bilevel Programming*. Kluwer Academic Publishers, 2002.

S. Dempe. Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints. *Optimization*, 52(3):333–359, 2003.

S. Dempe, V. Kalashnikov, and R. Z. Rios-Mercado. Discrete bilevel programming: Application to a natural gas cash-out problem. *European Journal of Operational Research*, 166(2):469–488, 2005.

S. T. DeNegre and T. K. Ralphs. A branch-and-cut algorithm for integer bilevel linear programs. In J. C. et al., editor, *Operations Research and Cyber-Infrastructure*, pages 65–78. Springer, 2009.

Z. Drezner, editor. *Facility Location: A Survey of Applications and Methods*. Springer-Verlag, New York, NY, 1995.

Z. Drezner and H. W. Hamacher, editors. *Facility Location: Applications and Theory*. Springer-Verlag, New York, NY, 2002.

T. A. Edmunds and J. F. Bard. An algorithm for the mixed-integer nonlinear bilevel programming problem. *Annals of Operations Research*, 34:149–162, 1992.

H. A. Eiselt. Equilibria in competitive location models. In H. A. Eiselt and V. Marianov, editors, *Foundations of Location Analysis*, chapter 7, pages 139–162. Springer US, 2011.

H. A. Eiselt and G. Laporte. Sequential location problems. *European Journal of Operational Research*, 96(2):217–231, 1996.

H. A. Eiselt, G. Laporte, and J.-F. Thisse. Competitive location models: A framework and bibliography. *Transportation Science*, 27(1):44–54, 1993.

S. Engevall, M. Göthe-Lundgren, and P. Värbrand. A strong lower bound for the node weighted steiner tree problem. *Networks*, 31(1):11–17, 1998.

A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. On a network creation game. In *PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 347–351, 2003.

N. P. Faisca, V. Dua, B. Rustem, S. P. M., and E. N. Pistikopoulos. Parametric global optimisation for bilevel programming. *Journal of Global Optimization*, 38:609–623, 2007.

P. Feofiloff, C. G. Fernandes, C. E. Ferreira, and J. C. de Pina. Primal-dual approximation algorithms for the prize-collecting steiner tree problem. *Information Processing Letters*, 103(5):195–202, 2007.

M. Fischetti. Facets of two steiner arborescence polyhedra. *Mathematical Programming*, 51(1): 401–419, 1991.

C. A. Floudas and P. M. Pardalos, editors. *Encylopedia of Optimization*. Springer US, 2009.

J. J. Gabszewicz and J.-F. Thisse. Location. In R. Aumann and S. Hart, editors, *Handbook of Game Theory with Economic Applications*, volume 1, chapter 9, pages 281–304. Elsevier, 1992.

M. X. Goemans. The steiner tree polytope and related polyhedra. *Mathematical Programming*, 63 (1):157–182, 1994.

M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24:296–317, 1995.

L. Gourvés and J. Monnot. Three selfish spanning tree games. In P. C. and Z. S., editors, *LNCS*, pages 465–476. Springer-Verlag, 2008.

Z. H. Gumus and C. A. Floudas. Global optimization of mixed-integer bilevel programming problems. *Computational Management Science*, 2:181–212, 2005.

S. L. Hakimi. On locating new facilities in a competitive environment. *European Journal of Operational Research*, 12(1):29–35, 1983.

Y. Halevi and Y. Mansour. A network creation game with nonuniform interests. In *Lecture Notes in Computer Science: Internet and Network Economics*, pages 287–292. Springer-Verlag, 2007.

M. Haouari, S. Layeb, and H. Sherali. The prize collecting steiner tree problem: models and lagrangian dual optimization approaches. *Computational Optimization and Applications*, 40(1): 13–39, 2008.

B. F. Hobbs and S. K. Nelson. A nonlinear bilevel model for analysis of electric utility demand-side planning issues. *Annals of Operations Research*, 34(1):255–274, 1992.

D. S. Hochbaum. Approximating covering and packing problems: Set cover, vertex cover, independent set, and related problems. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 94–143. PWS Publishing Company, 1997.

M. Hoefer. Noncooperative tree creation. In R. Kralovic and P. Urzyczyn, editors, *LNCS*, pages 517–527. Springer-Verlag, 2006.

E. Hoover. The dynamics of demonstrated interest. *The Chronicle of Higher Education*, May 25 2010.

H. Hotelling. Stability in competition. *The Economic Journal*, 39(153):41–57, 1929.

F. K. Hwang, D. S. Richards, and P. Winter, editors. *The steiner tree problem*, volume 53 of *Annals of Discrete Mathematics*. Elsevier, 1992.

E. Israeli and R. K. Wood. Shortest-path network interdiction. *Networks*, 40(2):97–111, 2002.

M. O. Jackson. A survey of models of network formation: Stability and efficiency. In G. Demange and M. Wooders, editors, *Group formation in economics: Networks, clubs, and coalitions*. Cambridge University Press, 2003.

R. H. Jan and M. S. Chern. Nonlinear integer bilevel programming. *European Journal of Operations Research*, 72:574–587, 1994.

R. G. Jeroslow. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical Programming*, 32(2):146–164, 1985.

D. S. Johnson, M. Minkoff, and S. Phillips. The prize collecting steiner tree problem: theory and practice. In S. '00, editor, *Proceedings of the eleventh annual ACM-SIAM symposium on discrete algorithms*, pages 760–769, 2000.

B. Kara and V. Verter. Designing a road network for hazardous materials transportation. *Transportation Science*, 38:188–196, 2004.

R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*. Plenum Press, 1972.

A. Klose and A. Drexl. Facility location models for distribution system design. *European Journal of Operational Research*, 162(1):4–29, 2005.

D. Kress and E. Pesch. Sequential competitive location on networks. *European Journal of Operational Research*, 217(3):483–499, 2012.

A. A. Kuehn and M. J. Hamburger. A heuristic program for locating warehouses. *Management Science*, 9(4):643–666, 1963.

A. Kushnir. Harmful signaling in matching markets. Working Paper 509, Fondazione Eni Enrico Mattei, 2010.

M. Labbe, P. Marcotte, and G. Savard. A bilevel model of taxation and its applications to optimal highway pricing. *Management Science*, 44:1595–1607, 1998.

I. Ljubić, R. Weiskircher, U. Pferschy, G. W. Klau, P. Mutzel, and M. Fischetti. An algorithmic framework for the exact solution of the prize-collecting steiner tree problem. *Mathematical Programming*, 105(2):427–449, 2006.

A. Lucena and M. G. C. Resende. Strong lower bounds for the prize collecting steiner problem in graphs. *Discrete Applied Mathematics*, 141(1-3):277–294, 2004.

N. Megiddo, E. Zemel, and S. L. Hakimi. The maximum coverage location problem. *SIAM Journal on Algebraic and Discrete Methods*, 4(2):253–261, 1983.

A. Migdalas. Bilevel programming in traffic planning: models, methods and challenge. *Journal of Global Optimization*, 7:381–405, 1995.

C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.

A. Mitsos. Global solution of nonlinear mixed-integer bilevel programs. *Journal of Global Optimization*, 47:557582, 2010.

J. T. Moore and J. F. Bard. An algorithm for the zero-one bilevel programming problem. Technical report, Department of Mechanical Engineering, University of Texas, Austin, 1987.

J. T. Moore and J. F. Bard. The mixed integer linear bilevel programming problem. *Operations Research*, 38(5):911–921, 1990.

R. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1991.

R. B. Myerson. Graphs and cooperation in games. *Mathematics of Operations Research*, 2(3): 225–229, 1977.

S. H. Owen and M. S. Daskin. Strategic facility location: A review. *European Journal of Operational Research*, 111(3):423–447, 1998.

C. Papadimitriou. Algorithms, games, and the internet. In *STOC*, pages 749–753, 2001.

C. M. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

G. Perakis and G. Roels. The price of anarchy in supply chains. *Management Science*, 53:1249–1268, 2007.

G. Piliouras and Y. Vigfusson. A delivery network creation game. Technical report, Cornell University Press, 2009.

F. Plastria. Static competitive facility location: An overview of optimisation approaches. *European Journal of Operational Research*, 129(3):461–470, 2001.

F. Plastria and L. Vanhaverbeke. Discrete models for competitive location with foresight. *Computers & Operations Research*, 35(3):683–700, 2008.

J. Puerto, A. Schobel, and S. Schwarze. The path player game: A network game from the point of view of the network providers. *Mathematical Methods of Operations Research*, 68:1–20, 2008.

C. S. ReVelle. The maximum capture or ”sphere of influence” location problem: Hotelling revisited on a network. *Journal of Regional Science*, 26(2):343–358, 1986.

C. S. ReVelle and H. A. Eiselt. Location analysis: A synthesis and survey. *European Journal of Operational Research*, 165(1):1–19, 2005.

C. S. ReVelle, H. A. Eiselt, and M. S. Daskin. A bibliography for some fundamental problem categories in discrete location science. *European Journal of Operational Research*, 184(3):817–848, 2008.

R. W. Rosenthal. A class of games possessing pure strategy nash equilibria. *International Journal of Game Theory*, pages 65–67, 1973.

G. K. Saharidis and M. G. Ierapetritou. Resolution method for mixed integer bi-level linear problems based on decomposition technique. *Journal of Global Optimization*, 44:29–51, 2008.

K. Sahin and A. R. Ciric. A dual temperature simulated annealing approach for solving bilevel programming problems. *Computers and Chemical Engineering*, 23:11–25, 1998.

M. P. Scaparra and R. L. Church. A bilevel mixed-integer program for critical infrastructure protection planning. *Computers & Operations Research*, 35(6):1905–1923, 2008.

A. Segev. The node-weighted steiner tree problem. *Networks*, 17(1):1–17, 1987.

D. Serra and C. S. ReVelle. Market capture by two competitors: The preemptive location problem. *Journal of Regional Science*, 34(4):549–561, 1994.

D. Serra and C. S. ReVelle. Competitive location in discrete space. In Z. Drezner, editor, *Facility Location: A Survey of Applications and Methods*, pages 367–386. Springer-Verlag, New York, NY, 1995.

D. Serra, S. Ratick, and C. S. ReVelle. The maximum capture problem with uncertainty. *Environment and Planning B: Planning and Design*, 23(1):49–59, 1996.

H. Simon. A behavioral model of rational choice. *The Quarterly Journal of Economics*, 69(1): 99–118, 1955.

L. V. Snyder. Facility location under uncertainty: a review. *IIE Transactions*, 38(7):547–564, 2006.

L. V. Snyder. Introduction to facility location. In J. J. Cochran, L. A. Cox, P. Keskinocak, J. P. Kharoufeh, and J. C. Smith, editors, *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley & Sons, Inc., 2010.

L. V. Snyder. Covering problems. In H. A. Eiselt and V. Marianov, editors, *Foundations of Location Analysis*, chapter 6, pages 109–135. Springer US, 2011.

G. J. Stigler. The economics of information. *Journal of Political Economy*, 69(3):213–225, 1961.

L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3:1–22, 1977.

D. Thirwani and S. R. Arora. An algorithm for the integer linear fractional bilevel programming problem. *Optimization*, 39:53–67, 1997.

L. N. Vicente and P. H. Calamai. Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, 5:291–306, 1994.

H. von Stackelberg. *Market Structure and Equilibrium*. Springer, 2011.

U. P. Wen and A. Huang. A simple tabu search method to solve the mixed-integer linear bilevel programming problem. *European Journal of Operational Research*, 88:563–571, 1996.

U. P. Wen and Y. H. Yang. Algorithms for solving the mixed integer two-level linear programming problem. *Computers & Operations Research*, 17(2):133–142, 1990.

H. Younies and H. A. Eiselt. Sequential location models. In H. A. Eiselt and V. Marianov, editors, *Foundations of Location Analysis*, chapter 8, pages 163–178. Springer US, 2011.

Vita

Tolga Han Seyhan was born in Ankara, Turkey in 1982. He received his B.S. and M.S. degrees in Industrial Engineering in 2004 and 2007, from Middle East Technical University (METU) in Turkey. His master's thesis was on developing a dynamic and spatial energy planning model for Turkey. Before joining Lehigh University, he was an Assistant Energy Expert at the Energy Market Regulatory Authority of Turkey. He joined Lehigh University in 2007 to pursue his doctoral degree. His research focuses on mathematical programming and game theory with applications to logistics, supply chain management and energy systems. He has presented his research at several international conferences. He is a member of Institute for Operations Research and the Management Sciences (INFORMS), and within INFORMS, Manufacturing and Service Operations Management (MSOM) and Transportation Science and Logistics (TSL) societies, and Section on Location Analysis (SOLA). Tolga will join Amazon.com's Inventory Planning and Control group as an Operations Research Scientist.