Network Revenue Management under Competition within Strategic Airline Alliances

Von der Mercator School of Management, Fakultät für Betriebswirtschaftslehre der

Universität Duisburg-Essen

zur Erlangung des akademischen Grades

eines Doktors der Wirtschaftswissenschaft (Dr. rer. oec.)

genehmigte Dissertation

von

Waldemar Grauberger

aus

Uljanowskoje (Russland).

Referent: Prof. Dr. Alf Kimms

Korreferent: Prof. Dr. Peter Chamoni

Tag der mündlichen Prüfung: 02.12.2015

Preface

This thesis presents results from about four years of research I conducted at the chair of logistics and operations research of Prof. Dr. Alf Kimms at the University of Duisburg-Essen. The reader of this thesis obviously only sees the results that I have gathered during this research. What he does not see, however, is the effort which had to be put into obtaining these results. While I as the author conducted the research and present its results here, this thesis was influenced by many persons without whom this work would not have turned out in the way it did. To these persons I herewith express my gratitude.

First, I would like to thank my adviser, Prof. Dr. Alf Kimms who has introduced the science of operations research to me as a university student and afterwards advised me as a doctoral student. He constantly inspired me to keep searching for solutions and to look at problems from different perspectives. I also thank Prof. Dr. Peter Chamoni for acting as a second assessor of the thesis. Further, I appreciate the support from the German Research Foundation (DFG) which supported my research financially under grant no. KI 1272/2-2.

Several more individuals at the chair of logistics and operations research were of great help and made work more pleasant. Many thanks go to Nicole Jaschinski who kept things running smoothly. Christoph Reiners also deserves a big thank for always having his door and ears open for discussions or even chats. Christina Ackermann and Igor Kozeletskyi helped with answers to several mathematical questions. I also thank Kerstin Seekircher, Marc Maiwald, and Andreas Elias for their colleagueship. Sophia Zorell was of great help concerning the search and provision of literature and Konstantin Obrubov did valuable work in the analysis and evaluation of data.

However, I would never have come so far as to write this thesis without the patience, support, and love of my family. Words cannot express the gratitude I feel for my parents who have taught me some of the most valuable lessons of life; the importance and might of education, the power of patience and endurance as well as the importance of keeping moving even in tough times. My brothers constantly encouraged my research and helped me finding answers by asking critical questions.

Last but not least I thank my three girls at home; my wife Dina and my precious daughters Luisa and Viktoria who have accompanied and supported me throughout my doctoral work, supplied me with coffee and cookies when it was necessary, and gave me a break when I needed it. Your love and support will always be remembered.

Wuppertal, April 2015 Waldemar Grauberger

Contents

Chapter 1

Introduction

The emergence of the concept of revenue management (RM) as it is known today is accounted for by a fierce competition faced by the established airlines in the USA since the late 1970's. In 1978 the U.S. Civil Aviation Board passed the Airline Deregulation Act allowing airlines to charge prices, serve routes and enter the market freely which till then was highly controlled by the board. As a consequence, new airlines with significantly lower costs and less offered services—hence called low-cost, low-fare, or no-frill airlines—entered the market charging prices up to 70% lower than the established airlines.

However, even though RM arose out of a competitive situation and can provide an advantage over a company's competitors (see e.g. Belobaba and Wilson, 1997), competition is still rarely taken into account explicitly in RM models and algorithms. After reviewing 221 RM papers published after 1999, Chiang et al. (2007) found that most publications still focus on the individual company and present monopolistic decision models. Consequently, there are still relatively few publications dealing with RM competition. Martínez-de-Albéniz and Talluri (2011, p. 1078) assumed this to be due to that fact that competitive information would render the models (even more) complicated making analysis (even more) difficult. Phillips (2007, p. 59) argued that for most pricing decisions competition can be ignored since these decisions are usually little adjustments to prices and are not even noticed or cannot be matched by the competition. Another cause to exclude competition from RM models might simply be the rejection of game theory (which is used to analyze the competitive interactions) because of the restrictive and sometimes unrealistic assumptions it makes (see e.g. Petersen, 1994). In this context Shugan (2002, p. 226) concluded that the "strong approximating assumption of no competitive response is sometimes better than the approximating assumption of pre-existing optimal behavior".

However, given the fact that virtually every company faces competition today, ignoring it in the decision making process can have quite severe consequences since this can lead to a spiral-down effect with ever lower prices as Cooper et al. (2006) showed. d'Huart and Belobaba (2012) showed through simulation studies that ignoring competition in an RM system leads to a double-counting of rejected customers—the same customer is counted at the denying airline and at the competing airline—and thus erroneously leads to higher forecasts and wrong decisions in terms of the number of seats to reserve for the higher-fare classes. Thus, being aware of competition and taking into account its actions is clearly seen as a necessary extension for future research and practice (see also e.g. Bobb and Veral, 2008, Dunleavy and Phillips, 2009, Nason, 2009 as well as Ratliff and Vinod, 2005). In this context game theory provides the necessary tools and concepts for modeling and examining competitive behavior as well as finding recommendations

and predictions for the competitors' actions which provides meaningful insights for the RM problem under competition. The most prominent game theoretic solution concept for competitive situations is the Nash equilibrium (NE). It describes a combination of competitors' actions from which no competitor has an incentive to deviate unilaterally, when the other(s) stick(s) to his/their equilibrium action(s).

Those publications that do treat competition in an RM context with game theoretic tools often restrict their investigation to only one flight leg and two fare classes. The fact that the competitors might be organized within an alliance is mostly completely ignored. However, these days many airlines are united within alliances to provide a better service to their customers, lower costs by using synergies, and raise the load factors on their planes. To the best of our knowledge until recently Netessine and Shumsky (2005) and Wright et al. (2010) were the only publications so far that took into account the possibility of alliance partners competing for customers. In addition, these models' scopes were very limited. The former only accounted for the possibility that the competitors serve different legs connected through a connecting airport and offer only two fare classes while the latter ignored the possibility of the airlines also offering identical products.

This thesis is intended to extend some existing ideas concerning competition within alliances and to help fill the research gap on RM under competition. We will model the competitors' decisions with mostly linear programs which are fairly simple and fast to solve; important aspects in RM practice. Existing models are often restricted to only one flight leg and two fare classes. Our models are able to optimize a competitor's behavior in any network and more than two fare classes. Game theory will be applied to analyze the competitive situations and predict their outcomes. By combining the disciplines of linear programming and game theory we will follow a line which has been drawn since the very beginnings of both disciplines which will become more evident later.

Secondly, we will propose algorithms to compute NE and thus to find recommendations for the competitors' decisions about the amounts of tickets to sell in different fare classes under competition. While other authors often seek to provide conditions under which an NE exists and provide algorithms that find an NE under these conditions, our algorithms are more general and do not rely on certain conditions. If an NE exists, our exact algorithm will find one with certainty. Another aspect often met in literature dealing with game theory is providing conditions under which an NE is unique. We circumvent this issue through a slight modification of our models and make sure that both competitors choose strategies that correspond to the same NE if several exist and both competitors apply the techniques presented in this thesis—if several NE exist. For the case that no NE exists or it cannot be computed within a given time limit, we propose a heuristic that finds an approximate solution. The main objective of the thesis is thus not to prove uniqueness of NE in RM games but to present models and algorithms which move along unique searching paths on their way to an NE and thus eliminate the coordination issues that arise with multiple NE. Thus, we strive at presenting algorithms for games in which uniqueness of NE is not guaranteed or cannot be proved which is the more general case in game theory. We will even provide conditions under which no pure NE exists in network RM games. In case no NE can be found, an approximation for it is computed in order to present solutions which are not worse but tendentially better than the solution which ignores competition altogether.

Finally, with our models and algorithms we take into account the fact that the competitors are organized in an alliance, a fact which has mostly been ignored so far. The reason we treat competition within a cooperative structure such as an alliance is that on the one hand the alliance

partners have different RM systems customized for their individual needs and that on the other hand complete mergers are prohibited by law in order to encourage competition. Whenever a new amendment to a model or a new algorithm is presented, we provide results from computational studies in order to evaluate the novelty and compare it with the other formulations. The networks used in the studies are of realistic sizes which means that our models and algorithms are capable of solving real-world examples.

The remainder of this thesis is structured as follows. Chapters 2, 3, and 4 lay the foundation for the rest of the thesis by introducing the essential concepts from RM, strategic airline alliances, and game theory, respectively. They introduce those concepts from which will be crucial for understanding the argumentations and discussions in the later chapters and motivate competition within alliances, an originally cooperative concept. These treatments are not only intended to explain what the described concepts deal with in general but in a first place serve the purpose of giving the reader an understanding of the issues one encounters in revenue management and game theory and which we solved with our models and procedures. As all treated topics are rich and offer tools for wide varieties of problems, we restrict ourselves to the relevant aspects playing a role in this thesis. Yet, some references covering the fields not treated here are also provided.

In Chapter 5, the existing literature applying game theory for investigating competition in an RM setting is reviewed and the basic model used for optimizing a competitor's behavior in the underlying competitive situations treated here as well as the core algorithm for solving these competitive situations are introduced. This algorithm employing the presented model is essential for the later course of the thesis. The chapter also presents a heuristic for computing an approximate NE in case no exact NE can be found (due to its non-existence or do to a limited computation time). In Chapter 6, we model the competitors' decisions within simultaneous price and quantity competition and present an algorithm to compute possibly approximate NE within this setting. Chapter 7 treats competition within an airline alliance, another essential contribution of this thesis. Both, horizontal competition, in which the competitors offer identical products, and vertical competition, where one airline might sell tickets for products occupying the partner's aircraft are considered. All these chapters provide computational studies to demonstrate the proposed algorithms' performance. Chapter 8 concludes this thesis. It summarizes the results of the foregoing chapters and provides suggestions for future research.

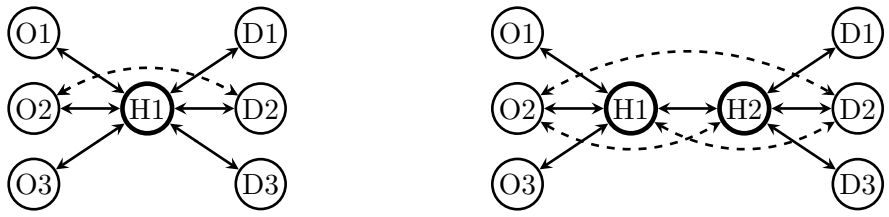
Chapter 2

Selected Topics in Revenue Management

This chapter introduces revenue management from a historical and conceptual point of view including conditions for the applicability of RM techniques. Afterwards, we will introduce the tool for controlling capacity used in the later chapters. Since RM was first used in the airline business, it is best developed in this sector and much of the terminology goes back to airlines. In this thesis RM competition between airlines builds the core topic and thus we will adopt the airline terminology as well. Examples of the notions' interpretations in other industries will be given were it is straightforward. In this context, we next define some terms which will be used throughout the thesis. See McGill and van Ryzin (1999) for a survey over publications contributing to different fields of RM.

A *leg* is a connection between an origin airport O and a destination airport D which is covered by a single, non-stop flight. It is thus the smallest fraction of a possible bundle of non-stop flights which can be sold to customers. A *resource* is a finite and fixed bundle of individual capacity units which can be utilized for serving customers. Hence, an airplane operated on a leg is a resource and a seat on this plane is a capacity unit. The complete set $L = \{1, \ldots, \mathcal{L}\}$ of legs operated by an airline is called *network*. We will assume the networks to be set up in a *hub-and-spoke* manner where not all cities can be reached directly from each other through a flight but different (spoke) airports are connected through central airports (hubs). Such hub-and-spoke networks enable the airlines to connect many cities with less planes as compared to *point-to-point* networks where all airports are connected directly with each other. Bundling demand from one spoke area to several destinations on one flight connection creates higher load factors and raises the profitability of operating a flight from and to rather rural areas. See also Goodwin and Cook (2008) for an economic and operative comparison between hub-and-spoke networks and point-to-point networks. Figure 2.1 shows networks with one and two hubs. There, the solid arcs stand for the legs that are actually operated using an airplane while the dashed arcs stand for connecting flight which necessitate a stopover in a hub (indicated by the bold nodes). I.e. a trip from O2 to D2 in the network of Figure 2.1a) needs a stopover in H1, while a trip from O2 to D2 in Figure 2.1b) has two stopovers, one in each hub. For the sake of visibility we do not show all connecting flights.

The mentioned trips (e.g. from O2 to D2) are called *Origin & Destination (O&D) pairs* or, equivalently *itineraries* and are made up of a (combination of) single-leg flight(s). The set of itineraries is $P = \{1, \ldots, \mathcal{P}\}$. An itinerary is usually sold at different prices, called *fares* which



a) Network with 12 Legs b) Network with 14 Legs

Figure 2.1: Simple Network Structures

are assigned to a set F of different *fare classes* (indexed $1, \ldots, \mathcal{F}$), depending on the time of purchase, length of stay, or restrictions on refunds. A combination of an O&D pair and a fare is called *product*. In other words, a product in the RM sense is a combination of a certain well-defined service (a flight from some origin to some destination in our case) and a price. An incidence matrix M_{lp} shows how many capacity units are needed on a leg l to serve an itinerary p (all classes of which travel with the same legs, so no index f is needed in the matrix).

We will describe the origin and development of RM and provide the conditions for its applicability in the next two Sections. Afterwards, Section 2.3 will explain capacity control as the core instrument of RM and introduce the model building the basis for the competitive models in the later chapters. In Section 2.4 assumptions concerning the RM tools for the further course of the thesis will be stated.

2.1 Origin and Development of Revenue Management

As mentioned in the previous chapter, the deregulation of airspace was the crucial factor for the emergence of the discipline of RM. However, even though deregulating the airline business was responsible for RM developing to the concept it is known as today, first applications of it were introduced before 1978. The first computer system for controlling bookings on flights (SABRE) was introduced in 1966 at American Airlines (Chiang et al., 2007, p. 98). This way the airline was able to process requests faster and more automated. One of the most prominent ideas for controlling capacity—a core concept of RM—in case of two classes was introduced by Littlewood (1972). This rule for limiting the seats to sell for the lower-fare class is still used today; directly or within heuristics in case of more than two classes. American Airlines' "Super Saver fares" were introduced in 1977 to offer discounted tickets one year before the deregulation (McGill and van Ryzin, 1999, p. 234).

After the deregulation, the probably most prominent example from this time is that of American Airlines and PEOPLExpress. The latter entered the market in 1981 and was able to charge prices up to 70% lower than the former due to significantly lower costs and less services. This way the entrant on the one hand stole customers from the established airline and on the other hand drew new customers that otherwise would have taken the train, the bus or would not have traveled at all. Since American Airlines was not able to cut costs enough to compete with the new threat on a price level, the company focused on managing revenue instead of costs and came up with ideas that still compose the core of RM today. In particular, they elaborated ways to keep the prices and demand steady for their "regular customers", the business travelers, and at the same time to offer discounts to leisure travelers for some seats which otherwise probably

would have remained empty. The discounted tickets were available under restrictions which most business customers could not meet, like e.g. minimum over night stays or purchases (long) in advance. At the same time the number of tickets available for discounted purchase was limited to retain enough capacity for the high-fare customers. With the help of RM, American Airlines was able to underbid this strong competitor (and others) and only six years after entering the market, PEOPLExpress was bankrupt never making use of RM itself. See also Cross (1998, Ch. 4), Phillips (2007, Sec. 6.1), and Talluri and van Ryzin (2004b, Sec. 1.2.1).

Due to airlines being the first to implement RM successfully, the understanding and even definition of RM is often reduced to this industry (Talluri and van Ryzin, 2004b, p. 10). A consequence is that the term "yield management" is often used interchangeably with revenue management. Here, "yield" means "the revenue per passenger-mile of traffic carried by an airline" (Belobaba, 1987, p. 11). However, strictly maximizing the yield defined this way may set wrong incentives because a maximal yield can already be reached by selling one single full-fare ticket and keep the rest of the plane empty. Furthermore, due to its connection to the airline industry, using this term in other industries might be misleading (although Kimes, 2002, p. 3 motivated using the term yield in other industries by redefining it to "yield (or revenue) per available time-based inventory unit"). This is the reason Weatherford and Bodily (1992, p. 833) proposed the term of (perishable-asset) revenue management which they defined as "the optimal revenue management of perishable assets through price segmentation". In a nutshell, RM is thus used to allocate a company's scarce resources to different customer segments and to take advantage of the customers' heterogeneous willingnesses to pay by a differentiated pricing of products.

However, a general and consistent definition of RM has not yet been proposed in the literature. Further definitions of RM were provided by e.g. Klein (2001) and Kimes (2002). The former gave a general definition in which RM employs quantitative methods for managing inflexible capacity most efficiently which is only available for a limited amount of time while the latter emphasized the use of information systems and pricing strategies. Talluri and van Ryzin (2004b, p. 2) defined RM by classifying its function within a company's operations:

> "RM is concerned with [...] *demand-management* decisions and the methodology and systems required to make them. It involves managing the firm's "interface with the market" as it were—with the objective of *increasing revenues*. RM can be thought of as the complement of *supply-chain management* (SCM), which addresses the *supply decisions* and processes of a firm, with the objective (typically) of *lowering the cost* of production and delivery."

Hence, the RM discipline takes costs as given which justifies concentrating on managing (maximizing) revenue instead of the gross margin which is the typical approach for maximizing profit in a business. This approach can be accounted for by the cost structure in typical RM businesses which consists of high fixed and low (ignorable) variable costs which will be explained in more detail in the next section. See also Jones (2002) and Kimms and Klein (2005) for surveys and discussions of different definitions of revenue (or yield) management.

Soon after RM was implemented successfully by airlines, its techniques were adopted by car rental firms (see e.g. Carroll and Grimes, 1995, Geraghty and Johnson, 1997, Savin et al., 2005 as well as Steinhardt and Gönsch, 2012) and hotel companies (see e.g. Bitran and Gilbert, 1996, Goldman et al., 2002 as well as Vinod, 2004). Though the latter's stimulus was not identical to that of the airlines for elaborating RM methods (low-fare competitors entering the market), they did have similar problems and characteristics; regular high-fare customers and at the same time

some unused capacities. Some were even losing money. Also, since many car rental counters and hotels are located near or at airports, the change in the structure of airline passengers also meant a change in customer structures for these industries (Talluri and van Ryzin, 2004b, p. 531). By applying differentiated prices and discounts and controlling capacities more sophisticatedly (e.g. denying less profitable requests in favor of future more profitable ones), car rental firms and hotels were able to stay in the market and to gain a lead over their competitors. See also Cross et al. (2010) for an overview of the milestones of RM.

Today, RM is a well-established discipline in these industries and found its way into many more including, among others, cruise lines (see e.g. Hoseason, 2002), restaurants, (see e.g. Bertsimas and Shioda, 2003), manufacturing (see e.g. Spengler et al., 2006), media and broadcasting (see e.g. Kimms and Müller-Bungart, 2007), and health care (see e.g. Gupta and Wang, 2008). See also Müller-Bungart (2007, Sec. 1.3) for further references which deal with RM in different industries. Talluri and van Ryzin (2004b, Ch. 10) and Klein and Steinhardt (2008, Ch. 1.4) provided a comparison of the similarities and differences of different industries applying RM. Cleophas et al. (2011, p. 12) gave an overview of the products sold and types of customer inventory segmentations in different industries applying RM. See also Chiang et al. (2007) for a more detailed list of the industries applying RM techniques and references to corresponding publications as well as Yeoman and McMahon-Beattie (2004, 2011) and Sfodera (2006) for collections of case studies with RM applications.

Hence, one can see that what started out as a mere reaction to new competitors entering the airline market, grew to an own science including e.g. schedule design and fleet assignment in a broader sense as well as pricing, overbooking and capacity control in a narrower sense (Klein and Steinhardt, 2008, Ch. 1.3.1). Schedule design is e.g. concerned with choosing which connections to offer and when (weekday and daytime) to depart from the origins. Based on this, it is decided which types of aircraft are to serve which legs within fleet assignment. Pricing deals with setting optimal prices to different customer segments. Overbooking comes into play when the possibility to return a ticket and thus to cancel a trip exists or if customers simply do not show up at the time of departure. This instrument is used to determine the number of tickets to sell beyond capacity in the expectation that at most as many customers make use of their tickets as there is capacity after cancellations and no-shows. Finally, capacity control—the core RM instrument and the instrument applied in this thesis—is used to decide which customer requests are to be accepted and which ones are to be denied.

Depending on the objectives, the RM concepts can be used quite differently. Often, RM is used to trade off a high capacity load factor and the desire to sell tickets at the highest possible price (Kimes, 1989, p. 349). However, other objectives are also possible like e.g. maximizing total revenue, maximizing customer goodwill and/or loyalty, extending market share or maximizing capacity utilization, see also e.g. Weatherford and Bodily (1992, Sec. 3.1) and Klein and Steinhardt (2008, Ch. 1.3.2). In the non-profit sector, for instance, RM serves to maximize the public interest and revenues are generated only to such an extent that costs are covered, see Metters and Vargas (1999) as well as de Véricourt and Lobo (2009).

2.2 Conditions for the Application of Revenue Management

While the characteristics of the industries applying RM are very similar all in all, small differences about the necessary conditions for RM exist which often rely primarily on passenger airlines (see e.g. Kimes, 1989, Weatherford and Bodily, 1992, and Talluri and van Ryzin, 2004b,

Sec. 1.3.3). This led Kimms and Klein (2005) to conclude that they are too restrictive and ignore some essential aspects for a successful RM implementation in other industries. Hence, the authors provided four general basic conditions for the application of RM which are reviewed below; integration of an external factor, heterogeneous customer behavior, restricted operational flexibility of capacity, and standardized products. While the first two are exogenous properties not necessarily controllable by the firm, the last two are endogenous and inherent to the industries or created by the firm. Müller-Bungart (2007, Ch. 1.3) explained in which ways and extents these conditions apply in various industries applying RM. Klein and Steinhardt (2008, p. 9) illustrated the connection of these conditions with others provided in the literature. Chiang et al. (2007, p. 116) pointed out that RM is being considered more and more in "non-traditional" industries in which these conditions might have to be relaxed. Consequently, the benefits from RM techniques might not be as high as in the "traditional" industries of airlines, hotels, and car rental companies.

Integration of an External Factor

One of the most crucial requirements for the application of RM techniques is the integration of an external factor to initiate production of the offered good or service. Without the necessity to integrate an external factor, i.e. with the possibility of producing a good in advance and storing it, more efficient techniques from production planning can be utilized to allocate capacity (Kimms and Klein, 2005, p. 9). As the term suggests, the external factor must come from "outside" of the firm. For instance, this can be a customer requesting a ticket on an airplane or an overnight stay in a hotel. Being dependent on input "from outside" means that the good cannot be produced in advance and stored to be sold at a later point in time as can be done with mass-produced goods like e.g. electronics. The dependency on an external factor necessitates selling the good before it is produced in order to induce the external factor to be supplied by the customers in the first place (Kimms and Klein, 2005, p. 9). More precisely, Maleri and Frietzsche (2008, p. 21) pointed out that the industries with an RM application can only sell their willingness to provide a service or to produce a good. However, the conditions under which the service or good is offered like departure time or route are set by the firm offering the service or the good.

Typically, integration of an external factor is associated with the service industry (see e.g. Fitzsimmons and Fitzsimmons, 2008, p. 21) which is why RM is often connected to service firms as in e.g. Kimes (1989) and Klein (2001). However, Klein and Steinhardt (2008, Sec. 1.2.2.2) and Müller-Bungart (2007, Sec. 1.2) pointed out that an external factor is also necessary in make-to-order industries—e.g. when a custom-made machine is ordered. See also Kimms and Müller-Bungart (2003) for a comparison of characteristics of service and make-to-order industries.

Heterogeneous Customer Behavior

Another requirement for applying RM is that the customer demand is not homogeneous but varies with time and customers have different willingnesses to pay and preferences concerning the scope of service. While integrating an external factor described above makes it necessary for the firm to sell a product before its production and use, the customers have different preferences of how much in advance they purchase a product which makes up a part of the heterogeneity of customer behavior. The traditional segmentation concerning the time of purchase is based on the customer structure consisting of leisure travelers and business travelers. See also Kimes (1989, p. 350) and Weatherford and Bodily (1992, p. 832). I.e. more price-sensitive (leisure)

travelers book early because they usually know in advance when they will be traveling (for holidays, honeymoons etc.). On the other hand, less price-sensitive (business) travelers often have short-dated meetings and appointments with customers or partners which does not allow them to book far in advance.

On the one hand, early-booking customers provide the airline with a certain utilization of capacity (neglecting no-shows and cancellations) and are rewarded with a price discount as opposed to later booking ones. The latter must accept a higher price for the possibility to book relatively late and/or the flexibility of switching a flight. On the other hand, early-booking (business) travelers with a high willingness to pay must be prevented from *cannibalizing*, i.e. from buying the discounted products even if they can plan and book their trip early. The instruments for preventing cannibalization are grouped up under the term of *fencing* and can include—beside an early booking—a minimum length of stay or the lack of being refundable (see also Klein and Steinhardt, 2008, Sec. 2.3.1.3). Hence, the customer heterogeneity can be taken advantage of indirectly which allows minimum effort for dividing customers into different segments. For instance, the fencing structures make discounted tickets unattractive for most late-booking (business) customers. At the same time, the number of tickets sold at a discounted rate must be limited to reserve capacity for the higher-paying customers.

Hence, this type of demand heterogeneity (different willingnesses to pay and different preferences about the time of purchase) provides a firm with the ability of raising revenues. Indeed, the ability of exploiting the customers' willingnesses to pay makes up the essential motivation of applying RM in most industries (Kimms and Klein, 2005, p. 12). On the other hand, it makes necessary the decision whether to accept an early request (for a cheap product) or deny it in the hope that a later request (for a higher-priced product) arrives utilizing this capacity unit (Kimes, 1989, p. 350). For, if all customers had a homogeneous willingness to pay, capacity control would be redundant. In such a case, a simple first-come-first-served rule could be used for accepting incoming requests until all capacity units are occupied or no more demand is registered (Müller-Bungart, 2007, p. 4). On the other hand, if the customers had different willingnesses to pay but an equal preference about the time of purchase, auctions would be a more efficient approach for controlling the sale of products (Kimms and Klein, 2005, p. 12). See e.g. Caldentey and Vulcano (2007) as well as Vulcano et al. (2002) for treatments of auctions in RM.

Finally, it must be noted that the distribution of demand over time is not fixed which contributes to the heterogeneity. I.e. an early-booking customer need not always have a lower willingness to pay than a later-booking one (which is often assumed) and vice versa. A high-fare customer might know of a conference well in advance and a low-fare customer might want to make a spontaneous city trip or holiday. Furthermore, the total demand for the different products is not known in advance but is subject to stochastic variations. According to Kimms and Klein (2005, p. 13) the varying and uncertain demand is not an essential requirement for the application of RM but it affects the instruments employed within its context.

Restricted Operational Flexibility of Capacity

After presenting the exogenous characteristics of RM problems, we will now turn to the endogenous ones. Recall that in this context, we defined a *resource* as a finite and fixed bundle of individual capacity units. E.g. an air plane is a resource while a seat on this plane is a capacity unit. Hence, a product (as defined above) can utilize one resource (e.g. a direct flight from Munich to London) or several resources (e.g. a flight from Munich to New York with a stop over in London).

A property inherent to most industries applying RM is a restricted operational flexibility of capacity. That means that variations in demand cannot be absorbed by variations in supply easily and cheaply (Talluri and van Ryzin, 2004b, p. 14). More precisely, short-term (operational) adjustments of capacity are not possible or require a high effort and/or cost and thus do not pay off. If enough time is left for adjusting capacity short-dated and it pays off, it is e.g. possible to add or remove a (few) seat row(s) on an airplane or move rental cars from one station to another (Kimms and Klein, 2005, p. 10). A hotel company can e.g. divert customers from one site to another in the same city. However, in many cases, adjusting the capacity in RM industries is too costly and is not worth the effort, i.e. the revenue loss of a denied product request with a fully occupied resource is lower than the cost of adjusting capacity to accept the request (Müller-Bungart, 2007, p. 4). Also, a spontaneous walk-in customer at a car rental company might not want to wait until a car in his fare class arrives from another station and leave (if upgrades are ignored). A business customer might prefer a hotel at a certain site because of a nearby workplace; instead of taking a room at a different site from the same company, he would most likely make a request from a competitor nearby.

Beside the economic effects caused by a possible adjustment in capacity, the extent to which the given capacity can be adjusted and the time needed to adjust it must be considered. Considering adjustments on an operational level, the last point is especially crucial because the need to reduce or extend capacity often arises and/or becomes apparent short-dated and may not leave enough time for executing the adjustment (Klein and Steinhardt, 2008, p. 12). Hence, capacity planning is rather based on the long-term (estimated) demand for the resource and is done years in advance, partly because of very long lead times in the airplane production (Mayer, 2001, p. 56). This is one of the reasons why capacity planning is not part of RM per se (Klein and Steinhardt, 2008, p. 18).

The essential aspect which makes short-term adjustments of capacity a difficult task in RM industries is that capacity is sold in bundles (resources) and hence the provided number of capacity units within a resource is a multiple of a demanded unit (either in terms of some quantity or in terms of time). I.e. an additional (or larger) plane can possibly hold more passengers than the excess demand amounts to while an additional rental car remains with the company for months or years even if it is only demanded for one day, see Kimms and Klein (2005, p. 10). Since investments in capacity tie up a lot of money, RM industries typically have a cost structure consisting of high fixed costs for buying and/or leasing and operating capacity but low variable costs per demand unit served. Indeed the variable costs for serving a unit of demand (e.g. catering a passenger on board or handling his baggage) can be neglected if compared to the high fixed costs arising through e.g. costs for fuel, staff, and fees for take-offs and landings. See also Kimes (1989, p. 349). This cost structure allows for concentrating on maximizing revenue and approximating the maximal profit through a maximal revenue (Kimms and Klein, 2005, p. 5) which justifies naming the proposed concept “revenue management” (instead of e.g. “profit management”). Note that in the make-to-order industries mentioned above, variable costs can play a significant role which is why maximizing total profit might be more appropriate there. However, there the applied instruments are also subsumed under the term revenue management.

Standardized Products

The RM instruments of price differentiation and capacity control rely on a given, well-defined product range. In case a standardized product range cannot be established, RM instruments lose their applicability and other tools e.g. from project management can be used for optimizing the

revenue (Kimms and Klein, 2005, p. 14). This product range must further be fixed and sold over a long period of time in order to allow for sophisticated forecasts e.g. about future demands and willingnesses to pay in order to be able to control capacity efficiently and to receive the highest possible revenue. The standardization encompasses both, a standardized result of the service (e.g. a defined flight from some origin to some destination) as well as a standardized procedure for generating this service (e.g. in terms of check in, flight, and check out), see Klein and Steinhardt (2008, p. 16). Recall that in general, a *product* in the RM sense is a combination of a certain well-defined service (a flight from some origin to some destination in our case) and a fare for this service. In the following sections and chapters we will use the word "product" in this sense as well.

2.3 Capacity Control

As was mentioned above, the RM instrument applied in this thesis will be capacity control, which builds the core of modern RM systems. Whereas pricing is not a sole RM instrument and overbooking was used before the emergence of RM, capacity control in a large scale was made necessary with the deregulation of airspace in the US and thus went hand in hand with the emergence of RM (Klein, 2001, p. 148). It helps implementing the price differentiation with the intention of maximizing revenues. For, the best differentiated prices are worthless if there is no sophisticated system of allocating capacity in such a way as to exploit the customers' willingnesses to pay. We will introduce the main concepts used for controlling capacity in this section.

Capacity control is used to limit sales of less profitable products to reserve capacity to sell to more profitable ones later. In particular, capacity control is used to trade off a certain revenue connected to a present request versus an uncertain (expected) higher revenue connected to a possible request in the future. For, selling the capacity unit(s) needed to serve the present request may lead to a displacement of revenue due to having to deny a request for a more profitable product in the future if no capacity is left to serve the latter. This displacement of revenue can be interpreted as *opportunity costs* since an occupied unit of capacity cannot be used to generate profit from a more profitable product. However, if the present request is denied and future demand does not suffice to occupy all capacity units, some capacity will remain idle at the time of departure and revenue will be lost as well. Controlling capacity thus means balancing the risk of a capacity unit sold too cheaply versus an unsold capacity unit.

In general, there are two types of capacity control to be distinguished; revenue-based and class-based (Talluri and van Ryzin, 2004b, p. 31). The former type of capacity control is based on minimum prices to be paid for utilizing a (set of) capacity unit(s) in order for a product request to be accepted. These benchmark prices are called *bid prices*. The latter type of control is a based on determining quantities of seats to sell to or reserve for certain fare classes (this type is hence called "quantity-based" in Klein and Steinhardt, 2008). This is done in form of booking limits and protection levels, respectively. In this section, we concentrate on these quantity-based capacity control methods since booking limits will be the instrument for controlling in the competitive situations assumed in the later chapters. See Klein and Steinhardt (2008), Phillips (2007) as well as Talluri and van Ryzin (2004b) for more detailed definitions and applications of as well as discussions about bid prices.

Both types of control can be used to allocate capacity within single-resource (e.g. a direct flight involving only one plane) problems and network problems with more than one resource

(e.g. stopover flight involving two or more planes/flight connections). While both problems can be solved optimally with a dynamic program, it can seldom be applied in practice since the number of states grows exponentially with the number of products offered (Talluri and van Ryzin, 2004b, p. 83, 92). Hence, for realistic problems heuristics and approximations are more commonly used. Since our competitors will be assumed to operate in a network setting, we will describe them in more detail. We will also motivate our use of a deterministic linear program (DLP) for modeling the competitors' decisions and approximating their optimal revenues in our applications in the later chapters. See also Pak and Piersma (2002) for an overview of OR techniques for airline RM problems.

2.3.1 Quantity-Based Capacity Control

Controlling capacity based on quantities is done with the help of booking limits and/or protection levels. Both concepts are related and can in fact be deduced from one another. *Booking limits* specify maximal quantities of units to sell for certain products. As such, they prohibit the corresponding products to access capacity which is assigned to more profitable products in order to avoid revenue displacement. *Protection levels*, on the other hand, specify how many units to reserve (protect) for certain products. Requests for products are accepted as long as capacity is available and the booking limits or protection levels are not reached. These concrete values for the numbers of requests to accept make the quantity-based approaches more precise than bid price approaches which can only distinguish between acceptable and not acceptable requests. Both, booking limits and protection levels can be defined to be *partitioned* or *nested*.

Partitioned booking limits are separate blocks of capacity units (quotas) which may exclusively be accessed by the corresponding products. If the demand is known with certainty (i.e. demand is deterministic), partitioned booking limits even yield an optimal allocation of capacity (Kimms and Klein, 2005, p. 19). However, if demand is subject to stochastic variations (which usually is the case), partitioned booking limits can have a big disadvantage. For, with exclusive allotments, on the one hand, a product cannot use any capacity from the block allocated to a more profitable product if its own quota is used up. On the other hand, it also means that a request for a product may have to be denied if its quota is used up although a less profitable product's is not, since no product may access the capacity allocated to another. Certainly, this is an undesirable result which can lead to revenue displacements.

In order to avoid this issue, nested booking limits were introduced. These are based on a fixed ranking of the products and allow a product to access a lower ranked one's capacity units if they are not yet used up, while at the same time still prohibit the lower ranked products to access capacity of higher ranked ones. In other words, products within nested booking limits may access the capacity allocated to them and the capacity allocated to all lower ranked ones. A nested booking limit thus denotes the maximum number of capacity units to sell to the corresponding product and all lower ranked ones. See also Talluri and van Ryzin (2004b, p. 29). In case of single-leg capacity control, partitioned protection levels do not differ from partitioned booking limits. Nested protection levels, on the other hand, are the opposite of nested booking limits. While nested booking limits stand for maximal numbers of capacity units to sell, nested protection levels define a minimal number to reserve.

Considering multi-resource problems, the relationship between partitioned and nested booking limits and protections is harder to establish. Indeed finding a nesting structure (ranking of products) constitutes a problem for itself. The existence of different capacities on the resources

further complicates the problem. An early approach for implementing nested booking classes on networks is called *virtual nesting*. See also Phillips (2007, Sec. 8.4) and Talluri and van Ryzin (2004b, Sec. 3.1.2.2). In this context Müller-Bungart (2007, p. 45) pointed out that the nesting order depends on the demand of products and thus the problems of an optimal nesting order and of optimal nested booking limits have to be solved simultaneously.

Once the nesting order is established and the booking limits are computed, product requests can be accepted or denied based on a *standard nesting* or a *theft nesting* scheme, see also Klein and Steinhardt (2008, Sec. 3.3.5.4) and Talluri and van Ryzin (2004b, Sec. 2.1.1.3). Both nesting approaches propose the same accept/deny decisions if demand arrives strictly in the order "lowest to highest rank", see e.g. Talluri and van Ryzin (2004b, p. 31). However, their performance depends on the order of arrival and the booking limits which is why none of them clearly dominates the other. See also Müller-Bungart (2007, p. 42) for a discussion of when and why one nesting structure may dominate the other. A nested booking limit is the sum of the partitioned booking limits for this product and all lower-ranked ones. Thus, computing partitioned booking limits suffices for implementing a nesting structure, since the nested booking limits can be computed from the partitioned ones. This is why we will only provide the latter in our models below.

A last point worth noting is that while capacity control is the core RM concept applied in this thesis, RM consists of more instruments. The instruments closest related to capacity control are overbooking and dynamic pricing. While the former is connected to and often goes hand in hand with capacity control (see e.g. Subramanian et al., 1999 as well as Zhao and Zheng, 2001), the latter can be seen as a substitute for capacity control by adjusting prices over time (see e.g. Gallego and van Ryzin, 1997 as well as Maglaras and Meissner, 2006). *Overbooking* subsumes instruments for selling tickets for more capacity units than there are available on a resource to take into account the fact that some of the customers with a reservation will not make use of it at the time of departure. See Klein and Steinhardt (2008, Ch. 4) and Talluri and van Ryzin (2004b, Ch. 4) for more details. See Dunleavy (1995) for instruments and concepts involved in overbooking and affecting it. In case a customer with a ticket needs to be turned away because too many showed up, regulations prescribe a compensation (see European Union, 2004). With the compensation fees being higher than the product prices, these regulations lead to adherence of service levels on the side of the airlines since and limit the use of overbooking. Phillips (2007, Sec. 9.6) presented a list of alternatives for overbooking and Müller-Bungart (2007, Sec. 2.4.2) gave a literature overview about it. *Dynamic pricing* does not rely on limiting the capacity units to sell to certain products but instead relies on price changes over time to control—i.e. reduce or increase—demand (instead of controlling capacity). This way the goal of an optimal utilization of capacity can be reached more profitably. However, the concept's applicability depends e.g. on the costs and complexity connected with changing prices. While e.g. the retail sector is a natural sector with a dynamic pricing application, companies committing to prices and services e.g. in catalogs are not. See Klein and Steinhardt, 2008, Ch. 5 as well as Talluri and van Ryzin, 2004b, Ch. 5 for more details. See Müller-Bungart (2007, Sec. 1.4.3) for a classification and comparison of capacity control and dynamic pricing. Bitran and Caldentey (2003) and Boyd and Bilegan (2003) offered thorough literature reviews on (dynamic) pricing models.

2.3.2 Network Capacity Control

While single-leg capacity control is fairly easy, a network structure severely complicates the capacity allocation problem because a capacity decision on each leg needs to consider its impact

on the whole network (and on the total revenue). It then becomes insufficient (and possibly fatal) to optimize the seat allocation on each leg independently. In a network setting, accepting a request for a product occupying capacity on one leg may lead to having to deny a request for another (possibly more profitable) product occupying capacity on the same leg (and possibly others, too). Hence, it can be advantageous to deny a local flight request if a more profitable multi-leg request using this local flight is expected to arrive and vice versa.

There are several—mostly approximate—methods how capacity can be allocated to different fare classes in a network setting. Besides the optimal control found by a dynamic program, e.g. decompositions and mathematical programs can be applied to find approximations to the optimal solution. See Talluri and van Ryzin (2004b, Ch. 3) for an exact approach as well as descriptions of basic approximation models. In what follows, after introducing a dynamic program for allocating capacity optimally, we will point out the difficulties with an optimal control policy (or rather with the dynamic program needed to determine it) and describe approximations to it. Some are based on decomposing the original problem, while others focus on mathematical programming approaches. We will present such approaches below and motivate our model of choice for the further course of the thesis. See e.g. Chen et al. (1998), Dror et al. (1988),Wong et al. (1993) as well as Talluri and van Ryzin (2004b, Sec. 3.4) for further procedures.

Optimal Network Capacity Control Through Dynamic Programming

With the arrival of requests for airline tickets being a stochastic process, a stochastic dynamic program is necessary for modeling this process and allocating capacity optimally. Such a model is stated in Equation (2.1) and is described in e.g. Talluri and van Ryzin (2004b, p. 88).

$$r(\overline{\mathbf{C}}, t) = E\left[\max\{\mathbf{Prob}(t) * \mathbf{v}(\overline{\mathbf{C}}, t, \boldsymbol{\pi}) + r(\overline{\mathbf{C}} - \mathbf{M}\mathbf{v}(\overline{\mathbf{C}}, t, \boldsymbol{\pi}), t+1)\}\right] \tag{2.1}$$

It is assumed that there are $T = \{1, \ldots, \mathcal{T}\}$ time periods with $\mathcal{T}$ as the time of departure. The vector $\mathbf{Prob}(t)$ stands for the request arrivals in period t which are connected to the product prices. $\overline{\mathbf{C}}$, $\boldsymbol{\pi}$, and $\mathbf{M}$ are vectors of the remaining capacities on the individual legs, product prices, and capacity consumptions, respectively. The time periods are assumed to be small enough so that in each period at most one product is requested. The vector $\mathbf{v}$ contains the binary variables for accepting a product request ($v_{pf} = 1$) or denying it ($v_{pf} = 0$). They are functions of the remaining capacity, time, and product prices. Solving Equation (2.1) optimally thus leads to a maximization of the total expected revenue through maximizing the revenue in the individual states which are represented by the time index. The revenue in each state is made up of the expected revenue of the requested product plus the revenue in the next period(s) depending on the decision in period t and the remaining capacity units. It must be decided whether to accept a request in a period t or to deny it. The boundary conditions are $r(\overline{\mathbf{C}}, \mathcal{T}+1) = 0$ for all $0 \leq \overline{\mathbf{C}} \leq \mathbf{C}$ and $r(0, t) = 0$ for all $0 \leq t \leq \mathcal{T}$. This means that empty capacities as well as requests after departure do not contribute to the total revenue. In Klein and Steinhardt (2008, p. 103) the second condition was stated so that fully occupied planes generate a future revenue of $-\infty$ to make sure that a request is denied with certainty in this case.

Due to the large amount of possible states needed to be considered, the amount of memory needed to solve a dynamic program grows exponentially and determining an optimal solution becomes a very time-consuming task. Auxiliary nodes containing the stochastic information further complicate the problem. Due to this "curse of dimensionality", solving real-sized network

capacity allocation problems optimally using a dynamic program is "for all practical purposes impossible" (Talluri and van Ryzin, 2004b, p. 83). See also Bellman (1957) for the theory of dynamic programming and Klein and Steinhardt (2008, Sec. 3.3.2.3) for an example.

Partly because of this curse of dimensionality, the optimal network solution is approximated heuristically more often than the original network dynamic program is solved optimally. Common approximations to optimal network solutions rely on decomposing the network problems into smaller (e.g. single-leg) ones and solving them with extended single-leg approaches. When the network is decomposed into single-leg routes, established single-leg approaches can be used to solve the individual problems quite efficiently and to get a good approximate solution in an acceptable time. The sum of the individual single-leg objective values is the approximate objective value for the original problem. Decomposing network problems into smaller ones also has a historical background. Historically, single-leg problems were the first ones to be considered in RM applications and became sophisticatedly implemented in the airlines' RM systems. Multi-leg products were added only gradually and made up a small portion of the airlines' portfolio so that extending the readily implemented single-leg approaches to control capacity in networks was a sufficient solution. See also Belobaba (2012). To this day, networks have grown so large that solving a network problem optimally is simply not possible in practice. Thus, approximations must be used to receive a solution in an acceptable time in the first place.

Some decomposition methods rely on decomposing the original dynamic program in several smaller ones. The capacity is allocated in these smaller problems independently from each other and the network revenue is approximated as the sum of the individual problems' revenues. The simplest approaches split the original network into individual single-leg problems and solve each of them with a dynamic program. This is why this approach is called *dynamic programming decomposition* (see Talluri and van Ryzin, 2004b, Sec. 3.4.4). Cooper and Homem-de-Mello (2007) introduced two ways to decompose the dynamic program for network capacity allocation, a time-based and a state-based one. The *limited lookahead policy* described in e.g. Bertsekas (2005, Sec. 6.3) is a general dynamic programming technique which takes into account not all but only a limited number of states at a time and optimizes the decisions within these. This way the amount of memory to store the state information drops drastically. Birbil et al. (2014) introduced a method to decompose a network by origins and destinations. Their method entails the method of Curry (1990) as a special case.

Deterministic Linear Programming Approximation

Another approximation of the optimal objective value is based on mathematical programs. One of the earliest formulations of the network revenue management problem as a deterministic linear program (DLP) can be found in Williamson (1992). This is a simple, yet very elegant and effective approximation method and builds the basis of the models presented later in this thesis. In the DLP, as its name suggests, demand is assumed to be deterministic, i.e. known with certainty and all stochastic uncertainty is removed from the model. This deterministic demand can be provided as some expected parameter or the demand can be assumed to equal the mean demand. The model (2.2) – (2.5) is a DLP for determining booking limits in a single-airline network capacity allocation problem.

$$\max \quad r(b_{pf}) = \sum_{p=1}^{\mathcal{P}} \sum_{f=1}^{\mathcal{F}} \pi_{pf} b_{pf} \tag{2.2}$$

$$\text{subject to} \quad b_{pf} \leq d_{pf} \qquad p \in P, f \in F \tag{2.3}$$

$$\sum_{p=1}^{\mathcal{P}} \sum_{f=1}^{\mathcal{F}} M_{lp} b_{pf} \leq C_l \qquad l \in L \tag{2.4}$$

$$b_{pf} \geq 0 \qquad p \in P, f \in F \tag{2.5}$$

The objective function (2.2) maximizes the total revenue summing over the product prices times their booking limits. The products' booking limits are not to exceed their expected demand due to restriction (2.3). Restriction (2.4) is the capacity restriction requiring the sum of all booking limits on a leg not to exceed the airplane's capacity on this leg. Clearly, this is valid for all legs. In restriction (2.5) the booking limit variables b_{pf} are assumed to be continuous, non-negative. These represent the partitioned booking limits in the optimal solution.

Note that although the variables are defined as continuous, under certain conditions, they will take on integer values in the optimal solution. These conditions include 1) both, demand and capacity values to be integer-valued, 2) the network to be acyclic, and 3) all products to occupy only one seat (see also de Boer et al., 2002, p. 77). We will assume all of these conditions, which is why we will assume Poisson distributed demand. Not requiring the variables to be integer-valued keeps the model linear which can typically be solved more easily and faster than an integer model. Glover et al. (1982) showed that under these conditions the network capacity problem can be reformulated as a network flow problem.

According to Jensen's inequality (see e.g. Birge and Louveaux, 2011, Sec. 4.3), an optimal objective function value of the DLP an upper bound for the original stochastic dynamic program's optimal objective function value. In order to improve revenues realized from decisions based on mathematical programs (i.e. in order to lower the upper bound produced with the DLP) they can be re-solved as the time of departure approaches with more updated data. Note that for this, the demand information in constraint (2.3) and the capacity information in constraint (2.4) need to be updated to map the demand-to-come and the remaining capacity (lowered by the number of seats already sold), respectively from the time of optimization. Jasin and Kumar (2012) have shown that with an appropriate re-solving schedule the difference in revenues between the optimal solution of the dynamic program (2.1) and the optimal DLP solution is constant and independent of problem size. Hence, the solution obtained with the DLP is asymptotically optimal. See also Jasin and Kumar (2013) for a follow-up study and a comparison of DLP-based booking limit controls and bid price controls. Instead of reoptimizing problems as time passes, it is also possible to define dynamic (i.e. time-dependent) booking limits that are made available over time (see e.g. Lee and Hersh, 1993, Subramanian et al., 1999, Talluri and van Ryzin, 2004a as well as Zhao and Zheng, 2001). Bertsimas and Popescu (2003) implemented a DLP within an algorithm based on approximate dynamic programming.

Given that the DLP ignores any stochastic information, Talluri and van Ryzin (1999) introduced a randomized linear programming (RLP) method as an extension to the DLP. For the RLP, a number of demand realizations for the products is generated randomly and the DLP is solved with every one of these random demand values as the deterministic demand values in constraint (2.3). The objective value and booking limits (or bid prices, respectively) for the RLP are determined as average values from the DLP solutions. Although the RLP was able to generate better results than the DLP in the test bed used in Talluri and van Ryzin (1999), it did not clearly dominate the DLP. Hence, the authors concluded that further investigation was needed.

However, the performance greatly depends on the distribution of demand, the size of the sample, and even on the assumed network. It is easy to see that with a very large sample of randomly generated demand values, the (average) objective function value from the RLP converges to the objective function value of the DLP with the demand being equal to the mean demand. This phenomenon is called the *law of large numbers* which makes demand distributions useful in the first place. This law states that if a random experiment is repeated many times under the same conditions, then the average outcome of the experiments tends to get ever closer to the mean outcome. See also e.g. Hodges and Lehmann (2005, Sec. 6.9) as well as Moore et al. (2009, p. 274). Transferred to our application it means that if the DLP is solved optimally many times with new values for the demand but equal prices and capacities, then the average revenue, average booking limits and/or bid prices will tend to get ever closer to those values obtained from the optimal solution of the DLP with the mean demand values.

2.4 Assumptions for the Further Course

The topic of RM is very rich of applications and concepts. Since we cannot take into account all of them, we must make some restricting assumptions for the further course of the thesis. These assumptions will hold for all models and computational studies, unless stated otherwise. The particular assumptions are as follows:

A deterministic linear model will be used for optimizing the competitors' decisions.
In order to simplify the computation of booking limits in real-sized networks, we will employ extended versions of the single-airline DLP (2.2) – (2.5) for controlling capacities and approximating a competitor's optimal solution. The DLP substitutes the stochastic demand information by deterministic demands and is thus simple, yet effective and efficient to solve. Although the original capacity allocation problem has a stochastic nature, approximating it with a deterministic model need not lead to drastically worse results than in optimum. According to Müller-Bungart (2007, Sec. 6.2), assuming deterministic demand information is justified if e.g. the degree of uncertainty (i.e. the demand variance) is very low in the application at hand. He provided concrete data from the broadcasting industry in support of this statement. Further, Talluri and van Ryzin (1999) showed that although the DLP ignores stochastic information, it was able to match results by the RLP proposed by them and even outperform a probabilistic non-linear problem (PNLP) which takes into account the stochastic information of demand.

Partitioned booking limits will be used as the competitors' decision variables.
The models that will be proposed in the next chapters can be used to compute booking limits and to determine bid prices for controlling capacity. We chose booking limits as the control variables in our studies since they allow for a direct interpretation and implementation and for a network-wide optimization which will be considered in our studies. Bid prices, on the other hand, only provide information on a leg basis which can lead to bid price controls being misleading and yielding wrong accept/deny decisions when used for controlling capacity in a network. See Talluri and van Ryzin (2004b, pp. 90–91) for an example. Since the DLP is a static model, it can only produce partitioned booking limits. However, as the previous section showed, nested booking limits can be computed from partitioned ones if the nesting structure is known. See also Curry (1990) and Talluri and van Ryzin (2004b, Ch. 3.3.1) for a further discussion in support of using the DLP for controlling capacity.

Demand will be assumed to be exogenous and to follow a Poisson process with mean demand μ.

We will draw random demand values from the Poisson distribution for our computational studies. Since demand for airline tickets is discrete in practice, using a Poisson distribution is intuitive. See e.g. Kimms and Müller-Bungart (2006) for a detailed literature review in support of this statement as well as a sophisticated procedure how to simulate stochastic demand as a non-homogeneous Poisson process. Müller-Bungart (2007, Ch. 5) elaborated further details and provided a thorough literature overview. Furthermore, we will assume demand for different products to be independent. Prices will be assumed as given and will not be updated throughout the booking horizon.

All products will be assumed to occupy only one capacity unit on the resources they utilize.

We will make the common assumption that all products only occupy one seat on each airplane needed to serve the product. I.e. group bookings with ≥ 2 requested seats on each plane are considered as separate single-seat requests or can be treated as such by accepting less than the total requested number (see also e.g. Talluri and van Ryzin, 2004b, Sec. 2.4). This means that the incidence matrices in our models only have entries of 0 and 1.

No buy-ups and buy-downs will be considered.

Buy-ups and *buy-downs* mean that a more expensive or less expensive product, respectively is requested by a customer whose originally requested product is not available. Such reconsiderations of products are subsumed under the term of *customer choice*. We will not consider customer choice, i.e. if a customer is turned down at his preferred airline a, he does not try to buy a ticket from a different class. Instead, he turns to the competitor $-a$ with a request for the same product (the same itinerary in the same class) with probability $\alpha_{pf}^{-a,a}$. If he is turned down there too, the customer is lost. See e.g. Kunnumkal and Topaloglu (2008) as well as Liu and van Ryzin (2008) for extensions of the basic DLP to incorporate customer choice behavior. Also, upgrades by the competitors—e.g. placing a customer with a second-class ticket in a first-class seat to fill the seats there—are not considered here, as e.g. done in Steinhardt and Gönsch (2012).

No overbooking will be applied.

Cancellations and no-shows will not be considered, i.e. we will assume that all booked seats will also be taken by the customers and that everybody who bought a ticket also appears at the time of departure. This makes an overbooking control unnecessary.

Chapter 3

Strategic Airline Alliances

This chapter begins with an introduction and motivation of a strategic alliance before coming to strategic airline alliances (SAA) as a special case, which will make up the environment in which the competition in the later chapters will take place. As strategic alliances can also be found in other industries apart from airlines, we cannot cover the whole spectrum of this topic. See e.g. Dussauge and Garrette (1999), Lorange and Roos (1993) as well as Shenkar and Reuer (2006) for more details on alliances. We will define code sharing, one of the fundamental concepts in SAAs and subject of coordination issues within them in Section 3.2. This concept is crucial when it comes to capacity control in SAAs, which is treated in Section 3.3. In Section 3.4, the necessity for forming SAAs will be pointed out before the competitive aspects of SAAs are described by showing why and how alliance partners continue competing to some extent. In Section 3.5, the relevant assumptions for the further course are summarized.

3.1 Definitions and Motivations

There are several, similar definitions of strategic alliances as all authors comprehend an alliance differently and deem some aspects more important than others. What is commonly agreed on and need no discussion is that, as with any form of cooperation, at least two partners are involved in an alliance and that it is laid out for a long (strategic) period of time which necessitates contractual agreements, unlike mere short-termed exchange deals. Lorange and Roos (1993) merely distinguished strategic alliances by the degree of vertical integration and the mutual interdependence between the partners. Das and Teng (2000, p. 77) defined a strategic alliance as an "interfirm cooperative agreement aimed at pursuing mutual strategic objectives". Casson and Mol (2006, p. 28) gave a more detailed yet general definition which takes into account that the cooperation is carried out by independent firms, is open-ended (i.e. may be subject to changes) and involves complementary investments and sharing of the benefits. The definition by Mohr and Spekman (1994, p. 135), at last, is the most comprehensive one in the sense that it also includes that the partners "acknowledge a high level of mutual interdependence".

What the last two definitions take into account explicitly and is important to stress is that the parties involved in strategic alliances are independent firms. This means that all partners must decide about questions concerning the alliance's goals and instruments how to achieve them what necessitates that the partners' interests concerning the cooperation must be equal or at least complementary for an alliance to be successful. Consequently, a strategic alliance needs to be distinguished from e.g. mergers or takeovers in which only one party takes the

control. Since strategic alliances are founded on contractual agreements (see Ariño and Reuer, 2006 for a literature overview on alliance contracts), the involved parties give up some of their economic autonomy and subordinate some of their goals to those of the alliance. Some partners even acquire equity shares of one another in order to demonstrate their long-termed interest on the partnership which also enhances the stability of the alliance (Maurer, 2006, p. 77). On the other hand, being independent from each other, the firms can act completely autonomously in fields not regulated by the alliance contract or even that the alliance partners can compete with each other. Indeed, Backhaus and Piltz (1990, p. 3) pointed out that strategic alliances are characterized as horizontal cooperations entered into by firms on the same step of the value chain which hence are present or potential competitors. See also Conrady et al. (2013, p. 278). In this thesis, we assume competition within alliances and show its effects on the competitors' and on the alliance's revenues.

Strategic airline alliances were formed as the airlines strove for expansions of their networks beyond country and continental borders. While the first cooperations were aimed at reaching domestic customers in low-demand areas leading large network carriers to team up with small regional actors, the trend quickly shifted to international, even global alliances to span the whole world (Boyd, 1998). Taking into account the time and effort necessary to build an own network reaching around the globe, airlines quickly learned that this was no real option, so they began considering fusions of existing networks with ready infrastructure, staff etc. Finding themselves faced with high financial and temporal burdens for a "natural" growth as well as legal aspects concerning mergers and acquisitions, though, the airlines looked for alternative ways which enabled them to offer destinations beyond their present network. An own expansion into a new market brings with it the further issue of having to compete with an established set of actors as an entrant. Chen and Ross (2000) even proposed allying with a possible competitor as a strategy for deterring him from entering the market. See Kleymann and Seristö (2004, pp. 3 – 5) for several reasons for building an SAA instead of e.g. merging with a partner or acquiring a competitor.

Comparing the risks and benefits connected with a merger or majority stakes, one finds that the former often outweigh the latter. These risks entail the high price for airline's equity, the industry's low profit margins, and hence its sensitivity to variations in demand, fuel prices, and personnel matters like requests for better wages or working conditions and possible strikes connected with them. Another burden merging airlines would have to face is that to this day the industry is still highly regulated by laws which, among other things, limit the amount of shares of a national airline to be held by foreign companies. This ensures that the control over an airline retains in domestic hands and e.g. allows the government to draw on civil resources in case of a national crisis or issues of national security (Conrady et al., 2013, p. 35). A third issue is antitrust regulation forbidding the complete merger of large airlines in order to retain competition in the airline market. Since the number of competitors in the airline markets is rather small, a further consolidation of companies would reduce competition even further and might lead to higher prices and possibly further disadvantages to the customers like a lower frequency of flights. Antitrust immunity might be granted to certain alliances or partners if this benefits the general public more than it harms competition, but this must be proved first. In context of the last two points, (Oum et al., 2000, p. 15) argued that legal barriers aimed at accessing a foreign market and limitations concerning the ownership of airlines (or their equity) by foreign airlines were the main reasons for alliance formations. See Oum et al. (2000, Ch. 11) as well as Oum et al. (2001) for an overview on regulatory issues concerning global airline alliances. See Brueckner and Pels (2005) for an investigation of the economic effects of the merger between

KLM and Air France. Zhang and Zhang (2006) considered such issues within a general alliance formation process.

Although many of these laws and regulations have been relaxed in the past, a completely liberalized airline market is deemed unrealistic. Oum et al. (2000, Ch. 10) used game-theoretic models to show why this is so. The authors also gave explanations of why bilateral liberalization agreements are more common than multilateral ones and give practical reasons. And even if it could be achieved, Pels (2001) argued that the need for building alliances would not disappear completely (in part because of the financial and organizational issues connected with expanding an airline's network named above). In this context, the technical issues of a merger need to be considered. For, the airlines' RM systems were customized to their individual needs with the aim of providing the respective airline a competitive advantage over its competitors. Thus, it is unlikely that there are two identical RM systems which would make merging of these systems a very complex, if not impossible, undertaking (Boyd, 1998). Finally, the personnel aspects must be taken into account as an obstacle (see e.g. Porter and Fuller, 1986, p. 329). For, a fusion between two (or more) airlines will inevitably bring with it a consolidation of the management team and possibly have further consequences for the staff including operational changes for the tasks performed on a daily basis which might encounter resistance. See also Lorange and Roos (1993, Ch. 5) for a treatment of human resources in strategic alliances.

Currently, the three biggest airline alliances are the Star Alliance, the first of its kind to be founded in 1997, OneWorld, founded in 1999, and SkyTeam, founded one year later. After having established themselves, these three alliances grew ever larger and in 2012 reached a total market share of 61.2% over all global airline traffic ("Airline Alliance Survey", 2013). However, the environment of SAAs is currently subject to change with Etihad Airways from Abu Dhabi actively establishing an own, loose alliance with different partners from over the world, see Etihad Airways (2014). Especially, Etihad is looking for partners around the world in which they can invest in terms of equity shares and thus influence strategic decisions.

3.2 Code Sharing

According to Zhang and Zhang (2006), strategic alliances are most common in network-orientated industries like logistics, shipping, and airlines. Yet, the importance of strategic alliances in the airline business was already shown by Hergert and Morris (1988) who found it to play the largest role in high-tech industries like automobiles, telecommunications, and aerospace. While alliances in the other industries seek cost reductions through e.g. more efficient operations, joint research activities, and joint purchasing, for SAAs a study conducted by Kleymann and Seristö (2004, Ch. 2) found joint technical operations (like ground handling and joint maintenance) aimed at reducing costs to play the least role. Instead, SAAs are mainly aimed at raising profits through the revenue side with code shares being subject of SAA contracts most frequently. Cooperation forms aiming at raising the revenue instead of lowering the costs are also due to the cost structure in RM industries described in Chapter 2 which does not allow for significant cost reductions but bears large potentials for increasing revenues. Other forms of cooperations are e.g. franchise and leasing agreements where an airline uses another one's planes and/or staff for its own use. See e.g. Maurer (2006, Sec. 2.4) as well as Conrady et al. (2013, Sec. 2.2.4) for more details.

Code sharing agreements allow the involved partners to sell products which utilize each other's resources as if they were their own (Oum et al., 2001, p. 57). Each of the partner airlines

(the marketing carriers) then sells the capacities under their own flight number (code), although a part or even all of the route is operated by a different airline (the operating carrier). Code sharing is thus the crucial instrument that allows the alliance members to offer destinations beyond their own network by utilizing their partners' resources. Depending on what the partners have agreed on, the marketing carriers pay the operating carrier a fee for using his resources. Consider e.g. a route from Frankfurt/Main to New York–Newark over Hamburg on a Friday morning departing at 7:00 a.m. The leg from Frankfurt to Hamburg is operated by Lufthansa and the leg from Hamburg to New York is operated by United Airlines, both members of Star Alliance. While Lufthansa sells the two flights under codes LH002 and LH7605, United has them listed as UA8986 and UA75, respectively. This form of code sharing is called *complementary* because the adjacent legs are operated by different airlines. It will also be the type of code sharing used in our models. *Parallel code sharing*, on the other hand, is found on routes which are operated by all involved partners. This form of code sharing is used to increase the frequency on the flight by coordinating flight schedules.

Beside the image improvement connected with a renowned partner, code sharing leads to higher load factors on the established flights and to more attractivity because more destinations are offered. The load factors increase since several partners' customers are placed in the same plane which leads to a more efficient utilization of resources. Further, the revenue can be increased by being able to accept the most profitable customers from the bigger pool. Such a cooperative form of market expansion is much more efficient and easier than having to compete with an established actor in a new market. Through coordinated operations of the ground personnel and baggage handling, customers do not even notice that they are traveling with different airlines if they use code shared flights. This *seamless travel* raises their satisfaction and ties them closer to the alliance as a whole. Using a common customer loyalty program (e.g. for frequent fliers) and uniform lounges amplifies this effect further. According to Oum et al. (1993, p. 16), a single airline would not be able to set up a flight network reaching around the whole world on its own due to the necessary financial, organizational or temporal efforts. Indeed, with the alliance partners cooperating closer and closer, they can generate almost the same benefits as if they had merged or entered a new market but with a much smaller effort and costs (Talluri and van Ryzin, 2004b, p. 121). It is even argued that these days alliances play such a great role in the airline business that "no airline [...] can operate in a vacuum" and not belonging to one means "going against the mainstream" (Kleymann and Seristö, 2004, p. 17, 97).

However, with code-sharing agreements in use, the problem is not only to allocate capacity to different products, but also to divide the available capacity among the partners within the alliance. Given these decisions, it must be decided how much of an airline's capacity should be made available to the alliance partners. Further, with more than one airline involved in the sale of tickets, the problem of how to divide the profit amongst them arises. Finally, the possibility of competitive aspects connected to code sharing like the coordination about the number of connecting passengers to book on a flight needs to be considered. These issues are described in more detail in the section after next.

3.3 Capacity Control in Strategic Airline Alliances

We now come to describing the capacity control problem within SAAs which adds complexity to the problem since it must consider all partners having access to an airplane's capacity. First, we define the different options for allocating seats to the partners, though. These can

be distinguished by the intensity of collaboration and differ in terms of the rigidity of allocated space, the risk bearing, and pricing (see also Boyd, 1998, Maurer, 2006, p. 70, as well as Conrady et al., 2013, p. 277). See Graf (2011, Sec. 3.2) for an overview of RM applications in alliances in other industries.

A typical form of reserving capacity for partners is to use *blocked space* agreements. These allocate a number (block) of capacity units to the marketing carrier and enable him to sell these capacity units freely to his customers. Blocked spaces can further be distinguished into *hard blocks* and *soft blocks*. The former represents a fixed amount of capacity which cannot be changed during the selling period, while the latter allows for an adjustment of capacity units in case of an unexpectedly high or low demand on either side. One can see that hard blocks have similar disadvantages like partitioned booking limits for both partners, the marketing and the operating carrier. If the demand for either partner turns out higher than the number of seats he has available while at the same time the demand for the other partner turns out lower, both lose money because customers have to be turned down by one while at the same time the other has free seats available. This is the reason, soft blocks are more commonly used than hard blocks. The code share agreement of a *free sale*, on the other hand, does not rely on reserving capacity at the outset. Instead, both partners have full access to the operating carrier's capacity and can sell seats freely up to the capacity limit.

Both agreements mentioned so far allow both, the marketing and the operating carrier, to set prices for the code shared products freely. More intense forms of collaboration, namely *revenue sharing* and *profit sharing* rely on setting prices jointly in order to appear more like being two parts of one alliance instead of simply working together. The revenues are then split according to some predefined rule. Profit sharing even goes one step further and involves dividing of the costs among the partners which can be seen as the tightest form of cooperation. This e.g. makes sense on parallel routes with two not fully occupied aircraft. One airline can then cease offering the flight and fully rely on the partner. Since the now free aircraft can be used on a different route, the corresponding airline can compensate the other for the expenses.

Given that the RM problem in SAAs is fairly young compared to individual airline RM, the research on it is fairly scarce. While the single-airline RM problem was to find capacity allocations for the different products, in alliances three problems arise connected to code sharing. First, the legs to be made available for code sharing with partners must be determined. Secondly, a fair scheme of sharing the partners' revenues from code shared products must be found so that all partners are satisfied and the alliance revenue is maximized. Only then the third problem can be solved; determining the number of seats reserved for code shared products.

Abdelghany et al. (2009), LaRoche et al. (2012) as well as O'Neal et al. (2007) tackled the first problem and provided rules for choosing the legs to be made available for code sharing. Abdelghany et al. (2009) used a non-linear model for this decision and took into account the possible revenue displacement by their local customers through code shared ones. They presented a genetic algorithm to solve the problem heuristically. LaRoche et al. (2012) investigated how the demand is affected by code sharing and also how a company's flight network is affected. They used a linear model to distribute the demand among the network and used a heuristic to choose the itineraries for code sharing in a second step. O'Neal et al. (2007) used a mixed integer problem with binary variables representing the legs to offer for code sharing. This model was integrated in a three-step procedure that first created the network itineraries, then made demand forecasts for this network, and finally chose the best legs to share codes. Given the simpler model, they were able to solve the model directly with commercial software.

Boyd (1998) was one of the first researchers who saw the need to coordinate the partners' decisions to maximize the alliance's total revenues. Recognizing the issues with centralized decisions, he suggested several decentralized rules for allocating capacity in SAAs. He discussed static rules for sharing revenue in a code share agreement and criticized that these rules—typically agreed upon and fixed before any demand is realized—can lead to suboptimal decisions concerning the total network revenue. The reason is that such rules allow the partners to behave selfishly and e.g. favor a local customer over a code shared one if the former is more profitable for the concerned airline, even if the latter may benefit the whole alliance more. One can see that the problem of alliance RM is similar to a single airline's network RM problem described above. However, alliances add complexity to the problem since there the partners need to share information about their seat availability and often pursue their individual goals instead of optimizing the alliance revenues. Boyd (1998) noted that in case of a free and complete exchange of information and capacities, decentralized decisions could yield the same output (maximal alliance revenue) like a centralized decision. However, due to technical challenges and legal constraints, this seems unrealistic and the partners' selfish behavior leads to suboptimal decisions.

The studies by Shumsky (2006) and Vinod (2005) investigated the practical challenges of SAAs and gave advice for the future. Both authors pointed out the need to loosen the revenue sharing agreements and, instead of using rigid rules, suggested more flexibility in order to be able to react to unexpected demand fluctuations during the booking period. Shumsky (2006) compared the alliance RM problem to a coordination problem of supply chains and provided suggestions for improvement of the revenue sharing problem. Vinod (2005) reviewed several rules for revenue sharing which would be applied in case one partner sells tickets for another's capacities and concluded that they were fairly simple concentrating e.g. on the distance flown or simply entailed a fixed price to be paid.

Belobaba and Jain (2013) further detailed the technical challenges connected with information sharing in SAAs as airlines try to determine optimal code sharing booking limits in absence of the partners' information in order to maximize their own and overall alliance revenues. They showed two procedures for sharing the airlines' information to increase overall revenues. Taking into account the selfishness of airlines, Kimms and Çetiner (2012) developed a revenue sharing mechanism to share revenues in a fair way and thus to ensure the long-term stability of the alliance so that none of the partners has an incentive to leave the alliance. They used cooperative game theory for their analysis showing that the core of their game was not empty and made use of the nucleolus concept for their revenue sharing mechanism. In particular, they approximated the centralized nucleolus solution which would necessitate solving an exponential number of linear programs. In a follow-up study, Çetiner and Kimms (2013) used a method to find decentralized revenue sharing mechanisms which is a more realistic setting since airline partners behave selfishly and pursue mechanisms that benefit themselves most. On the one hand they introduced an assessment procedure for decentralized revenue sharing schemes and proposed a new procedure for sharing revenues based on dual prices on the other hand. See also Çetiner (2013).

Graf and Kimms (2011) have developed procedures based on real options to solve the problem of allocating code shard capacity for a two-airline alliance. There, the marketing carrier buys options for occupying the operating carriers's capacity and pays another price in case he makes use of an option. To optimize the partners' decisions—seats to make available for code sharing and number of code shared seats and options to reserve, respectively—the authors proposed using two DLP-like models (one for each partner) or EMSR heuristics. The prices to be paid for

the operating carrier's capacity were assumed to be fixed. This assumption was relaxed in Graf and Kimms (2013) who introduced a transfer price optimization rule. The authors developed an iterative procedure for optimizing booking limits and transfer prices. A similar method was used by Topaloglu (2012) who proposed a decomposition approach for determining alliance booking limits and transfer prices based on a centralized DLP. In particular, the centralized DLP was decomposed into several smaller ones (one per alliance member) and the members' information was modeled in the decentralized model with dual variables. These dual variables indicated how to share the revenues between the partners. In Chen and Hao (2013) the problem of overbooking in SAAs was addressed for a two-partner alliance offering parallel flights. Besides the capacity allocation, the authors addressed the question of service levels, a main concern in overbooking policies. Their solution can be solved numerically using a spread sheet application which is essential for operative use.

Netessine and Shumsky (2005) were the first authors to use non-cooperative game theory for determining optimal booking limits for code shared products in an alliance. They proposed a competitive game for a two-partner alliance in which both partners operated adjacent legs and both had to decide how many of their capacity units to reserve for code shared (connecting) customers and how many to leave for their own (local) products. Hu et al. (2013) used the same setup, yet they described the alliance formation and operation process as a two-stage game. In the first stage, a cooperative game was used to determine optimal revenue sharing rules for code shared products, while the second stage used a non-cooperative game to determine optimal booking limits for all products using the airlines' legs. The authors focused on revenue sharing mechanisms that lead to maximal revenues for the complete alliance. They modeled the individual partners' decisions so that their models incorporated the central solution.

As the first authors, Wright et al. (2010) picked up the suggestion of dynamic and flexible revenue sharing rules and presented a dynamic stochastic game to optimize the alliance capacity control over time. In particular, they described dynamic revenue sharing rules to guide alliance behavior during the booking horizon and help make better accept/deny decisions as customer requests arrive. However, the authors assumed complete information on all sides which is little realistic. This approach was refined in Wright (2014) assuming that the allies were unwilling or unable to share certain information because of technical or legal restrictions and hence had to make decisions under incomplete information. Indeed, he assumed that the only information available to the partners was the transfer (bid) price per unit of capacity. The author introduced a decomposition rule for a central dynamic program to determine approximate bid prices in an airline alliance for the individual partners.

3.4 Competitive Aspects in Strategic Airline Alliances

As was shown above, allying with competitors was a good (and often even the only) strategy for airlines to expand their global reach and will continue to do so. However, while SAAs are meant to enable cooperation between airlines, they also have multiple competitive aspects within themselves that are not to be ignored. Recall that an SAA is a horizontal alliance and hence constitutes a form of cooperation between competitors. As described above, an essential characteristic of strategic alliances is that their members retain their economic autonomy and that alliances are hence open-ended, even if they are intended to last a long time. This means further that a firm which is a partner today might be a competitor in the future which consequently necessitates some caution of which information to make available to the ally. E.g. merging RM

systems would mean revealing sensitive information about demand and RM instruments used. In this context, Lorange and Roos (1993, Ch. 4) advised retaining a firm's core competences which need not necessarily be revealed to the allies in so-called "black boxes" which protect this sensitive knowledge from being used against oneself in the case of a failing alliance. See Kleymann and Seristö (2004, pp. 25 – 28) for several cases of SAA failures in the past decades. Black boxes have the further advantage that they keep the respective company interesting for the partner and make the latter dependent on the former.

Indeed, as with effect from 31 March 2014, US Airways, a former member of Star Alliance, merged with American Airlines, a OneWorld member. In effect, US left the Star Alliance and joined OneWorld. At the same time, TAM, a Brazilian airline, switched from Star Alliance to OneWorld joining its partner LAN airlines from Chile. See OneWorld (2014b) for more details. If US Airways and TAM had merged their RM systems with their partners of Star Alliance, upon them switching to OneWorld, the latter alliance might have gotten access to sensitive information about all other Star Alliance members, be it by intention or by mistake. This, on the other hand, would lead to an invaluable benefit for OneWorld and a huge catastrophe for Star Alliance.

Competition is already evident in the formation process of strategic alliances. For, irrespective the long-term cooperative goals of the alliance, the partners constantly pursue their individual goals in the first place. This means that after entering an alliance, each airline wants its utility (in forms of e.g. revenues and brand image) to not be lower than without the alliance (otherwise allying would make no sense at all). Even more, entering a certain alliance must bring with it the highest utility than all other alternatives (including other alliances and not entering an alliance at all). Beside the airlines themselves, their home governments are involved in the alliance formation process as well since each nation has the complete sovereignty about its airspace and has the right to grant or to withdraw permissions (freedoms of the air) for e.g. landings and takeoffs to foreign airlines. Political friction can thus endanger an alliance as happened e.g. in 2007 when Russia withdrew the airline Lufthansa Cargo the right to pass the Russian airspace in order to induce the airline choosing a Russian airport as a hub in the region instead of one in Kazakhstan (Conrady et al., 2013, p. 39).

And even after an alliance is founded, competition remains a noticeable factor. Not only that each member tries to steer the alliance into a direction which is most preferable for him. The competition in SAAs continues to exist on an operational level by the partners offering similar connections from the same (or close) origins to the same (or close) destinations which makes them compete for customers actively on these routes. To this "horizontal" competition, as we will come to call this direct competition type, comes another "vertical" form, namely the competition for an airline's capacity when it comes to code sharing. On the one hand, an operating carrier might benefit from code sharing to receive a higher load factor and gain revenues. On the other hand, when he accepts bookings for code shared products, he runs the risk of having to deny one of his own (possibly more profitable) customers. This issue will be addressed in Chapter 7.

To sum it up, the financial burdens, legal aspects limiting the amount of shares held by foreign companies, and antitrust laws restricting the possibilities of international mergers and takeovers for airlines make it almost impossible for airlines to grow into an international player on their own or by merging with others. Also, the heterogeneity of the airlines' RM systems make them almost impossible to be merged and the resistance on the side of the management team might lead to a failure of merging plans after all which often leaves alliances as the sole option for airlines to offer world-wide services. Yet, the formation of SAAs and even their operation

on parallel flights and concerning code sharing are characterized by competitive aspects as each member seeks its own maximum benefit. This means that SAAs still need to be considered on a decentralized, even competitive, level. This is done in this thesis.

3.5 Assumptions for the Further Course

In this section we shortly state the assumptions concerning the alliance environment under which the airlines will compete in the later chapters of the thesis. Since the structures of RM systems or financial aspects are not part of our analysis, we will focus on the operational issues only. In particular, we assume the following:

The alliance partners will be assumed to have no antitrust immunity.

We will assume the alliance partners to be involved in a fairly young stage of the alliance formation process which does not include antitrust immunity yet. This means that the allies cannot set prices and booking limits centrally to benefit the whole alliance. Instead, each partner (and competitor) is only interested in maximizing his own revenue based on the information he has gathered about the other.

The cooperative aspect will be taken into account by means of code sharing only.

As was mentioned above, SAAs may bring along with them the coordination of schedules, common service agreements, and lounge usage, among others. We will ignore these aspects of allying and concentrate solely on code sharing, i.e. on the determination of booking limits for local and code shared products. Further, the revenue sharing rule will be assumed to having been agreed on before the determination of booking limits and will not be subject to optimization. In particular, we will assume only complementary code sharing indicating a fairly young stage of the alliance formation process.

The partners will not be allowed to share crucial information for the capacity control problem of code shared products.

We will assume the partners to have no common RM system but to optimize their own decisions decentrally. I.e. no information about the relevant capacity control decisions will be actively shared by and with the partner. Also, no other information (e.g. demand) will be assumed to be shared actively between the airlines. This assumption is motivated by the absence of antitrust immunity on the one hand and by the complexities of joining RM systems on the other. This does not mean that the competitors will be assumed not to possess this information. Instead, they will be assumed to acquire the relevant data themselves e.g. over third-party providers.

A blocked space agreement will be assumed to reserve capacity for the partner.

We will assume that the partners are engaged in a blocked space agreement to decide how much capacity to reserve for code shared products. In particular, since the model used (a DLP) yields deterministic and static quantities, we only get hard blocks since we do not assume a reconsideration of decisions.

Chapter 4

Selected Topics in Non-Cooperative Game Theory

In this chapter we will define the relevant concepts of game theory used in the later chapters. We will restrict ourselves to the non-cooperative aspects of game theory since competition is the key issue treated in this thesis. We will begin with general definitions and explain important concepts with the help of simple examples of RM under competition in Section 4.1. Since not all of game theory's concepts can be treated here, we refer to the textbooks of e.g. Fudenberg and Tirole (1991), Holler and Illing (2009), Myerson (2001) as well as Osborne and Rubinstein (1995) for further reading. Section 4.2 will introduce algorithmic game theory which is a fairly young discipline focusing, among other things, on the computation of Nash equilibria. This is the central solution concept in non-cooperative game theory which will play a role in solving the competitive situations in the later chapters. Section 4.3 treats the computation of so-called extreme Nash equilibria using the algorithm by Audet et al. (2001). This algorithm will be helpful for motivating our own algorithm in the next chapter and will be used for comparison. Finally, in Section 4.4 we will summarize the assumptions on which the competition will be based in the later chapters.

4.1 Definitions

Game theory (GT) is used to analyze situations in which two or more decision makers interact whose utility is affected by each other's actions (see e.g. Holler and Illing, 2009, p. 1, Myerson, 2001, p. 1 as well as Osborne and Rubinstein, 1995, p. 1). With the theory's first application being the analysis of games such as e.g. chess and poker, these situations of interaction are still called *games* and the decision makers are called *players*. For an overview of the various fields of application of GT and a discussion see e.g. Binmore (2003, Sec. 0.4).

In order to be able to use GT for analyzing games, it is crucial to assume that the players be *intelligent* and *rational*, see also Myerson (2001, p. 2 – 3). According to Osborne and Rubinstein (1995, p. 1) the players are intelligent if they "reason strategically". An intelligent player knows the situation he is in. More precisely, he knows all characteristics about the situation and knows that every player has the same knowledge as he does. Hence, an intelligent player has the same state of information as the person studying the game. A player is rational if he pursues only his own objective, irrespective of all other players' outcome. In particular, he chooses one of his actions so that his utility is maximized, given the action chosen by the other player.

A game is made up of the set $\mathcal{A}$ of *players* playing the game, a non-empty set S^a of *strategies* for each player a as well as a *utility function* $u^a(s_1, \ldots, s_{|a|})$ for each player depending on his own and the other players' strategies (Fudenberg and Tirole, 1991, p. 4 and Osborne and Rubinstein, 1995, p. 11). In this thesis, the players are the airlines which compete for costumers within an RM setting. Consequently, we will use the terms "player", "airline", "competitor", and "opponent" interchangeably. We will assume throughout this thesis that in our games only two players are involved and thus further definitions of concepts will be made based on two players only. However, they can be extended to ≥ 3 players as well without loss of generality. We will call the players $a \in \{1, -1\}$ whereas index a will refer to the considered player, while $-a$ will refer to the competitor. Strategies can be divided into *pure strategies* and *mixed strategies*. The former represent actual (sets of) actions a player can take, while the latter define probability distributions on the available pure strategies (Fudenberg and Tirole, 1991, p. 5). In our case, an action stands for a product's booking limit while a pure strategy is a vector entailing all decisions about booking limits for a player's offered products. A player's utility will be measured by the total revenue which is to be maximized in the applications throughout this thesis. We next clarify these and further components in more detail using an RM example.

Assume two airlines (1 and 2) to compete on only one flight leg O – D offering the flight OD in two fare classes, l and h, with prices given and fixed as $\pi_l^1 = \pi_l^2 = 1$ and $\pi_h^1 = \pi_h^2 = 2$, respectively. Assume further both competitors' planes to have a capacity of 100 seats and demands to be $d_l^1 = d_l^2 = 67$ and $d_h^1 = d_h^2 = 31$. Let the players' pure strategies $s^a = (b_l^a;\ b_h^a)$ be limited to three booking limit combinations: $s_1^a = (60;\ 40)$, $s_2^a = (65;\ 35)$, and $s_3^a = (70;\ 30)$. Note that in this case we need not explicitly state the booking limits for both classes since the booking limit for each class is uniquely determined as the difference of the capacity and the other booking limit. However, according to our definition of pure strategies (being vectors entailing all decisions about booking limits for a player's offered products), we state all booking limits explicitly.

One can see that with strategies s_1^a or s_2^a the players accept less low-fare passengers (60 and 65, respectively) than there is demand (67) which means that some of the passengers willing to buy a low-fare ticket will be turned away. Likewise, strategy s_3^a leads to denied passengers in the high-fare class. In practice, a passenger will try to purchase a ticket from the competing airline with some probability $\alpha_{pf}^{-a,a}$ in case his request is not accepted by his preferred one. This effectively increases the competitor's demand for the respective product which is now given as $d_{pf}^a + \alpha_{pf}^{-a,a}(d_{pf}^{-a} - b_{pf}^{-a})^+$ with $(d_{pf}^{-a} - b_{pf}^{-a})^+ = \max\{0; (d_{pf}^{-a} - b_{pf}^{-a})\}$ denoting the non-negative number of booking requests denied by the competing airline $-a$ (spillover demand of $-a$). In this example we assume $\alpha_{pf}^{-a,a} = 1$ for all products and both players, i.e. all denied customers make a request for the respective product at the competitor.

Taking this fact into account, the airlines have the opportunity to accept the passengers denied by their competitor and possibly increase their revenue by playing appropriate strategies. All in all, their revenues (prices times the lower value of booking limit and demand for a product) would be as shown in Table 4.1. The rows and columns represent pure strategies of player 1 and player 2, respectively while the entries in the matrices represent the players' revenues if player 1 plays strategy s_i^1 and player 2 plays strategy s_j^2. Note that the matrices suffice to describe all of a two-player game's components; the number of matrices reveals that there are only two players, the rows and columns describe the strategies and the matrix entries show the utilities. Two-player games being represented by the players' payoff matrices are called *bimatrix game*.

Taking a look at the entries of matrix r^1, one can e.g. see that with player 2 playing strategy s_2^2 (s_3^2), strategy s_3^1 (s_2^1) yields player 1 the highest revenue. Such strategies which maximize a

r^1	s_1^2	s_2^2	s_3^2
s_1^1	122	122	124
s_2^1	127	127	129
s_3^1	130	129	127

a) Payoff Matrix r^1

r^2	s_1^2	s_2^2	s_3^2
s_1^1	122	127	130
s_2^1	122	127	129
s_3^1	124	129	127

b) Payoff Matrix r^2

Table 4.1: Payoff Matrices for the One-Leg, Two-Class Game

player's utility given the other player's strategy are called a *best response*. Applying the same reasoning to matrix r^2, one can see that s_3^2 (s_2^2) is a best response to player 1 playing strategy s_2^1 (s_3^1). Hence, the pure strategies s_2^1 for player 1 and s_3^2 for player 2 (likewise, strategies s_3^1 and s_2^2) are mutual best responses which constitute a so-called *Nash equilibrium* (NE) of a game. In this state of a game implementing a different vector of booking limits does not lead to a higher revenue for any of the airlines if the other one implements it equilibrium strategy. More generally, in an NE no strategy s_i^a can yield player a a higher utility than the equilibrium strategy s_{NE}^a given that the competitor plays his equilibrium strategy s_{NE}^{-a}. This makes a deviation from the equilibrium strategy unprofitable. See also e.g. Fudenberg and Tirole (1991, p. 11). Introduced by Nash (1950), the NE has become the central solution concept in non-cooperative game theory. The algorithms presented in the later chapters were developed to find NE in RM games. For the players behaving according to NE, rationality as defined above is an important aspect (see also Holler and Illing, 2009, Sec. 3.3.3). Particularly, if both players have all of a game's relevant information and both are aware of this fact, they are able to find their equilibrium strategies by a simple "what if...?" analysis. In other words, the players form expectations about each other's behavior and in an NE the expectations are fulfilled.

Since the two NE (s_2^1, s_3^2) and (s_3^1, s_2^2) in the game from Table 4.1 are made up of only pure strategies, they are called *pure NE*. The game also has a so-called *mixed NE* in which both players play their pure strategies with a probability of 50% each. In the mixed NE each player receives an expected utility of $2 * 0.5 * 0.5 * 129 + 2 * 0.5 * 0.5 * 127 = 128$ (the certain utilities multiplied by the players' probabilities of playing the respective strategies). Note about mixed NE that all of a player's pure strategies which are played with positive probability in the mixed NE must yield him the same expected payoff, i.e. he does not have a profitable pure strategy to deviate to. Even more, a mixed strategy played by one player leads to the opponent being indifferent about his pure strategies played with positive probability in the NE (see e.g. Myerson, 2001, p. 99). This randomization serves to confuse the opponent and not to have one's move anticipated by him (von Neumann and Morgenstern, 1953, p. 146). Also, the ability of players to randomize over their strategies leads to the strategy sets being continuous instead of discrete and enabled Nash (1951) to prove that every finite game has at least one equilibrium point, in mixed strategies at the least.

For airlines competing over costumers in practice, mixed NE may not be desirable outcomes since they still leave room for uncertainty. Hence, pure NE are the desirable solution concept for the games considered in the later chapters of this thesis. While the existence of pure NE can be proved (see e.g. Holler and Illing, 2009, Sec. 3.3.4 and Osborne and Rubinstein, 1995, Sec. 2.4), such conditions might be sufficient but not necessary. I.e. even if they are not satisfied, a game might still possess a pure NE (Fudenberg and Tirole, 1991, p. 35). Furthermore, a game can have several (pure) NE. This led to many authors dealing with NE to provide conditions under

which a pure NE exists and even stronger and more restrictive conditions under which this NE is unique to rule out coordination issues. Among them were Gao et al. (2010), Li et al. (2008) as well as Netessine and Shumsky (2005) for RM games. However, these conditions do not hold in general and a slight change in data might change the game's properties and hence make the prove and possible algorithms connected to it useless. On top of it all, games of interest might be too complex to set up the conditions of existence and uniqueness of (pure) NE in them. Hence, it this thesis algorithms will be introduced to compute NE in general RM games and rules are provided to choose best responses to avoid coordination issues connected to multiple NE. This way, we avoid the complexity of deriving any existence conditions and provide algorithms that do not depend on the form of the game or on the data but can be applied to general RM games.

A closer look at the game from Table 4.1 reveals that both players have only one best response to each of the opponent's pure strategies, i.e. only one highest value in each column of matrix r^1 and in each row of matrix r^2, respectively. Such games are called *non-degenerate*. Consequently, a game in which a player has more than one best response to another player's strategy is called *degenerate* (see also von Stengel, 2007, p. 56). Non-degenerate games have the advantage that one knows with certainty, which best response to a player's strategy the opponent will play (since there is only one). In degenerate games, this statement cannot always be made. To make this clear, consider the above example with a slight change in data. Assume—with all else being as before—that the first player now has demand of $d_l^1 = 64$ and $d_h^1 = 30$. Then the payoff matrices have the form as in Table 4.2.

r^1	s_1^2	s_2^2	s_3^2
s_1^1	120	120	122
s_2^1	125	125	126
s_3^1	130	126	124

a) Payoff Matrix r^1

r^2	s_1^2	s_2^2	s_3^2
s_1^1	122	127	130
s_2^1	122	127	127
s_3^1	122	127	127

b) Payoff Matrix r^2

Table 4.2: Payoff Matrices for the Degenerated One-Leg, Two-Class Game

Now player 2 has two best responses (s_2^2 and s_3^2) each to player 1's strategies s_2^1 and s_3^1 with a payoff of 127 each. With this payoff structure, it is difficult to make a prediction about how player 2 will respond to the concerned strategies of player 1 and find a corresponding best response for him. With the number of strategies rising in a game, the probability of a player having several best responses to the opponent's strategies grows and leads to coordination issues. Likewise, with increasing strategy sets, the number of NE and with it the players' uncertainty about which NE will constitute the game's final solution tendentially increases as well. For, by definition, an NE says nothing about which of a game's solutions will eventually be implemented if several exist. Such coordination issues will also play a role in the later chapters and we will provide a means of resolving them in Section 5.3 by providing a way of choosing unique best responses. This way both players will know which best response the opponent will choose and play strategies that correspond to the same NE in the end, even if several NE exist.

On the other hand, some games do not possess pure NE at all. The game shown in Table 4.3 shows the players' payoff matrices in a small network game without a pure NE from Section 5.4. For such games mixed NE constitute the only solutions. A game possessing no pure NE leads to a similar issue like when a pure NE cannot be found within an acceptable time which will be the case sometimes in our computational studies in later chapters. For this reason, we introduce

a heuristic for computing approximate solutions in RM games in Section 5.4.

r^1	s_1^2	s_2^2
s_1^1	130	180
s_2^1	170	110

a) Payoff Matrix r^1

r^2	s_1^2	s_2^2
s_1^1	170	160
s_2^1	110	160

b) Payoff Matrix r^2

Table 4.3: Network RM Bimatrix Game without Pure NE

An important factor influencing a player's strategy set, which has been already assumed, is the set of information he has available when taking an action. For, the set of available actions available to a player might vary with the state of the game (e.g. a point in time) a player is in and on the action taken by the other player before. In a game with *complete information*, all players have all relevant information about the game, i.e. the players involved, the strategies available to them, and their utility functions. Furthermore, this information is *common knowledge* in the sense that all players know about it and all players know that everybody knows about it and so forth. See Osborne and Rubinstein (1995, Sec. 5.2) for a formal definition. In a game with *incomplete information* one (or all) player(s) lack some of the information concerning the opponent(s), i.e. they either do not know all of their strategies or their utility functions. In a game of *perfect information*, on the other hand, a player knows all the moves made before his decision and he knows which state of the game he is in, while in case of *imperfect information* some of this knowledge is lacked. This aspect is especially important in dynamic and/or repeated games in which an action and/or a game's outcome influence the players' choices in a later game or a state of it. This represents the uncertainty under which the players must make their decisions.

Games with a single simultaneous interaction, as they will be assumed in our analyses of competition below, assume imperfect information. This need not mean that the players indeed must decide which action to take at the same point in time. The simultaneity merely refers to the fact that both players learn simultaneously the decisions made by both players and the realized utilities. See also Osborne and Rubinstein (1995, p. 13). For our applications this means that both competitors have to decide on the strategy to implement without knowing what the other one chose and they only know after takeoff the final outcome of the game.

4.2 Algorithmic Game Theory

As the above definition of a game made clear, a game only describes the conditions under which the players interact, the situation. A game does not, however, include a prescription of how the involved players will behave (i.e. which strategies they will play) or which utility each player will receive after the game is over. In order to get to know these information, provided by NE, the game must be solved.

By definition, an NE is only a steady state (i.e. a solution) of a game but it does not contain any information of how this state can be reached. For this, algorithms need to be applied to reach (one of) a game's solution(s) in accordance with its description of the situation. In part out of this necessity the fairly young science of algorithmic game theory (AGT) arose as an interdisciplinary combination of computer science and game theory. Algorithmic game theorists provide algorithms for computing NE, investigate the (computational) complexity connected

with that and develop ways of approximating NE. See Nisan et al. (2007) for a collection of introductory texts on algorithmic game theory. Roughgarden (2008) presented a summary of algorithmic game theoretic concepts and a future outlook on the topic. Mavronicolas et al. (2008) gathered a series of applications of algorithmic game theory. Vöcking (2013) is a collection of recent publications on the field.

One simple way of finding an NE is given by eliminating all of the players' dominated strategies. See Knuth et al. (1988) for an efficient algorithm. If afterwards every player has only one strategy left available, these strategies make up the NE. Caution must be used when eliminating weakly dominated strategies, though. In general, the strategies left available to the players after eliminating weakly dominated strategies (and hence the residual game) may differ, depending on the order in which the weakly dominated strategies are eliminated (see Myerson, 2001, p. 60 as well as Osborne and Rubinstein, 1995, p. 63). As a result, one may receive different NE after eliminating weakly dominated strategies since they can be best responses. Such an issue does not arise when eliminating strongly dominated strategies since they can never be a best response. Hence, eliminating only strictly dominated strategies yields unique residual games while eliminating weakly dominated strategies need not. See also Gilboa et al. (1990). Marx and Swinkels (1997) showed cases in which the order of elimination of weakly dominated strategies does not matter.

Another standard way to find an NE is to find fixed points (intersections) of the players' reaction functions. A player's *reaction function* (or *best-response function*) is the first derivative of his utility function and assigns a best response for this player for every of the competitor's strategies (Osborne and Rubinstein, 1995, p. 15 and Holler and Illing, 2009, p. 56). One procedure for this needs a system of equations (one for each player) to be solved in order to find the weights (probabilities) to be assigned to the players' pure strategies. This approach is based on the assumption that all of a player's (mixed) strategies must yield him an equal expected payoff given his and the competitor's probability distributions. However, this approach might not be very efficient with a growing number of players and strategies involved, see Osborne and Rubinstein (1995, p. 15). A more elegant way to reach the fixed point is to compute best responses iteratively taking the latest best response of the competitor as given. This *tâtonnement process* is described in e.g. Fudenberg and Tirole (1991, Sec. 1.2) and dates back to Cournot (1838) who used it to describe an adjustment of best responses over time, i.e. after observing the competitor's latest move.

However, both procedures—elimination of dominated strategy and seeking intersections of best response functions—do not guarantee finding an NE. After eliminating the dominated strategies, often more than one strategy still remains available for each player. For finding an intersection of reaction functions (which are the first derivatives of the players' utility functions), the players' utility functions must be differentiable at the least (see Fudenberg and Tirole, 1991, p. 14 for further conditions). However, this is not always the case which implies that one might not even be able to derive the reaction functions in the first place. Thus, if iterated elimination of (strictly) dominated strategies does not leave only one strategy per player or if the reaction functions cannot be set up or their fixed point(s) cannot be isolated, general algorithms are needed to find an NE.

This is exactly what algorithmic game theory is about, among other things. For this, the general setup of bimatrix games is often used in order to be able to compare algorithms. Some of the first publications presenting so-called constructive proofs and hence algorithms for finding NE in bimatrix games were Vorob'ev (1958), whose approach was simplified by Kuhn (1961)

and further by Mangasarian (1964). The latter author was also one of the first to program his algorithm on a computer and to show computational results, a common practice today. These publications presented algorithms for finding all equilibrium points in bimatrix games and built the basis for algorithmic game theory (although at that time nobody could imagine which dimensions this line of research would take on). See also Borm and Peters (2002, Ch. 6) for a comparison of these and further early approaches.

We will follow this branch of AGT in Section 5.3 of this thesis and develop an algorithm for computing pure NE in two-player RM games. Using linear programs to compute best responses for the players, our approach is somewhat similar to Audet et al. (2001), Audet et al. (2006) as well as Sandholm et al. (2005). However, their models have a major drawback since they were developed for bimatrix games and as such rely on the players' complete payoff matrices as input. Setting these up, on the other hand, requires quite a lot of time for the large, realistic-sized games considered in our cases. We will hence avoid setting up the complete strategy and payoff matrices and compute the best responses with linear programs which yield the corresponding strategies and payoffs as their optimal solutions.

Another branch of AGT deals with examining the (computational) complexity of problems related to the computation and enumeration of NE. These complexity results kicked off research about approximate NE. We will touch on this branch of AGT in Section 5.4 and develop a heuristic to compute approximate NE in RM games in case an exact (pure) NE as defined in Section 4.1 cannot be found with our algorithm from Section 5.3. However, we will define our own notion of approximate NE which differs from the concept used in the mentioned literature.

4.3 Computing All Extreme Equilibria

Our heuristic from Section 5.4 is flexible in such a sense that if it cannot find a pure NE, it computes an approximate one in order to have a solution to a game at all. The computation of approximate NE will rely on bimatrix games and will choose the approximate NE of the original game from the set of NE in the bimatrix game. To understand how the set of a bimatrix game's NE can be found, this section will explain and illustrate the algorithm for enumerating all so-called extreme NE in bimatrix games by Audet et al. (2001). For this, consider again the game from Table 4.1 which in re-stated in Table 4.4. For illustrating the necessary concepts, we will reduce the game, though.

r^1	s_1^2	s_2^2	s_3^2
s_1^1	122	122	124
s_2^1	127	127	129
s_3^1	130	129	127

a) Payoff Matrix r^1

r^2	s_1^2	s_2^2	s_3^2
s_1^1	122	127	130
s_2^1	122	127	129
s_3^1	124	129	127

b) Payoff Matrix r^2

Table 4.4: Payoff Matrices for the One-Leg, Two-Class Game

Note that the strategy s_1^a leads to strictly lower revenues for both players than the other two strategies. Hence, no matter, which of his three strategies player $-a$ plays in this example, it is never optimal for the competitor to respond with strategy s_1^a. This strategy is thus *strongly dominated*. On the other hand, a player's strategy s_i^a is weakly dominated by another (*dominant*) strategy s_j^a if s_i^a never yields a higher payoff, given any of the competitor's strategies, but

in at least one case each, s_i^a yields an equal payoff and a lower payoff than s_j^a, respectively. See also Osborne and Rubinstein (1995, Secs. 4.2 – 4.3) and Myerson (2001, Sec. 2.5). Since strictly dominated strategies can never be best responses, they can never be included in NE and hence can be excluded from consideration. The corresponding row(s) and column(s) can then be removed from the payoff matrices. Table 4.5 shows the above game with reduced matrices.

r^1	s_2^2	s_3^2
s_2^1	127	129
s_3^1	129	127

a) Payoff Matrix r^1

r^2	s_2^2	s_3^2
s_2^1	127	129
s_3^1	129	127

b) Payoff Matrix r^2

Table 4.5: Reduced Payoff Matrices for the One-Leg, Two-Class Game

We can also perform a reduction of the payoffs which will be useful later. As e.g. Kontogiannis et al. (2009, Sec. 2.4) have proved, the structure of a bimatrix game, in particular the set of NE therein, does not change if a constant is added or multiplied to all entries of both matrices. Hence, we can subtract 126 from all entries in the reduced matrices from Table 4.5 and receive the normalized matrices from Table 4.6.

r^1	s_2^2	s_3^2
s_2^1	1	3
s_3^1	3	1

a) Payoff Matrix r^1

r^2	s_2^2	s_3^2
s_2^1	1	3
s_3^1	3	1

b) Payoff Matrix r^2

Table 4.6: Normalized Payoff Matrices for the One-Leg, Two-Class Game

Define the vectors $\mathbf{x}^1$ and $\mathbf{x}^2$ as entailing the probabilities player 1 and player 2 assign to their (remaining) strategies, respectively. These strategies form the vectors $\mathbf{m}^1$ and $\mathbf{m}^2$, which both have only two elements in this case. Then, player 1 can find a best response (optimal values for the vector $\mathbf{x}^1$) to one of player 2's strategies (values for the vector $\mathbf{x}^2$) if he solves the linear program LP^1 (4.1) – (4.3) optimally. Likewise, player 2 can solve LP^2 (4.4) – (4.6) optimally to find a best response to a strategy of player 1.

$$LP^1: \max \sum_{i \in \mathbf{m}^1} \sum_{j \in \mathbf{m}^2} x_i^1 r_{ij}^1 x_j^2 \tag{4.1}$$

$$\text{subject to} \sum_{i \in \mathbf{m}^1} x_i^1 = 1 \tag{4.2}$$

$$x_i^1 \geq 0 \qquad i \in \mathbf{m}^1 \tag{4.3}$$

$$LP^2: \max \sum_{i \in \mathbf{m}^1} \sum_{j \in \mathbf{m}^2} x_i^1 r_{ij}^2 x_j^2 \tag{4.4}$$

$$\text{subject to} \sum_{j \in \mathbf{m}^2} x_j^2 = 1 \tag{4.5}$$

$$x_j^2 \geq 0 \qquad j \in \mathbf{m}^2 \tag{4.6}$$

In the objective functions (4.1) and (4.4) the expected payoff is maximized for player 1 and player 2, respectively. Constraints (4.2) and (4.5) require the weights assigned to the players' pure strategies to sum up to 1 while in Constraints (4.3) and (4.6) none of the weights is to be negative. With $\mathbf{x}^1$ ($\mathbf{x}^2$) being a variable in LP^1 (LP^2) and a parameter in LP^2 (LP^1), the two models have to be solved simultaneously to find an NE so that $\mathbf{x}^{1*}$ is an optimal solution to LP^1 while at the same time $\mathbf{x}^{2*}$ is an optimal solution to LP^2 (Audet et al., 2001, p. 323). Moreover, the NE computed in such a way are called *extreme NE* which cannot be expressed as convex combinations of other NE (Avis et al., 2010, p. 18).

With the players having up to three strategies, we can define extreme NE graphically, see von Stengel, 2007, Sec. 3.3 for an exact formal approach. For our example with only two strategies per player, we are able to use a two-dimensional graph with one dimension, a scale between 0 and 1, corresponding to the probability of a player playing the strategy. The other dimension stands for the expected payoff the opponent receives for playing a best response to the corresponding strategy of player 1. The graph is illustrated in Figure 4.1. We used the probability x_1^1 of player 1 playing the first of his remaining strategies in the normalized game of Table 4.6, strategy s_2^1. The probability x_2^1 is then uniquely determined as $x_2^1 = 1 - x_1^1$. Next, we draw the expected payoff curves for the strategies of player 2 in case of player 1 assigning x_1^1 the probability on the scale. For a probability of $0 \leq x_1^1 < 0.5$, the best response for player 2 is his strategy s_2^2 which yields him a payoff of $2 < r^2 \leq 3$. With player 1 playing his strategy s_2^1 with a probability $0.5 < x_1^1 \leq 1$, the best response for player 2 is his strategy s_3^2 with an expected payoff of $2 < r^2 \leq 3$. At $x_1^1 = 0.5$, player 2 is indifferent between his pure strategies and will respond with a mixed strategy as well with a payoff of 2. The best-response curves for player 2 are constructed analogously. The graph has the same form since the players' relevant payoff matrices are identical.

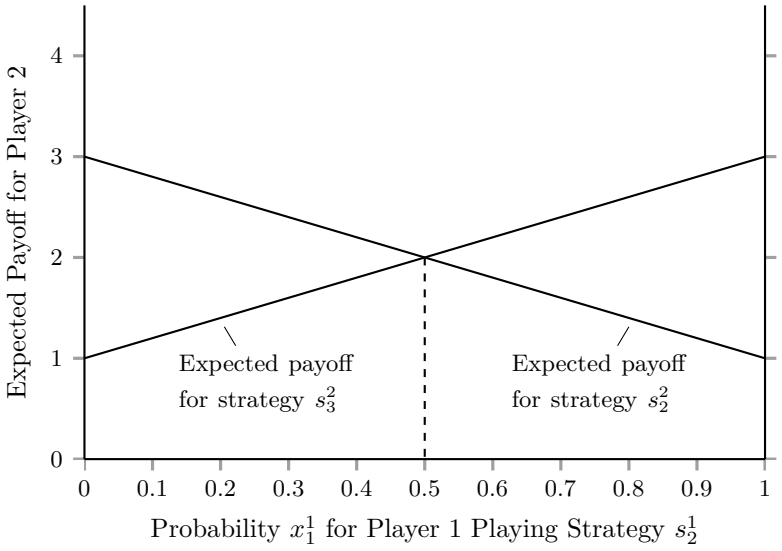
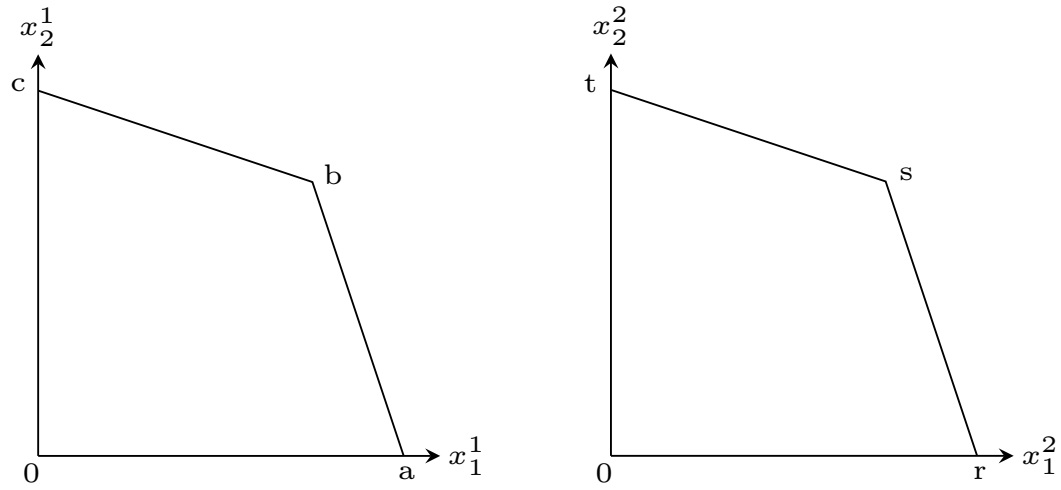


Figure 4.1: Best-Response Graph for Player 1

Using the higher one of the best-response curves for player 2 at any point along the 0–1 scale, we can determine the *strategy polytope* for player 1 a "projective transformation" (see von Stengel, 2002, Sec. 2.4). Figure 4.2 shows the players' polytopes for the game from Table 4.6. Note the axes merely serve illustrative purposes and do not have a linear scale as a consequence of the projective transformation. This means e.g. that the middle of the segment 0–a in Figure 4.2a) does not mean that at this point, $x_1^1 = 0.5$.

To find extreme equilibria based on the players' strategy polytopes, one must find pairs of vertices, one from each polytope, in which the corresponding strategies constitute mutual best responses (see Shapley, 1974 as well as von Stengel, 2002, 2007). As can be seen from Figure 4.2, pure NE lie on the axes since here only one of the pure strategies is played with positive probability (i.e. 100%). The pure equilibria from our example lie in the vertex pairs (a,t) and (c,r) where the players play only their pure strategies (s_2^1, s_3^2) and (s_3^2, s_2^2), respectively. The mixed NE lies in the vertex pair (b,s) in which both players play both of their strategies with a probability of 50% each. Recall that this is the probability needed to make the opponent



a) Polytope for Player 1 b) Polytope for Player 2

Figure 4.2: Players' Polytopes for the Simple Revenue Management Game

indifferent about his pure strategies, as the polyhedron in Figure 4.1 showed.

The special feature of extreme NE is that knowing all of them suffices to quantify all NE of a game (Audet et al., 2001, p. 324). If in an NE one player's pure strategy has two best responses on the side of the opponent, every convex combination of these best responses is an NE as well. This, on the other hand, means that there are possibly infinitely many NE in a (degenerated) game. In order to quantify all NE of a game, it suffices to find the extreme points of these convex combinations, i.e. all extreme NE in which the corresponding player plays only one or the other best response. This is also true for the degenerate game from Table 4.2. Here, the pure NE (s_2^1, s_3^2) and (s_3^2, s_2^2) are extreme NE which can be used to determine further NE of the game. With player 1 playing one of his pure strategies, any probability combination of the pure strategies of player 2 leads to an NE. Of these NE, the probability combination $x_1^2 = \frac{2}{3}$ and $x_2^2 = \frac{1}{3}$ leads to a mixed extreme NE with one of player 1's pure strategies.

Having defined extreme equilibria, we will next describe the algorithm by Audet et al. (2001) and apply it to find all extreme NE in the bimatrix game from Table 4.6. For this, the dual programs of LP^1 and LP^2 are set up first. These are model DP^1 (4.7) – (4.8) and model DP^2 (4.9) – (4.10). Here, σ^1 and σ^2 stand for the dual variables. These are to be minimized in the objective functions (4.7) and (4.9), respectively. According to Constraints (4.8) and (4.10), respectively, the dual variables must amount to the players' maximal expected payoff given the opponent's strategy.

$$DP^1: \min \ \sigma^1 \tag{4.7}$$

$$\text{subject to} \sum_{j \in \mathbf{m}^2} r_{ij}^1 x_j^2 \leq \sigma^1 \quad i \in \mathbf{m}^1 \tag{4.8}$$

$$DP^2: \min \ \sigma^2 \tag{4.9}$$

$$\text{subject to} \sum_{i \in \mathbf{m}^1} r_{ij}^2 x_i^1 \leq \sigma^2 \quad j \in \mathbf{m}^2 \tag{4.10}$$

One can see that the players' dual problems depend on the respective opponent's best response only and do not entail the respective player's own best response. Hence, the primal problems LP^1 and LP^2 are merged with the opponent's dual problem in order to have a player's variables in one model. The resulting models are model $\widetilde{LP^1}$ (4.11) – (4.14) for player 1 and

model $\widetilde{LP}^2$ (4.15) – (4.18) for player 2. Note that Mangasarian and Stone (1964) proposed a single model with linear constraints and a quadratic objective function which is a combined version of the models $\widetilde{LP}^1$ and $\widetilde{LP}^2$.

$$\widetilde{LP}^1: \max \sum_{i\in\mathbf{m}^1}\sum_{j\in\mathbf{m}^2} x_i^1 r_{ij}^1 x_j^2 - \sigma^2 \quad (4.11)$$

$$\text{subject to} \sum_{i\in\mathbf{m}^1} x_i^1 = 1 \quad (4.12)$$

$$\sum_{i\in\mathbf{m}^1} r_{ij}^2 x_i^1 \le \sigma^2 \quad j\in\mathbf{m}^2 \quad (4.13)$$

$$x_i^1 \ge 0 \quad i\in\mathbf{m}^1 \quad (4.14)$$

$$\widetilde{LP}^2: \max \sum_{i\in\mathbf{m}^1}\sum_{j\in\mathbf{m}^2} x_i^1 r_{ij}^2 x_j^2 - \sigma^1 \quad (4.15)$$

$$\text{subject to} \sum_{j\in\mathbf{m}^2} x_j^2 = 1 \quad (4.16)$$

$$\sum_{j\in\mathbf{m}^2} r_{ij}^1 x_j^2 \le \sigma^1 \quad i\in\mathbf{m}^1 \quad (4.17)$$

$$x_j^2 \ge 0 \quad j\in\mathbf{m}^2 \quad (4.18)$$

In the algorithm of Audet et al. (2001), a search tree is set up where in each node both combined models $\widetilde{LP}^1$ and $\widetilde{LP}^2$ are solved successively while certain constraints are added to one of the models. In particular, two branching rules are used. In the *dual branching*, two child nodes are added to every parent node while in the *plural branching* the number of child nodes added to a parent node amounts to the total number of variables. The latter branching rule makes sure that all extreme NE are found in degenerate games.

In the first node, arbitrary values for a player's probability weights $\mathbf{x}^a$ are picked and plugged in the opponent's model to seek a best response for him. Audet et al. (2001) suggested assigning weights $\frac{1}{|\mathbf{m}^1|}$ to player 1's strategies. With this information, first model $\widetilde{LP}^2$ is solved and afterwards model $\widetilde{LP}^1$ is solved with the best response of player 2. Depending on the models' solutions, two new nodes are added to the search tree which again entail both players' models one of which is extended. In one node, a player's model is extended so that a certain of his pure strategies is forced to be played with probability 0 (the other player's model is the same as in the parent node). In the other child node, the other player's model is extended so that one of his strategies is a best response (the first player's model remains the same as in the parent node). For this, a Restriction (4.13) or (4.17) is turned into an equality restriction. Hence, the algorithm implements an iterative search for best responses in every node and the branching guides the search towards different NE.

For determining the additional constraints, auxiliary variables $\boldsymbol{\rho}^a$ are defined for the players' strategies. For player 1, $\rho_i^1 = -1$ if his variable x_i^1 is forced to zero or the corresponding Constraint (4.13) is turned into an equality constraint. Otherwise, it is set to $\rho_i^1 = x_i^1(\sigma^1 - \sum_{j\in\mathbf{m}^2} r_{ij}^1 x_j^2)$. The variables ρ_j^2 for player 2 are assigned analogously. Out of all of a player's auxiliary variables, the one with the largest value is compared to the opponent's largest auxiliary variable (when two variables have the same value, one is picked randomly). Call these largest variables ρ_{i*}^1 and ρ_{j*}^2.

If $\rho_{i*}^1 \ge \rho_{j*}^2$, the models in the child nodes are extended so that in one node the constraint $x_i^1 = 0$ is added to player 1's model. In the other node, the model of player 2 is extended by constraint $\sum_{j\in\mathbf{m}^2} r_{ij}^1 x_j^2 = \sigma^1$ which forces player 2's strategy to be a best response to strategy x_i^1 of player 1. If $\rho_{i*}^1 < \rho_{j*}^2$, constraint $x_j^2 = 0$ is added to player 2's model in one new node and constraint $\sum_{i\in\mathbf{m}^1} r_{ij}^2 x_i^1 = \sigma^2$ is added to player 1's model in the other node. Every branching level adds to the search tree's depth, which amounts to 1 in node 1. At depth $|\mathbf{m}^1|+|\mathbf{m}^2|+1$, all of a non-degenerated game's extreme NE have been found. The plural branching is applied only at this depth and deeper to find all of a degenerated game's extreme NE as well. In particular,

with the constraints of this branching type, it is checked whether a player's strategy is a best response and played with probability 0 at the same time which is only possible in degenerated games. One can check this in the degenerated game from Table 4.2. If e.g. player 1 plays strategy s_3^1, both strategies s_2^2 and s_3^2 are best responses for player 2. If one of them is not allowed to be played, the model $\widetilde{LP}^1$ still has a feasible (and thus optimal) solution which is not the case in non-degenerated games.

In the new nodes, first the extended model is solved optimally with the opponent's strategy from the parent node and then the opponent's model is solved optimally with the new best response of the extended model. If the extended model (and thus also the other one) can be solved optimally, the search tree is augmented by further nodes. In this case, the new constraints remain in the model until the termination of the algorithm. However, if the extended model in a new node cannot be solved optimally (e.g. because all of a player's strategies are forced to be played with zero probability), the respective node is not considered any further. In this case, the constraint causing the infeasibility is removed from the model again and a node not yet considered is examined. When there are no more nodes in which a model can be checked, the algorithm stops. According to Audet et al. (2001, p. 329), the order in which the nodes are examined does not matter, since all possibilities for extending the models must be checked. The authors implemented a depth-first search, though which is used here as well. Figure 4.3 shows the complete search tree for the non-degenerate game from Table 4.6. Here, the constraints added to the models are written next to or below the edges leading to the respective nodes.

In node 1 we set $x_1^1 = x_2^1 = 0.5$ and plugged these values into the model $\widetilde{LP}^2$ for player 2. The best response for player 2 was $x_1^2 = x_2^2 = 0.5$ which was used as input for $\widetilde{LP}^1$, the model for player 1. In this node, all values for ρ^1 and ρ^2 amounted to 0 leading to two new nodes 1 and 13 with new constraints $x_1^1 = 0$ in the former and $\sum_{j \in \mathbf{m}^2} r_{1j}^2 x_j^2 = \sigma^1$ in the latter. Here, only ρ_2^1 needed to be considered together with ρ_1^2 and ρ_2^2 because the constraint $x_1^1 = 0$ has been added to the model of player 1. The extreme NE were found in nodes 7, 17, and 27, respectively (marked bold) with the first two NE being the two pure NE and the last one being the mixed NE. These can be tracked along the tree's branches following the respective edges. Since e.g. the constraint $x_1^1 = 0$ added in node 2 leads to a feasible solution there, it is clear that in the end of this branch a pure NE must be found with $x_2^1 = 1$ since both players only have two strategies. The pure NE in node 17 is found analogously with $x_2^1 = 0$ and $x_1^2 = 0$ in nodes 14 and 15, respectively. Since the leftmost branch 1–13–23–25–27 forces all of the players strategies to be best responses, node 27 leads to the mixed NE with probabilities of $x_1^1 = x_2^1 = x_1^2 = x_2^2 = 0.5$.

Note that in the course of the algorithm, it is possible to find extreme NE before the tree reaches depth $|\mathbf{m}^1| + |\mathbf{m}^2| + 1$. E.g. all best responses in the branch leading to the mixed NE in node 27 had the same probabilities assigned to the players' strategies. However, the extreme NE are only saved at depth $|\mathbf{m}^1| + |\mathbf{m}^2| + 1$ in order to reduce the effort of checking for equilibrium conditions at every node. In nodes 3, 5, 12, 16, 22, 24, and 26 the branching is not continued because the additional constraints in the respective node lead to infeasible models. E.g. in node 3, both of player 1's strategies are forced to be played with zero probabilities. Since the players have only two strategies each, the constraint (4.12), $\sum_{i \in \mathbf{m}^1} x_i^1 = 1$ cannot hold. In like manner, the plural branching leads to infeasibilities after the extreme NE were found.

Since its introduction, the algorithm by Audet et al. (2001) just described and applied has been subject to extensions and improvements. Audet et al. (2006) extended the algorithm to compute NE in bimatrix and *polymatrix* games, i.e. games with ≥ 3 players. For the latter, not only one but several (one "principal" and further "secondary" trees) were used. Avis et al. (2010)

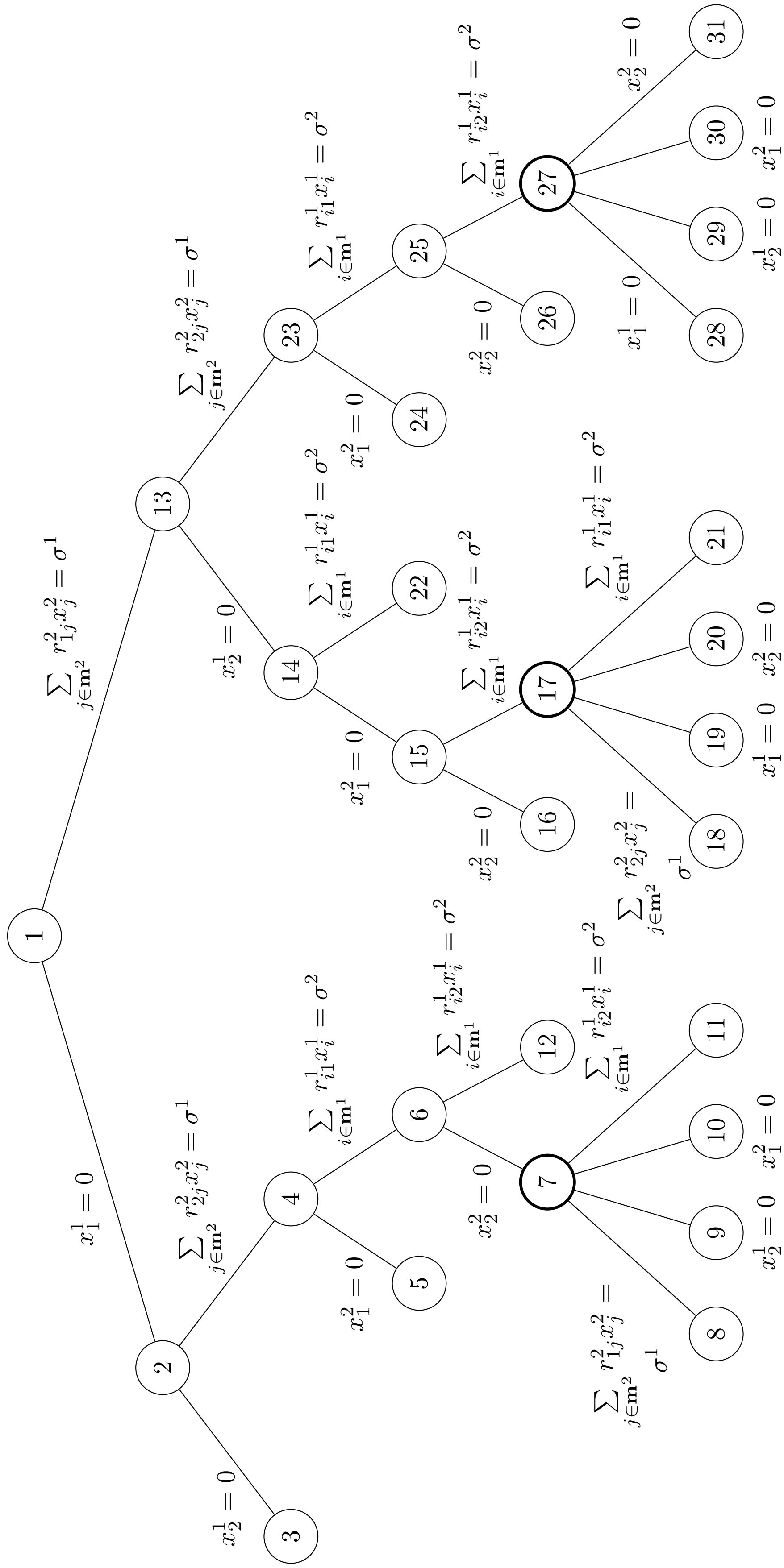


Figure 4.3: Search Tree for the Non-Degenerate Two-Leg Game

presented an improved degeneracy check which leads to a reduction of the search tree. E.g. the plural branching is not necessary in non-degenerated games with the approach presented there. A similar algorithm as the one presented in this section is implemented in the Gambit software by McKelvey et al. (2014) for computing NE in bimatrix games. This software will be mentioned in Section 5.4 when our heuristic for computing an approximate NE in RM games is presented.

4.4 Assumptions for the Further Course

As the topic of GT offers a wide range of tools and concepts for analyzing players' behavior, we must restrict ourselves to those concepts that are relevant and best suited for our investigations. Here we summarize the game theoretic assumptions which will be made for the remainder of this thesis. Although some of the assumptions might seem redundant and have already been stated, they are still included here for the sake of completeness. In particular, we will assume the following:

The games will be played by two airlines as players.

We will assume only two competing airlines as our players in the competitive situations below. Although airlines certainly compete with several others simultaneously, we will concentrate on this simple case for analyzing the competition in this thesis to provide a first grasp with the proposed models and algorithms. However, we will describe how to extend the models and algorithms in such a way that more than two competitors are considered.

The players' utility functions will be their respective total revenue functions.

It will be assumed that the players are interested only in maximizing their own total revenues modeled in their objective functions. Although we will assume alliance involvement in Chapter 7, every airline will be assumed to maximize only its own revenue, not e.g. the alliance revenue.

The players' pure strategies will be their vectors of booking limits.

The use of booking limits as decision variables was motivated in Section 2.4. Here, it is important to point out that one product's booking limit is only one action. A player's complete pure strategy is a vector quantifying the booking limits for all of his products. Since there is only a finite number of possible booking limit combinations (limited by the capacities and/or by the product demands), we will have finite games.

The players will interact in games of complete information.

The players will be assumed to have all relevant information about each other and to know about this fact. Although this is a somewhat unrealistic assumption, it simplifies analysis and allows for concentration on the core problems. The same assumption was also made in e.g. Jiang and Pang (2011), Li et al. (2008) as well as Netessine and Shumsky (2005). An airline e.g. can conclude from the type of aircraft employed by the competitor the number of seats available on its plane and buy relevant data about the competitor's demand from research institutes or consulting companies. Prices are public information available on the internet. An airline can then deduce the competitor's vector of booking limits (information that is needed for our models below) by solving a model optimally which is provided with the competitor's data.

The players will interact in games of imperfect information.

We will assume single, simultaneous interactions in which the players do not get to know which

strategy their opponent played until the game is over. Hence, the players must take guesses and perform "what if...?" analyses about each other's behaviors to determine their own strategies. Furthermore, the players will be assumed to set their competitive booking limits only once for the whole booking horizon and not to update these as the time of departure approaches. I.e. we will not consider repeated or dynamic games.

The desired solutions for the games will be made up of pure Nash equilibria.

The main goal of the algorithms to be presented in the later chapters is to find pure Nash equilibria. Pure NE are motivated by the fact that they can directly be interpreted and implemented in RM games. Further, since a game might possess several (pure) NE, our approaches will be set up so as to find a distinct NE in order to avoid coordination issues. Mixed equilibria will be computed only as a compromise in case a pure NE cannot be found in order to present an approximate solution which is tendentially still better than the non-competitive one.

Chapter 5

Airline Network Capacity Control under Competition

After having defined the basic concepts from revenue management, strategic airlines alliances, and game theory, in Section 5.2 we will introduce the fundamental competitive model that will also serve for modeling the competition in the later chapters. Afterwards, Section 5.3 will introduce the core algorithm for computing exact pure NE for two airlines applying network RM under competition. We will also illustrate the algorithm's functioning with a numerical example and provide an extension to it to generate unique searching paths in order to avoid coordination issues connected to the multiplicity of optimal solutions and NE. Since this algorithm might terminate without an NE, we will propose a heuristic to approximate an NE in RM network games in Section 5.4 along with a definition of our own notion of approximate NE after showing that network RM games might not posses pure NE. Computational studies are presented to illustrate the algorithms' performances in Sections 5.3.5 and 5.4.6, respectively. Before we come to the competitive model and the algorithms, though, we will give an overview over relevant literature in Section 5.1. All in all, this chapter is based on work described in Grauberger and Kimms (2014b) and in Grauberger and Kimms (2014c).

Linear programs were used before for computing NE. The first such approach was made by Dantzig (1951) for *zero-sum games*. These are games in which a player gains what the other one looses. Hence, the sum of the players' gains and losses is 0. The author has shown that an NE in zero-sum games can be found by solving a single linear program which represents a player's actual decision problem. This is due to the minmax-theorem, introduced by von Neumann (1928), being closely related to the theory of duality in linear programming (see e.g. Murty, 1983, Sec. 4.6.6 for the latter). RM games are not zero-sum games, though as Wilson (1995) showed and which will become clear later. This makes the approach by Dantzig (1951) inapplicable for our purposes.

In algorithmic game theory, LPs are employed for computing NE in bimatrix games. However, as Section 4.3 showed, the variables therein stand for probabilities assigned to the players' pure strategies. What our approach has in common with such approaches is that we also employ linear program to search for best responses iteratively. In our case, the LPs will be used to determine pure strategies in a first place, though. Hence, our methods present middle courses between computing an NE directly by solving a single model and computing NE via best response based on the complete payoff matrices.

5.1 Related Literature

So far very little research has been done on RM under competition. Li and Oum (1998), Li et al. (2007), Li et al. (2008), Gao et al. (2010) as well as Netessine and Shumsky (2005) analyzed competition for single-leg flights and two fare classes. They all found—among other things—that under competition the booking limit for the lower-fare class was lower compared to the centralized case when the airlines cooperated. They did not, however—except for Gao et al. (2010)—compare the (expected) revenues that could be achieved in these scenarios, which is the more interesting question we address.

Netessine and Shumsky (2005) also considered a kind of network competition. They defined *vertical competition* for two airlines competing over connecting passengers changing planes at a hub. In this setting a passenger flying from an origin O to a destination D over a hub H uses the leg O–H operated by one airline and leg H–D operated by another. If local passengers using only one leg and connecting passengers using both legs exist, the airlines must choose how many seats to protect for local and connecting passengers in absence of cooperation or coordination. *Horizontal competition*, on the other hand, is characterized by both airlines operating substitutable flights which the same origin and destination and where a customer might request a ticket from the competitor if he is denied by his preferred airline. This is the concept we will use in this chapter. Chapter 7 treats simultaneous vertical and horizontal competition for alliance partners.

Jiang and Pang (2011) assumed the airlines to compete in the same network offering identical products, i.e. identical local as well as connecting flights. They formulated two competition models based on the DLP and PNLP. With their models more than two players can be considered. The authors proposed an iterative algorithm to solve the game and showed that it converged for games based on the PNLP "under suitable conditions". This result could not be made for their DLP game because of its special structure. We will explain this structure in more detail in Section 5.4.

Friesz et al. (2007) and Mookherjee and Friesz (2008) described a general stochastic and dynamic service network competition model. The former authors unified the problems of pricing and resource allocation while the latter augmented the model to include overbooking as well. However, the allocation decision did not affect the competitor's revenue in this model, only the pricing decision did (Mookherjee and Friesz, 2008, p. 461). The authors transformed the problems into a differential variational inequality (Friesz et al., 2007) and a "plain" variational inequality (Mookherjee and Friesz, 2008) and solved them using a fixed-point algorithm. Their main finding was that in an oligopoly NE, there is a bias toward pricing too low. Friesz et al. (2007) further found that in the NE there is a bias towards overallocation of resources.

Lim (2009) also investigated the question of overbooking in a competitive environment and showed that it may end up in a prisoner's dilemma. This is a situation in which the players are best off cooperating, however all have the incentive to break cooperation and in the end all end of worse than in the cooperation. Other authors, like Adida and Perakis (2010), Gallego and Hu (2014), Levin et al. (2009), Lin and Sibdari (2009), Liu and Zhang (2013) as well as Martínez-de-Albéniz and Talluri (2011), picked up the problem of dynamic pricing under competition. Levin et al. (2009) as well as Lin and Sibdari (2009) even took into account strategic customers which might anticipate possible price reductions and hence delay a buy which makes the problem more complicated. However, we will not pursue dynamic pricing any further; with or without competition.

5.2 Competitive DLP

Our competitive DLP is a reformulation of the DLP for the network revenue management problem from Section 2.3.2 which we adjust to incorporate competition information. This model was proposed by Grauberger and Kimms (2014b) and also used in Grauberger and Kimms (2014c). Jiang and Pang (2011) proposed a similar, yet simpler, model. However, they made some assumptions which made the interpretation of their results difficult from a game theoretic point of view. We adjust these shortcomings here to picture the problem more realistically.

In particular, Jiang and Pang (2011) let the capacities of the aircraft, the random demand, and the decision variables—the booking limits for the individual products—to be continuous non-negative numbers. Allowing all three values to be continuous, the optimal booking limits can turn out to be continuous as well. However, what is a sufficient approximation for an isolated airline leads to problems in the interpretation of the alleged NE. For, in practice, only integer booking limits can be implemented. While an isolated airline can heuristically round the continuous variables in an optimal solution to integer booking limits, in a game theoretic setting with rationally acting players such rounding shifts the best responses away from the NE if it consists of continuous numbers. This, in turn, leads to different best responses by both players and makes the information of continuous best responses in a NE worthless within an integer setting such as the capacity allocation problem in revenue management. For, in an integer setting the best response functions are not continuous and non-integer best responses do not exist except in mixed strategies.

In order to overcome such failures, the decision variables should be integer-valued in the optimal solution. An advantage of the DLP models we use is that modeling the decision variables as integers is not necessary if (1) the flight network is acyclic which is always the case for time-indexed flight networks used in revenue management, (2) the demand values are integer parameters, (3) the capacity values are integer parameters, and (4) all products only occupy one seat (see also de Boer et al., 2002, p. 77). All of these conditions apply in our model, as we will see and hence we receive only integer values for the variables in the optimal solution. That way the respective solution can be interpreted directly as a pure strategy and one does not have the problem of floating-point numbers as it may arise in the approach described by Jiang and Pang (2011). Note that the problem and the algorithms presented below are based on only two players. However, they can be augmented to incorporate n players as will be shown later.

We next adjust the notation from the single-airline DLP from Section 2.3.2 for the competitive DLP. Consider two airlines, denoted as $a \in \{1, -1\}$. In this context, index a refers to the considered player, while $-a$ refers to the competitor. The sets of O&D pairs and fares offered by both players together are P and F, respectively. As before, a combination of an O&D pair p in a fare class f will be called *product*. The set of flight legs served by both airlines is L. Player a offers $P^a \subseteq P$ O&D pairs (indexed $1, \ldots, \mathcal{P}^a$) on a network with $L^a \subseteq L$ legs (indexed $1, \ldots, \mathcal{L}^a$). Given the network structures from Figure 2.1, which will also be assumed for the players in the competitive models, there is only one way in each airline's network to get from an origin O to a destination D. Hence, the route taken by each airline for an itinerary is unique and it can sufficiently be defined only by means of the origin and the destination without considering possible intermediate stations. Even if the airlines take different routes, all that is contained in a competitor's set P^a is the combination of the origin and destination. This follows in part from the assumption of ignoring customer choice made in Section 2.4. For, having several routes to choose from, a customer might want to consider a different route from from O to D by his

preferred airline before turning to the competitor. While this is an interesting aspect, it is not considered in this thesis.

The O&D pairs for which player a's total demand is affected by competition are contained in the set $A = \{P^a \cap P^{-a}\}$. It entails those itineraries which are offered by both airlines, i.e. a customer can get from a desired origin to a desired destination with any of the competitors. Since we assume only two players, we need no superscript a denoting the player for this set because it would hold that $A^a = A^{-a}$. We further assume that both players offer the same set F of fare classes (indexed $1, \dots, \mathcal{F}$) for all of their O&D pairs. Hence, we also need no superscript a here. O&D pairs not affected by competition make up the set $NA^a = \{P^a \backslash P^{-a}\}$. More precisely, this set contains those itineraries of a player for which the opponent does not have a substitute O&D combination. I.e. either the competitor does not offer any flights from the corresponding origin or he does not offer any flights to the corresponding destination, or both. We will clarify the sets A and NA^a by means of Figure 5.3 below.

$\mathbf{M}^a$ is the incidence matrix for player a's network with $M^a_{lp} = 1$, if leg l is used by O&D pair p and $M^a_{lp} = 0$ otherwise. This means that all products occupy only one seat on the planes they use. The initial (exogenous) demand player a expects to receive for a product during the booking period is d^a_{pf}. This is the only information about demand we need. Particularly, we need not make any assumptions about the order of request arrivals. Player a's total capacity on leg l is C^a_l. His revenue for a product is π^a_{pf}. The proportion of customers requesting a product from player a if they are denied a ticket by their preferred airline $-a$ is $\alpha^{-a,a}_{pf} \in [0, 1]$.

The decision variables for airline a are its booking limits b^a_{pf} for its offered products. All in all, airline a solves model $\mathcal{M}^a$ consisting of equations (5.1) – (5.5) in response to the competitor's behavior:

$$\mathcal{M}^a: \max \quad r^a(b^a_{pf}, b^{-a}_{pf}) = \sum_{p=1}^{\mathcal{P}^a} \sum_{f=1}^{\mathcal{F}} \pi^a_{pf} b^a_{pf} \tag{5.1}$$

$$\text{subject to} \quad b^a_{pf} \leq d^a_{pf} \qquad p \in NA^a, f \in F \tag{5.2}$$

$$b^a_{pf} \leq d^a_{pf} + \lfloor \alpha^{-a,a}_{pf} (d^{-a}_{pf} - b^{-a}_{pf})^+ \rfloor \qquad p \in A, f \in F \tag{5.3}$$

$$\sum_{p=1}^{\mathcal{P}^a} \sum_{f=1}^{\mathcal{F}} M^a_{lp} b^a_{pf} \leq C^a_l \qquad l \in L^a \tag{5.4}$$

$$b^a_{pf} \geq 0 \qquad p \in P^a, f \in F \tag{5.5}$$

The objective function (5.1) maximizes the total revenue of airline a summing over the product prices times their booking limits. The booking limits of those products not affected by competition are not to exceed their original expected demand due to restriction (5.2). Restriction (5.3) limits the booking limits of those products affected by competition to their (rounded down) total demand. Recall that the term $(d^{-a}_{pf} - b^{-a}_{pf})^+$, on the right hand side of Restriction (5.3) stands for the non-negative number of booking requests denied by the competing airline $-a$ (spillover demand of $-a$). This constraint, which is not included in the single-airline DLP from Section 2.3.2 distinguishes it from the competitive DLP ($\mathcal{M}^a$). Since $\alpha^{-a,a}_{pf}(d^{-a}_{pf} - b^{-a}_{pf})^+$ cannot be negative, including competition does not lead to a worse payoff than ignoring it if the model is solved optimally. Moreover, it shows that airline RM games modeled this way are not zero-sum games. This term is rounded down in order to have integer values for the demand. This makes modeling the decision variables as integers unnecessary as was noted above. Restriction

(5.4) is the capacity restriction requiring the sum of all booking limits on a leg not to exceed the airplane's capacity on this leg. In restriction (5.5) the variables b^a_{pf} are assumed to be continuous.

Recall that in game-theoretic terms, an optimal solution to a player's model $\mathcal{M}^a$ constitutes a best response to the competitor's vector of booking limits $\mathbf{b}^{-a}$. Since the solution can be interpreted directly in terms of how to set the booking limits for the individual products, it forms a pure strategy. Hence, the NE computed using these models will constitute pure NE.

If more than two competitors are to be considered in the competitive DLP, Constraint (5.3) will have to be changed to Constraint (5.3').

$$b^a_{pf} \leq d^a_{pf} + \lfloor \sum_{-a \in \{\mathcal{A} \setminus a\}} \alpha^{-a,a}_{pf} (d^{-a}_{pf} - b^{-a}_{pf})^+ \rfloor \qquad p \in A^a, f \in F \tag{5.3'}$$

In particular, the spill over demand will have to sum over all players with $-a$ now standing for all players in set $\mathcal{A}$ except the considered player a. Further, the set of itineraries affected by competition would have to be extended by the index a to A^a since this set might not be identical for all players anymore. Finally, the complete summed spill over demand would have to be rounded down (instead of the individual summands) to have a higher total demand.

5.3 Computing an Exact Pure Nash Equilibrium

To the best of our knowledge efficient algorithms for computing pure NE in general network RM games considered here do not exist. In case of competition within a single leg and two fare classes it is easier because one only has one variable (e.g. the lower fare class booking limit). The difference between the available capacity and this variable's value is reserved for the other booking class. See e.g. Gao et al. (2010), Li et al. (2007), Li et al. (2008) as well as Netessine and Shumsky (2005). There, also conditions for the existence of a pure NE in single-leg, two-class games were presented. However, most of these publications assumed continuous demand distributions which can lead to the same difficulties as the formulation in Jiang and Pang (2011) mentioned in the previous section. Li and Oum (1998) did not specify this detail. Although all the other authors provided conditions under which a pure-strategy NE existed, it did not in general. Further, it is not always clear whether the booking limits were integer-valued in the pure NE. Gao et al. (2010) even presented an example with continuous booking limits in a unique, pure-strategy NE. This, however complicates decision making since continuous booking limits are hard to interpret and to implement in practice, especially in a competitive situation where rounding can lead to a different best response of the competitor. We want to fill these gaps.

For finding pure NE in RM games, one might consider using a modified version of an algorithm which computes NE in bimatrix games. The most prominent algorithm for computing NE in bimatrix games is a pivoting algorithm introduced by Lemke and Howson (1964). This approach (LHA in short) is similar to the simplex algorithm used for solving linear programs (see e.g. Murty, 1983, Ch. 2). Yet, the LHA only searches for a feasible solution which must satisfy certain constraints and does not optimize an objective function as the simplex algorithm does. Further, while the simplex method traverses the vertices of one polytope (the simplex), the LHA travels along the vertices of two polytopes, one per player. The pivoting in their algorithm is in principle an iterative search for best responses along two polytopes, one for each player, which are based on the players' utility matrices. van den Elzen and Talman (1991) provided an alternative algorithm which is able to find all NE in a game. In van den Elzen and Talman (1994)

the authors extended their algorithm to find an NE in *polymatrix* games, i.e. games with more than two players. Howson (1972), Rosenmüller (1971), and Wilson (1971) independently extended the LHA to compute equilibria in polymatrix games. Shapley (1974) illustrated the LHA graphically and thus provided an alternative, graphical approach in addition to an alternative proof that the number of (non-degenerate) NE in bimatrix games is finite and odd.

In Sandholm et al. (2005) mixed-integer programs were used to compute NE in bimatrix games. The authors provided different objective functions for their program for finding NE with certain properties, e.g. highest summed utility. Porter et al. (2008) introduced two algorithms, one for computing an NE in bimatrix games and an extended one for polymatrix games. Their algorithms rely on providing a set of pure strategies for the players and checking whether this set constitutes an NE with a constraint satisfaction program. See also McKelvey and McLennan (1996) who provided an overview of common algorithms for computing only one or all NE in bimatrix and polymatrix games. So did von Stengel (2002, 2007) who reviewed some of the algorithms mentioned above. See von Stengel (2010) for a journal's edited special issue on computing NE.

Although some algorithms were developed for computing pure NE in special games, these are similar but not identical to our application and are not applicable here. Jiang and Leyton-Brown (2007) and Ryan et al. (2010) presented algorithms to compute pure NE in certain symmetric games in which all players have the same strategies available. In our situation this is not the case in general. Here, the players' strategies (booking limits) depend on their demands for the respective products. Since the demand values are not equal in general, our RM games are not symmetric in general. Echenique (2007) developed an algorithm for computing all pure NE in so-called games with strategic complements. In such games, if one player increases his output (e.g. raises his booking limit for a product) it is optimal for the other player to increase it as well. We do not have such a case in our games, so this algorithm is not applicable here either. Panagopoulou and Spirakis (2006) and Harks et al. (2010) presented algorithms for finding pure NE in so-called congestion games. In such games, the players compete for a resource with limited capacities such as a road or an internet network. The cost of each resource depends on the amount of players choosing this resource and the players' choices depend on the cost of each resource. This case would apply in RM games if demand for a product were a scarce good. Since we do not have this type of game here either, we cannot use such an algorithm.

The specialized algorithms for computing pure NE mentioned above cannot be used for our purposes because the games examined there cannot be transferred to RM games. The general algorithms described can be used to find pure NE in our games after they have been converted to bimatrix games. However, many of the algorithms find one (possibly mixed) NE in each run. Rerunning the algorithm with a different starting strategy after a mixed NE is found can raise the probability of finding a pure NE because the new starting strategy might lead to a different search path than before and hence end up in a different NE. However, this "trial-and-error approach" is not very promising because one does not know at the beginning, which type of NE will be found. One could define the weights assigned to players' strategies in the models $\widetilde{LP}^a$ (4.11) – (4.18) as binary variables. If further exactly one strategy is to be chosen by a player, the algorithm by Audet et al. (2001) would find pure NE. However, this approach would put a further burden on the computation time (which is already exponential for continuous weights) since mixed-integer problems would be used.

Indeed, many results connected to algorithmic game theory reveal a high computational complexity when computing or proving the existence of NE. Roughgarden (2010) offered an

introduction of computational complexity in combination with AGT, while Papadimitriou (2007) is more technical. Gilboa and Zemel (1989) showed that proving the existence of an equilibrium with a minimal payoff is NP-hard. Conitzer and Sandholm (2008) demonstrated that proving the existence of further equilibria with certain desirable properties (e.g. pure NE) is NP-complete. Papadimitriou (1994) introduced the complexity class of PPAD (Polynomial Parity Arguments on Directed graphs) as a subclass of NP. The problem of finding NE in bimatrix games is entailed in PPAD, among others. Daskalakis et al. (2006) proved the PPAD-completeness of finding an NE in four-player games. Chen et al. (2009) strengthened this prove and showed that computing NE is complete in PPAD even for two-player (i.e. bimatrix) games which means that computing NE in bimatrix games takes exponential time in general. Hazan and Krauthgamer (2011) showed complexity results about approximate NE. Note however, that although we could represent the games treated in this thesis as bimatrix games, these complexity results need not apply to our games algorithms because we treat (special) games in which linear programs serve for determining best responses.

Due to excessive memory consumption to store the players' complete strategies and payoff matrices we cannot compare our algorithms with the algorithms mentioned based which are based on the players' complete payoff matrices, a further drawback common to these general algorithms. The complete strategy and payoff matrices are very cumbersome to set up for games with large strategy spaces which will be considered in the later chapters. In the application of the general algorithms named above, this burden would have to be considered on top of the computational complexities of the algorithms themselves. A more efficient algorithm to compute pure NE in RM games was developed by Grauberger and Kimms (2014c) and is used in this and later chapters. The acceleration stems from the avoidance of setting up the complete strategy sets and payoff matrices. The exact algorithm is explained in Section 5.3.2. Before we get to this more abstract description, we explain the general concepts here to help understand the individual steps.

Our algorithm is initiated by providing a starting player and an arbitrary starting strategy for the competitor. The players then search for best responses iteratively by solving their model $\mathcal{M}^a$ until a player's best response is chosen a second time. Every time a best response is found, player a's revenue is saved in vector $\mathbf{R}^a$ and his best response (a $\mathcal{P}^a \times \mathcal{F}$ matrix) is saved in vector $\mathbf{B}^a$ if this best response was not chosen before. We call this search *forward search*.

When a best response is chosen a second time, there are two cases to be distinguished. In one case, the best response is the same as in the previous iteration. This would lead to the competitor also choosing his best response from the previous iteration, which in the end means that a pure NE was found since the players' equilibrium strategies are mutual best responses.

In the other case, the best response was chosen more than one iteration ago. In this case the algorithm would get stuck in a loop and recur endlessly with the players choosing the same two or more best responses repeatedly. In order to break out of such a loop, the current strategy is forbidden and an alternative best response in form of an alternative optimal solution for the current player's model is seeked. This way our algorithm is able to handle both, degenerated and non-degenerated games. In our case, the degeneracy of a game depends on the number of optimal solutions to the employed models. If the models always have unique optimal solutions, the game is non-degenerated and vice versa.

If an alternative best response can be found, the forward search continues until another best response is chosen repeatedly. It is then again checked when this best response has been chosen before and it is forbidden, if necessary. If an alternative best response cannot be found (the model

becomes infeasible) for this player after forbidding the current strategy, a *backtracking procedure* is initiated which searches for an alternative for the competitor's latest best response. If an alternative best response cannot be found for him either, the algorithm goes back one more step and seeks an alternative best response for the player whose best response has been chosen before. In this manner, the procedure continues until either an alternative best response can be found and the forward search can continue or until the first iteration is reached without an alternative best response for the starting player. In the latter case the algorithm would terminate with the message that no pure NE can be found with the given starting player and starting strategy.

We provided the possibility of such a termination for our algorithm because pure NE do not always exist in general—opposed to mixed equilibria which exist in every finite game (Nash, 1951)—and we cannot be sure of a pure NE's existence in the games considered here. Thus, the algorithm we provide can be applied to general RM games, i.e. also to such games in which the existence of a pure NE is not guaranteed or cannot be proved.

5.3.1 Searching for Alternative Best Responses

In order to seek an alternative optimal solution for a player's model, additional constraints forbidding the current strategy while keeping the revenue equal must be added to it. Constraints (5.6) – (5.8) form the backbone for this. They are added to the model when a strategy must be forbidden for the first time and remain in place as long as strategies must be kept forbidden.

$$b^a_{pf} = \sum_{i=LB^a_{pf}}^{UB^a_{pf}} i y^a_{pfi} \qquad p \in P^a, f \in F \tag{5.6}$$

$$\sum_{i=LB^a_{pf}}^{UB^a_{pf}} y^a_{pfi} = 1 \qquad p \in P^a, f \in F \tag{5.7}$$

$$y^a_{pfi} \in \{0;1\} \qquad p \in P^a, f \in F, i \in [LB^a_{pf}, UB^a_{pf}] \tag{5.8}$$

In constraint (5.8) a binary variable y^a_{pfi} is defined which indicates whether i capacity units in the range between a pre-defined lower bound LB^a_{pf} and a pre-defined upper bound UB^a_{pf} are allocated to a product. We will explain the values for the bounds in more detail in Section 5.3.5.1. In restriction (5.6) the variable y^a_{pfi} is connected to the booking limit variables requiring a product's booking limit to equal the sum of individual capacity units allocated to this product. Due to constraint (5.7) each product is allocated only one (integer) number of capacity units in the interval within the bounds.

Once these variables and constraints are in place, constraints (5.9) and (5.10) are added to the respective player's model as well and it is solved anew. Here, B^a_{jpf} and R^a_j stand for player a's best response and his payoff in iteration j, respectively. Constraint (5.9) forbids a strategy which was used in iteration j to be chosen again. In particular, it forbids the sum of those binary variables that were set to 1 in iteration j to amount to the number of products. Together with constraint (5.6) this constraint affects the booking limits directly and leads to a new (different) best response if one exists. Constraint (5.10) prescribes the objective function to take on the same optimal value as in iteration j to make sure that the new strategy is indeed a best response.

$$\sum_{p=1}^{\mathcal{P}^a} \sum_{f=1}^{\mathcal{F}} y^a_{pf,i=B^a_{jpf}} \leq \mathcal{P}^a * \mathcal{F} - 1 \tag{5.9}$$

$$\sum_{p=1}^{\mathcal{P}^a} \sum_{f=1}^{\mathcal{F}} \pi_{pf}^a b_{pf}^a = R_j^a \tag{5.10}$$

Note that it is also possible to keep the models linear and forbid strategies via lower upper bounds (UBs) or higher lower bounds (LBs) for the booking limits. However, this might be quite complicated. For, if two products have the same price, they are interchangeable and raising the LB for one's booking limits has the same effect as lowering the UB for the other's booking limit by the same amount. So, one would have to keep track, which products are interchangeable in order to avoid getting the same alternative solution by lowering the UB for the booking limits of one and raising the LB for the booking limits of the other. This interaction must be monitored, especially in case of multiple forbidden strategies. The process would be complicated even further by the fact that one product might be traded off for two or more others if their summed revenue is equal. Also, by changing a variable's bound, one effectively reduces the feasible region of the problem because all solutions using the original bound are cut off. Finally, we will assume perturbed price matrices later to provide unique best responses. With the perturbed prices one would have to determine all alternative best responses to find the next best one which might take more time than solving a binary model since there might be very many optimal solutions to a model. In the worst case, all possible combinations of lower UBs/higher LBs would have to be tried to find that an alternative optimal solution does not exist. Thus, keeping a model linear and forbidding strategies by changing bounds would necessitate an own algorithm which would raise the computational burden within the algorithm for finding NE. Using the proposed binary approach is much easier, much more flexible, and still computationally tractable (in part because of the integrality of booking limits) as the computational studies below show. This is the reason, we use the binary approach for forbidding strategies.

Note further that in the search for alternative best responses Constraint (5.10) is not necessarily needed to keep the revenue at the desired level. For, if a solution is indeed an alternative optimal solution to a model, the payoff will be the same anyway. However, in this case one would have to monitor the new payoff and, in case it is not equal to the desired payoff (i.e. not alternative optimal solution exists), initiate the backtracking procedure. Including the constraint avoids this effort without having a noticeable impact on the computation time.

5.3.2 Basic Algorithm

Proving the existence of pure NE often fails in complex games for which a pure NE still would be desirable to be computed. Furthermore, the conditions for proving such existence are sufficient but not necessary, i.e. a game might possess a pure NE even if this conditions are not satisfied. On the other hand, uniqueness results might be (even more) difficult to establish and several pure NE might exist in realistic games leading to coordination issues about which NE will be chosen as the game's outcome. The algorithm we present in the following was developed for such games to compute a unique NE even if several might exist.

Our algorithm for computing a pure NE is described by Algorithm 5.1. Its main part consists of a loop which terminates when an NE is found or none can be found. The algorithm is an iterative procedure which is common for searching NE in games. Even for solving the game for one leg and two classes as considered in Netessine and Shumsky (2005) an iterative approach is needed since their derivatives of the players' objective function (the best response functions) turned out to be an implicit function from which isolating a variable cannot be done easily. See

also Topkis (1998, Sec. 4.3) for more details on this property and for an iterative algorithm how to solve games with implicit best response functions.

```
1: k ← 1
2: choose a = a_0
3: choose B_0^{-a}
4: loop
5:     if b^{-a} was not updated last in iteration k then
6:         Update b^{-a}
7:     end if
8:     Solve M^a
9:     Drop constraint (5.10) from M^a
10:    if M^a was solved optimally then
11:        if r^a ∈ R^a then
12:            if b^a was chosen in an iteration j with j ∈ I^a(r^a) then
13:                if j == k − 1 then
14:                    if a == a_0 then
15:                        Pure NE with B^a_{k−1} and B^{-a}_{k−1}
16:                    else
17:                        Pure NE with B^a_{k−1} and B^{-a}_k
18:                    end if
19:                    Stop!
20:                end if
21:                Add constraints (5.9) and (5.10) to M^a
22:                Remove iteration j from I^a(r^a)
23:                continue
24:            end if
25:        end if
26:        R^a_k ← r^a
27:        Add k to I^a(r^a)
28:        B^a_k ← b^a
29:        a ← a ∗ (−1)
30:        if a == a_0 then
31:            k ← k + 1
32:        end if
33:    else
34:        Drop all constraints of type (5.9) added to M^a in iteration k
35:        if a == a_0 then
36:            if k == 1 then
37:                No pure NE possible with B_0^{-a}
38:                Stop!
39:            else
40:                k ← k − 1
41:            end if
42:        end if
43:        a ← a ∗ (−1)
44:        Add constraints (5.9) and (5.10) to M^a with j = k
45:        Remove iteration k from I^a(R^a_k)
46:    end if
47: end loop
```

Algorithm 5.1: Algorithm for Computing an Exact Pure Nash Equilibrium

Before the forward search begins, the iteration count k is set to 1 in line 1. In lines 2 and 3 a starting player a_0 and a starting strategy for the competitor is chosen, respectively. The latter can be interpreted as the competitor's best response in iteration 0 B_0^{-a} and is needed for updating the competitor's strategy in constraint (5.3) of player a's model. This is done in line

6 if the current player's model was not updated last in the current iteration. For, updating a model is not necessary if an alternative best response is seeked repeatedly in the same iteration. This way the computational effort is decreased.

If necessary, the competitor's best response $\mathbf{b}^{-a}$ is updated in constraint (5.3). Here the assumption about complete information is of help and allows to keep the model deterministic. For the starting player, the competitor's best responses from the last iteration is taken here (in iteration 1, this is the starting strategy B_0^{-a} chosen in line 3). For the second player, the starting player's strategy from the current iteration is taken. In line 8 the current player's model $\mathcal{M}^a$ is solved (e.g. with commercial optimization software) before constraint (5.10) is removed from it in line 9. If the model was solved optimally—which is checked in line 10—the forward search continues with line 11. Otherwise, the backtracking procedure starting in line 34 is initiated. We will explain it in more detail below. In line 11 it is checked whether the current payoff has occurred before. If this is not the case, it is saved in the k^{th} position of the current player's vector of payoffs $\mathbf{R}^a$ in line 26. This leads to line 27 where the current iteration k is added to the set of iterations $\mathbf{I}^a(r^a)$ in which this payoff occurred, followed by saving the chosen vector of booking limits $\mathbf{b}^a$ in position k of the current player's vector of best responses $\mathbf{B}^a$ in line 28.

Note about set $\mathbf{I}^a$ that one set of iterations is created for each distinct element in vector $\mathbf{R}^a$. We save the iterations in such a way in order to minimize the effort of comparing best responses. For, if a strategy was chosen earlier as a best response to a certain strategy, it must have yielded the same payoff as the currently found one. When a model is solved optimally, the calculated best response thus only needs to be compared with the best responses from iterations with an equal payoff which further reduces the computational burden.

After saving the best response, the player is changed in line 29 and the iteration count is increased by one if necessary. In order to make sure that both players get to optimize their models in the same iteration, the iteration count is augmented only if the updated player is the starting player a_0. Once the player (and possibly the iteration) has been updated, the loop is rerun for the current player in the current iteration.

If in line 11 it is observed that the current revenue has occurred before, the algorithm continues with line 12 (instead of line 26) where it is checked whether the current vector of booking limits is equal to one of the vectors in $\mathbf{B}^a$ chosen in those iterations where the payoff was equal to the current payoff. If the current vector of booking limits was not chosen before, the algorithm continues with line 27.

If the current vector of booking limits was chosen before, it is checked whether it was chosen in the previous iteration in line 13. If this is the case, an exact pure NE was found and one only needs to determine which combination of booking limits makes up the NE. If the current player is the starting player (line 15), both players' vectors of booking limits from the previous iteration make up the NE. If the current player is the second player (line 17), the NE is made up of this player's strategy from the previous iteration and the first player's strategy from the current iteration. After finding an NE, the algorithm terminates in line 19.

However, if the current vector of booking limits was not chosen in the previous iteration, an alternative best response for the current player a is seeked beginning with augmenting his model in line 21. Here, constraints (5.9) and (5.10) are added to the model (possibly after adding constraints (5.6) – (5.8)). A strategy chosen in earlier than the previous iteration indicates a "somewhat" dominant strategy which is a best response to several strategies of the opponent.

Once the binary variables and constraints are added to the model, they remain in place as long as there are forbidden strategies in the model. A strategy forbidden in an iteration j

remains forbidden until the current player's model becomes infeasible in iteration j. As long as this is not the case, the forbidden strategy need not be compared with any strategy to be chosen in the forward search. Therefore iteration j (in which the forbidden strategy occurred) is removed from the set $\mathbf{I}^a(r^a)$ of iterations where the revenue was equal to the current revenue in line 22. This speeds up the algorithm further. Line 23 takes the algorithm to the beginning of the loop where the current player's model is re-solved with the additional constraints.

If his model is solved optimally, the algorithm continues with line 11. If, however the additional constraints rendered the model infeasible, the backtracking procedure is started in line 34. In this case all constraints of type (5.9) that were added to the current player's model in the current iteration are dropped in order to allow for these strategies to be chosen as best responses—to different strategies of the competitor—again later. If the model does no longer contain any constraints forbidding a strategy, the binary variables and constraints are removed from it as well. This makes the model linear again and speeds up computation again. Next, if the current player is not the starting player (checked in line 35), the player is changed in line 43 and an alternative best response for the starting player is seeked beginning with adding constraints (5.9) and (5.10) to his model in line 44 (possibly after adding the binary variables and constraints to it).

Note that if dropping one of the second player's constraints in line 34 leads to a previously forbidden strategy to be chosen again as a best response to one of player 1's best responses, the algorithm might get caught in a loop visited before. However, keeping these strategies forbidden may lead to an NE not being able to be found if it is made up of one of these strategies. We will illustrate this in Section 5.3.3.

If the backtracking procedure is performed with the starting player, the iteration count needs to be checked in line 36. If the current iteration is not the first, the iteration count is decreased in line 40 in order to seek an alternative best response for the competitor in the previous iteration. If the starting player's model becomes infeasible in iteration 1, no pure NE can be found with the starting strategy B_0^{-a} chosen in line 3 and the algorithm terminates in line 38. This ensures termination within finite time even if a pure NE cannot be found. The extension in Section 5.3.4 makes sure that the algorithm finds a pure NE with certainty if one exists.

Computing a pure NE with this algorithm in a game with more than two players would necessitate a few extensions. As in the competitive DLP, the index $-a$ would have to entail all players except the current player a in this case. This would be of relevance for choosing the starting strategy in line 3, for updating the competitors' in line 6, and for determining the NE strategies. With more than two players, the latter point is more complicated. E.g. assume five players 1, 2, 3, 4, and 5 with their number standing also for the order in which they choose best responses in an iteration. If now player 3 would choose a best response in two consecutive iterations, the pure NE would be made up of the previous iteration's strategies of players 3, 4, and 5 and of the current iteration's strategies of players 1 and 2. More general, if the current player a would choose a strategy in two consecutive iterations, the NE would consist 1) of the previous iteration's strategies of the current player and those players succeeding him in an iteration and 2) of the current iteration's strategies of those players preceding the current player in choosing best responses.

The last modification of the algorithm necessary to consider more than two players would be made to change players in lines 29 and 43. For this purpose, two functions would be necessary to determine the "next" player. In the forward search, after the current player would have found a best response, the succeeding player would be determined in line 29 as $a = (a \; mod \; |\mathcal{A}|) + 1$ with

mod standing for the modulo operator which returns the remainder of dividing a by $|\mathcal{A}|$. This function for determining the succeeding player makes sure that after the last player has chosen a best response, the next iteration is started again with the first player. This follows from the fact that in the case of the players' number corresponding to the order in which they choose best responses, $(a \ mod \ |\mathcal{A}|) = a$ for all values of a except for $a = \mathcal{A}$. Hence, for $a \leq |\mathcal{A}| - 1$ the function $a = (a \ mod \ |\mathcal{A}|) + 1$ reduces to $a = a + 1$. After the last player has chosen his best response, $a = (|\mathcal{A}| \ mod \ |\mathcal{A}|) + 1$ would turn a to 1 again due to $(|\mathcal{A}| \ mod \ |\mathcal{A}|) = 0$.

In case of a backtrack, on the other hand, the preceding player would be determined in line 43 as $a = ((a - 2) \ mod \ |\mathcal{A}|) + 1$. This function would make sure that the players would be counted down by 1 for $a \geq 2$ and that the previous iteration would be started with the "last player on the list" again. For, with $a = 1$, we would have $a = (-1 \ mod \ |\mathcal{A}|) + 1 = |\mathcal{A}|$.

5.3.3 Numerical Example

Consider again the degenerated game from Section 4.1. The bimatrix game is re-stated in Table 5.1 with matrix r^1 and r^2 for player 1 (the row player) and 2 (the column player), respectively.

r^1	s_1^2	s_2^2	s_3^2
s_1^1	120	120	122
s_2^1	125	125	126
s_3^1	130	126	124

a) Payoff Matrix r^1

r^2	s_1^2	s_2^2	s_3^2
s_1^1	122	127	130
s_2^1	122	127	127
s_3^1	122	127	127

b) Payoff Matrix r^2

Table 5.1: Payoff Matrices for the Numerical Example

For the numerical example we will concentrate on the two pure NE of this game; $(s_2^1;\ s_3^2)$ and $(s_3^1;\ s_2^2)$ with payoffs $(r^1 = 126;\ r^2 = 127)$ in both equilibria. Recall that strategies s_1^1 and s_1^2 are strictly dominated and could be eliminated. However, since for our algorithm the whole payoff matrices are not needed, we do not know their structure in advance. We merely show the complete matrices here to illustrate our procedure. In the course of our algorithm, only the best responses are generated as shown in Figures 5.1 and 5.2. Since dominated strategies cannot be best responses, examination of the payoff matrices and eliminating dominated strategies is not important for our procedure.

Assume that in lines 2 and 3 of Algorithm 5.1, player 1 is chosen to be the starting player a_0 and the starting strategy for player 2 is chosen to be $B_0^2 = s_2^2$, respectively. Next, player 1's model is updated and optimized. As payoff matrix r^1 in Table 4.1 shows, player 1 chooses his strategy s_3^1 as his best response. Since this means that his model was solved optimally (line 10), we proceed to line 11 of the algorithm where the payoff $r^1 = 126$ is compared to the vector of payoffs $\mathbf{R}^1$. As this set is empty in iteration 1, we go on to line 26 where the first entry of the vector of payoffs is taken by the current payoff. This is followed by adding iteration 1 to the set of iterations $\mathbf{I}^1(126)$ and saving the current strategy in the first position of the vector of strategies $\mathbf{B}^1$ in lines 27 and 28, respectively. Afterwards, the player is changed. Since the player is then player 2, the iteration count is not increased.

Now, the same steps as just described for player 1 are performed for the second player. Player 2 has two best responses to $s_3^1;\ s_2^2$ and s_3^2. If he plays strategy s_2^2, the pure NE $(s_3^1; s_2^2)$ will be

found in the next iteration by player 1. After optimizing his model in iteration 2, all checks in lines 10 – 14 will be evaluated with "true" and the algorithm will terminate in line 19.

This order of events is depicted in Figure 5.1a) where the arcs indicate that the strategy pointed to is a best response to the strategy the arc is pointing from. The arc from the best response chosen twice is dotted to only indicate which best response would follow since this step is not performed anymore. In accordance with the definition of NE in Section 4.1, two strategies—one per player—with arcs pointing to each other indicate an NE in form of mutual best responses.

Iteration	1	2
Strategy Player 1	s_3^1	s_3^1
Strategy Player 2	s_2^2	

a) NE $(s_3^1; s_2^2)$

Iteration	1	2
Strategy Player 1	s_3^1	s_2^1
Strategy Player 2	s_3^2	s_3^2

b) NE $(s_2^1; s_3^2)$

Figure 5.1: NE Through Forward Search

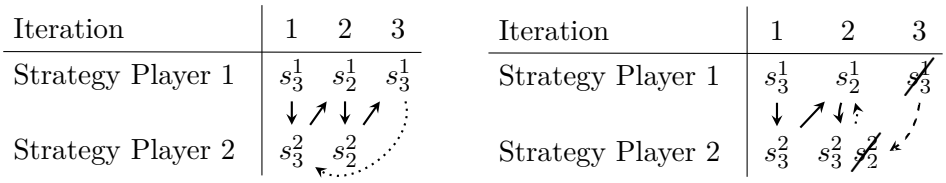
Note that with strategy s_1^2 as the starting strategy, the order of events would be similar with the NE being found by player 2 in iteration 2. For, strategy s_3^1 would yield player 1 a different revenue (126) in iteration 2 than in iteration 1 (130) being a best response to different strategies. Thus, although s_3^1 would still be played in two consecutive iterations, the algorithm would not terminate with player 1 in iteration 2 due to the check of revenues in line 11. Hence, in this case a further step would be needed for player 2 to play s_2^2 a second time (with the same revenue both times) and find the NE. Though the last step seems unnecessary in this case, it is not always performed as seen with s_2^2 as the starting strategy. With s_1^2 as the starting strategy, the redundant step would not be performed and the NE would be found earlier without checking the payoffs in line 11 of the algorithm but comparing the strategy to all previous strategies directly instead. However, in case an alternative to a best response needs to be computed—possibly several times—it is quicker to compare the payoffs before comparing the strategies. Comparing the new strategy with all previous ones (and not finding a match) every time an alternative best response is found, takes more time than comparing the payoffs first since the latter is only one value and the former is a vector. Thus, a possibly redundant step—which is not even always performed—is acceptable and does not slow down the algorithm noticeably.

When player 2 chooses strategy s_3^2 as a best response in iteration 1, no redundant steps are performed either. In this case player 1 would choose strategy s_2^1 in iteration 2. If then player 2 were to choose strategy s_3^2 again, the second pure NE would be found as depicted in Figure 5.1b). If, however s_2^2 is chosen as a best response to s_2^1, the algorithm gets stuck in a loop in iteration 3 as depicted in Figure 5.2a). For, then player 1 would play strategy s_3^1 a second time, but not in two consecutive iterations which would lead to player 2 playing strategy s_3^2 from iteration 1 again. This loop is averted through the backtracking procedure.

In Algorithm 5.1, player 1 choosing s_3^1 in iteration 3 would lead from line 13 to line 21 since he did not choose this strategy in the previous iteration for the first time. In line 21, Constraints (5.11) and (5.12) would be added to the model of player 1.

$$\text{Constraint to forbid strategy } s_3^1: \quad y^1_{OD,l,70} + y^1_{OD,h,30} \leq 1 * 2 - 1 \tag{5.11}$$

$$\text{Constraint to keep the revenue:} \quad 1 * b^1_{OD,l} + 2 * b^1_{OD,h} = 126 \tag{5.12}$$



a) Getting Caught in a Loop b) Backtracking and NE $(s_2^1; s_3^2)$

Figure 5.2: NE After Forbidding a Strategy

Afterwards, iteration $j = 1$ is removed from set $\mathbf{I}^1(126)$. The algorithm then continues with line 23 which means going to the beginning of the loop and solving player 1's augmented model in iteration 3. Since no other strategy yields him a payoff of 126 with player 2 playing strategy s_2^2, player 1's model would become infeasible with these constraints and from line 10 the algorithm would go to line 34 initiating the backtracking procedure. There, the constraint to forbid strategy s_3^1 would be dropped from player 1's model. (The constraint to keep the revenue at 126 would have been removed in line 9.)

Since the current player would be the starting player and the current iteration would not be the first, the iteration count would be decreased by one in line 40. Afterwards, an alternative for player 2's best response from iteration 2 (s_2^2) would be seeked beginning with changing the player in line 43. Next, strategy s_2^2 would be forbidden while his revenue would be kept at 127. His alternative best response would be s_3^2 which constitutes an NE with player 1's strategy s_2^1.

The backtracking procedure as well as the continued forward search with the finding of the NE are shown in Figure 5.2b). There, the forbidden strategies are crossed out and the dashed arcs indicate the backtrack.

To understand why a constraint to forbid a strategy must be dropped from a player's model, assume that the entry at $(s_2^1; s_3^2)$ in matrix r^2 is 126 so that $(s_3^1; s_2^2)$ is the only pure NE in the game. Now assume that the order of events is as in Figure 5.2a). However, after forbidding strategy s_3^1 in iteration 3 and strategy s_2^2 in iteration 2, the backtracking procedure would continue in iteration 2 and strategy s_2^1 would also have to be forbidden since there would be no alternative for s_2^2. Finally, in iteration 1 s_2^2 would be chosen as the alternative for s_3^2 to make up the pure NE.

Note that both of the strategies that constitute the NE in this modified game (s_3^1 for player 1 and s_2^2 for player 2) would have been forbidden in the course of the backtracking procedure. If the constraints to forbid these strategies would not have been dropped again from the players' models as these became infeasible, the NE would not have been found. For, strategy s_2^2 would not have been able to be chosen as an alternative for s_3^2 in iteration 1 after having been forbidden in iteration 2 and s_3^1 would not have been allowed to be picked as a best response to s_2^2 since it would have been forbidden in iteration 3.

5.3.4 Generating Unique Searching Paths

As the above example showed, for an NE to be found with Algorithm 1, one of the players' equilibrium strategy must be chosen in the forward search. I.e. an equilibrium strategy must be a best response to at least one strategy of the competitor which is chosen in the forward

search. Only then the equilibrium strategies can be chosen as best responses; either directly in the forward search or in the backtracking procedure.

As a counter example consider the game described by Table 5.2. If one of the strategies s_1^1, s_2^1, s_1^2 or s_2^2 is chosen as the starting strategy in line 3 of our algorithm, the pure NE (s_3^1, s_3^2) will not be found by Algorithm 1 because none of the equilibrium strategies is a best response to one of these strategies and hence cannot be chosen in the forward search. In order to find the pure NE, one of the equilibrium strategies would have to be chosen as the starting strategy in Algorithm 1 as it is. In case this starting strategy is not picked, a new one would have to be chosen and the algorithm would have to be rerun. To avoid this, we here extend the algorithm so that it only needs to be started once and finds a pure NE with certainty if one exists.

r^1	s_1^2	s_2^2	s_3^2
s_1^1	2	1	0
s_2^1	1	2	0
s_3^1	0	0	3

a) Payoff Matrix r^1

r^2	s_1^2	s_2^2	s_3^2
s_1^1	1	2	0
s_2^1	2	1	0
s_3^1	0	0	3

b) Payoff Matrix r^2

Table 5.2: Payoff Matrices for the Exemplary Game

For this, a unique starting strategy must be provided in line 3 of the algorithm to allow the starting player to exploit his full range of booking limit combinations—from non-competitive to competitive. We will explain this in more detail in Section 5.3.5.1. Next, the algorithm continues with line 4. However, if it now becomes necessary to forbid the starting player's strategy from the first iteration and to seek an alternative strategy, constraint (5.9) is added not with constraint (5.10) but with a constraint that merely makes sure that the starting player's revenue must not be lower than his non-competitive revenue. This allows the starting player to choose all of his possible strategies in iteration 1 and has the same effect as if the algorithm were rerun repeatedly with these strategies as starting strategy and player 2 as the starting player.

If now the starting player's model cannot be solved in the first iteration, the game does not have a pure NE. For, in this case all possible starting strategies will have been considered without finding an NE. However, the extended algorithm will still terminate in finite time because the players have only finite numbers of available (integer) strategies.

However, depending on the parameters (demand, capacities, prices etc.) there might be several (pure) NE in a game as the example in the previous section showed. This, on the other hand, would lead to coordination problems about the players' choice of NE strategies. If both players choose strategies that constitute the same NE, they indeed end up in one. However, if they choose strategies corresponding to different NE, they fail to pick an NE. See also (Fudenberg and Tirole, 1991, Section 1.2.4). In Harsanyi and Selten (1988) a rule for selecting a unique NE was provided.

Furthermore, in order to pick a certain NE, all of them need to be computed first. Finding all of them, on the other hand, might not be possible since degenerated games might have infinite sets of equilibria as noted in Section 4.3. So one can be satisfied with finding only one NE. An issue that arises with algorithms which terminate after finding only one NE, such as the algorithms by Lemke and Howson (1964) and McKelvey and McLennan (1996), is that they might find different NE in degenerated games depending on the starting strategy and, even more, on the choice of best responses during the iterative search. Practically speaking, different

solvers and/or algorithms employed by the competing airlines for solving the models on hand might lead to different optimal solutions (and hence different best responses) if several exist. Similarly, in (primally) degenerated LPs there may be more than one candidate for entering and/or exiting the basis in a pivot step. Depending on the choice of the variables leaving and entering the basis, one might end up in different optimal solutions of the LP if several exist.

Obviously, the coordination problem with NE disappears if there is only one (pure) NE in the game or if every model has only one unique solution (in which case the iterative search would have a unique searching path). Indeed, the majority of literature mentioned above that investigates competition in an RM setting using game theory, is concerned with deriving conditions under which a game's NE is unique. However, if these conditions are not fulfilled, the uniqueness is not guaranteed and an algorithm possibly tailored to find the unique NE is useless. The same applies when a player has several best responses to his opponent's equilibrium strategy and is completely indifferent between them. However, coordination issues might have their roots even earlier, namely in the iterative search for best responses. If a player has several best responses to his competitor's strategy—i.e. his model has several optimal solutions—and he is completely indifferent about them, the players might end up making different predictions about the other's behavior if they have different solvers and/or employ different algorithms to solve the models.

For this reason, we extend our algorithm to generate a unique searching path in the course of the iterative search independently of the type or structure of the game, the parameters, or the number of optimal solutions for a model (and thus best responses to a player's strategy). This makes the algorithm suitable for games where uniqueness of an NE is not guaranteed or cannot be established. For this, we use a perturbed price matrix so that the model turns out to have a unique solution and hence generates unique best responses. Perturbation, first described by Charnes (1952), is a common tool in linear programming for turning degenerated models into a non-degenerated ones. See also e.g. Murty (1983, Ch. 2.5.7–2.5.8). In Lemke and Howson (1964) perturbation was used to turn a degenerated game into a non-degenerated one.

In a degenerated game a model's candidate optimal solutions (a player's best responses) express themselves through equal payoffs. However, if several optimal solutions exist for a player's model $\mathcal{M}^a$, they differ in the booking limits for substitutable (combinations of) products which use the same legs and generate the same revenue. In order to make the optimal solutions unique and to eliminate the possibility of substituting one product for another we add a matrix $\boldsymbol{\tau^a}$ (which has the same dimensions as the player's price matrix) of an exponentiated factor τ^a to a player's matrix of prices. For this, the prices must be sorted in a fixed order which must not be changed once it is determined. Depending on the order of perturbation, the elements in the matrix $\boldsymbol{\tau^a}$ are exponentiated differently. Equation (5.13) shows such a matrix with the perturbation order "first alphabetic by O&D name, then by fare class". The rows of the matrix stand for the O&D pairs and the columns stand for the fare classes. The second superscripts in the matrix stand for exponents which in this case are computed as "$\mathcal{F}*$ (row number -1) + column number".

$$\boldsymbol{\tau^a} = \begin{Bmatrix} \tau^{a,1} & \tau^{a,2} & \tau^{a,3} & \dots & \tau^{a,\mathcal{F}} \\ \tau^{a,\mathcal{F}+1} & \tau^{a,\mathcal{F}+2} & \tau^{a,\mathcal{F}+3} & \dots & \tau^{a,\mathcal{F}+\mathcal{F}} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ \tau^{a,\mathcal{F}*(\mathcal{P}^a-2)+1} & \tau^{a,\mathcal{F}*(\mathcal{P}^a-2)+2} & \tau^{a,\mathcal{F}*(\mathcal{P}^a-2)+3} & \dots & \tau^{a,\mathcal{F}*(\mathcal{P}^a-2)+\mathcal{F}} \\ \tau^{a,\mathcal{F}*(\mathcal{P}^a-1)+1} & \tau^{a,\mathcal{F}*(\mathcal{P}^a-1)+2} & \tau^{a,\mathcal{F}*(\mathcal{P}^a-1)+3} & \dots & \tau^{a,\mathcal{F}*(\mathcal{P}^a-1)+\mathcal{F}} \end{Bmatrix} \tag{5.13}$$

Together with the unique starting strategy motivated above, applying perturbation generates unique best responses and thus unique searching paths which leads to both players ending up in the same NE even if several exist. This way, instead of dealing with uniqueness results of NE, we can guide the search for an NE and are able to avoid coordination issues in our own way.

In our case, the information connected to the perturbation matrices can be interpreted as an airline's long-term goals. If an airline e.g. wants to establish itself as a premium airline, it can perturb the prices so that in case of a tie, the higher-class product is given a priority. The opposite is the case if the airline pursues a diversified customer structure. Such strategic goals are made public e.g. in the companies' annual reports so that competitors can deduce the order of perturbation from each other's strategic goals. Hence, this order can be thought of as common knowledge drawn from the airlines' long-term goals which can be interpreted as *implicit communication* between the players and resolve coordination issues (Holler and Illing, 2009, p. 86). What is important, is that the assumption of complete information can be maintained given the perturbation. We could also model the implications connected to the perturbation explicitly through constraints but more constraints might lead to lower payoffs and the model still can have several optimal solutions. Instead, the order in which the price matrix is perturbed can be used to control the choice of the optimal solution and make sure that it is unique.

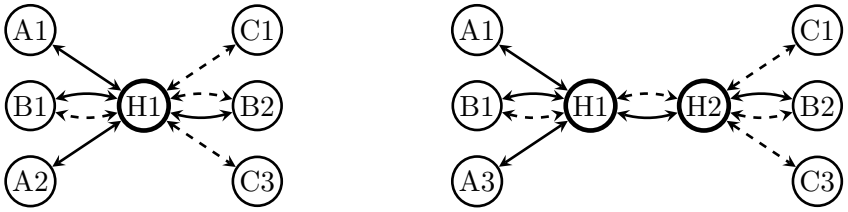
Comparing our algorithm with the algorithm by Audet et al. (2001), we can see that the latter is less target-oriented since it tries almost all combinations of strategies to see whether they constitute an equilibrium. While our Algorithm 5.1 searches for best responses in every step, the algorithm by Audet et al. (2001) allows the modification of only one player's model in each step (node of the search tree). Furthermore, while our algorithm uses a backtracking procedure in case of repeating best responses which do not constitute an NE, their algorithm does not suppress this possibility and actively adds certain constraints to the models repeatedly. Also, with our algorithm, we cannot receive infeasible models in the forward search which can happen with the algorithm for enumerating all extreme NE. Finally, while we satisfice with a single NE and propose a perturbation of the players' prices to deal with degenerated games, Audet et al. (2001) explicitly developed their algorithm to find all NE in degenerated games as well.

5.3.5 Computational Study

5.3.5.1 Test Bed

We next present a computational study with the complete algorithm described in the previous sections. We assumed networks with one and two hubs—similar to those in Figure 2.1 and numbers of 20, 40, 60, 80, and 100 airports to be connected with each hub, respectively. These are realistic numbers comparable to the different sized networks of the members of the three major airline alliances OneWorld, SkyTeam, and Star Alliance. There, at least two thirds of the members have one or two hubs and most airlines do not serve less than 20 and more than 200 destinations, see OneWorld (2014a), SkyTeam (2014), and StarAlliance (2014). Figure 5.3 shows exemplary networks operated by the competitors. The solid and dashed arcs stand for the different airlines' networks. Node pairs connected by both types of arcs indicate competition. E.g. the O&D combination B1–B2 is offered by both airlines and hence this itinerary belongs to the set A. On the other hand, the itinerary A1–H1 is only offered by one airline, which is why this itinerary belongs to this airline's set NA^a. Throughout this thesis it will be assumed that the customers only buy a ticket from one airline, i.e. either from a or from $-a$. Thus, we do not

consider the possibility of e.g. booking a flight from A1–H1 from one airline and H1–B2 from the other, if the former does not accept requests for the complete itinerary A1–B2 anymore.



a) Network with One Hub b) Network with Two Hubs

Figure 5.3: Simple Network Structures

The spoke airports were connected with each other only through the hubs, i.e no direct flights between two spoke airports were allowed. With one hub, products using up to two legs could be offered. If two hubs were used, they were connected with each other so that itineraries with at most three flight legs were offered. Thus, 420 up to 40,602 O&D pairs (1,680 up to 162,408 products) were offered by each player in total. Since competitors are seldom identical, we assumed the percentage of O&D pairs affected by competition (the competition intensity, CI) to amount to 25%, 50%, and 75% of an airline's total offered itineraries. I.e. we assumed both competitors to have the same hubs and 25%, 50%, and 75% respectively of the feeder flights to be identical to the competitors' feeder flights. We set the capacities on half of the flights going in and out of the hubs to 100 seats and 200 each and the capacities on the legs connecting the hubs to 300.

Out of a player's possible products, we chose randomly so many that each leg was demanded by at most 60 products. For these products ten values of original demand d^a_{pf} were generated randomly for each player from the Poisson distribution with means $\mu \in \{2, 4, 6\}$ for each network setting. With the different mean demands and capacities, we had different demand-to-capacity ratios, computed as $\frac{60*\mu}{C_l}$. On the legs with capacities of 100, 200, and 300, these ratios amounted to at most 1.2, 0.6, and 0.4 with a mean demand of $\mu = 2$, at most 2.4, 1.2, and 0.8 with $\mu = 4$, and at most 3.6, 1.8, and 1.2 with $\mu = 6$, respectively. In fact the ratios were lower, though, since the Poisson distribution is positively skewed which means that lower values are more probable than higher ones and which is most noticeable at such low values for the mean demand.

Due to such low values of the average demands, the lower bounds LB^a_{pf} were set to 0. Since we assumed deterministic demands, we could set the upper bounds UB^a_{pf} precisely as well. These were assumed to be d^a_{pf} for the products not affected by competition and to $d^a_{pf} + \alpha^{-a,a}_{pf} d^{-a}_{pf}$ for the products affected by competition to allow a player to capture all spillover demand. Note that a stochastic model should set $LB^a_{pf} = 0$ and UB^a_{pf} to the capacity of the smallest plane demanded by a product to cover all possible values for a product's booking limits. A wider range between the bounds means more binary variables and will tendentially lead to a better solution (possibly taking more time for optimization) while narrowing the range lowers the number of binary variables and might speed up the optimization (at the expense of a worse solution).

The unique starting strategy needed for the extended algorithm was set as follows. As mentioned above, the strategy must allow the starting player to exploit all of his possible strategies, i.e. all non-competitive as well as all competitive ones. In order to enable the competitive strategies, the starting player must receive the highest possible spillovers to be able to set his booking limits as high as possible. Hence, we could determine all non-competitive booking limits for

the second player (not the starting player) in a pre-process and set the booking limits in the starting strategy to the lowest values of all these non-competitive booking limits. However, this pre-process turned out to be very inefficient because determining all non-competitive solutions could take quite a lot of time—often more time than needed to find an NE afterwards—because there are possibly many non-competitive optimal solutions. Hence, simply set the booking limits in the starting strategy to the lower bounds (= 0) of the second player. In view of the relatively low values for the lower bounds, it is not even an unreasonable assumption that the overall lowest booking limits turn out to 0.

The prices were drawn from the uniform distribution. The intervals for the first (i.e. most expensive, full) class lay between 300 and 400 for single-leg products, between 600 and 800 for two-leg products and between 900 and 1200 for three-leg products, respectively. We assumed four fare classes and the interval boundaries of the higher classes 2 – 4 were 75%, 50%, and 25% of the first class boundaries, respectively. The prices were drawn individually for both players to avoid completely identical prices. For, in practice competitors seldom charge completely identical prices.

The ratios for the spillover demand were functions of the players' prices to model the customers' willingness to pay: $\alpha_{pf}^{-a,a} = 0.5 - 0.1\frac{\pi_{pf}^{a} - \pi_{pf}^{-a}}{\overline{\pi}_{pf} - \underline{\pi}_{pf}}$. Here, $\underline{\pi}_{pf}$ and $\overline{\pi}_{pf}$ are the lower and upper interval boundaries from the price distribution, respectively. The denominator of the fraction is fixed for a product by the interval boundaries $\overline{\pi}_{pf} - \underline{\pi}_{pf}$ from the price distribution. After generating the product prices, the enumerator's value is determined. Since the players' prices lie between the interval boundaries, the fraction's value turns out between -1 and 1. It is negative when $\pi_{pf}^{a} < \pi_{pf}^{-a}$, positive when $\pi_{pf}^{a} > \pi_{pf}^{-a}$ or 0 when $\pi_{pf}^{a} = \pi_{pf}^{-a}$. Multiplying the fraction with -0.1, the value for $\alpha_{pf}^{-a,a}$ is raised above 0.5 when $\pi_{pf}^{a} < \pi_{pf}^{-a}$, while it is reduced below 0.5 when $\pi_{pf}^{a} > \pi_{pf}^{-a}$. With $\pi_{pf}^{a} = \pi_{pf}^{-a}$, we have $\alpha_{pf}^{-a,a} = 0.5$. Hence, if both competitors charged identical prices, half of the customers requested a ticket from the competitor if they were refused by their preferred airline. Charging a higher price than the competitor meant less requests for the respective airline and vice versa. The lowest value $\alpha_{pf}^{-a,a}$ could attain was 0.4 and the highest value was 0.6. Note that the higher a fare class (the cheaper a ticket), the more sensitive the customers reacted to price differences because the intervals for the prices were narrower. This is a realistic assumption.

The value for τ^a, the perturbation parameter, depends on the considered game and the players' prices. This value must be chosen in such a way that the (unique) optimal solution of the perturbed problem is also an optimal solution to the original problem. More precisely, choosing τ^a we must make sure that 1) two (or more) equal prices in the original problem are unequal in the perturbed problem, that 2) unequal prices in the original problem remain unequal while at the same time 3) a (sum of) price(s) which was lower (or higher) then another (sum of) price(s) in the original problem remains lower (or higher) in the perturbed problem. Since our original prices are integers and we add an exponent vector to the price vector, a value $\tau^a < 1$ takes care of points 1) and 2).

Concerning point 3), we need to consider the following case while bearing in mind that the smallest positive difference between a (sum of) integer price(s) is 1. Considering two individual prices within this context is unnecessary because this case is covered by point 2). Hence, we must only compare a sum of prices to a single price (e.g. prices $\pi_{AB1}^{a} + \pi_{BC1}^{a}$ and π_{ABC1}^{a}) or a sum of prices to another sum of prices (e.g. $\pi_{AB1}^{a} + \pi_{BC1}^{a} + \pi_{CD1}^{a}$ and $\pi_{ABC1}^{a} + \pi_{CD1}^{a}$). Here, we must make sure that the perturbed summands of the smaller sum do not add up to a larger

value than the perturbed summands of the higher sum. Since the price difference only amounts to 1 in the smallest case, it suffices to make sure that the perturbing values added to the smaller sum's summands do not add up to ≥ 1.

Since we assume 300 as the highest capacity on a plane and all products only occupy one seat, the highest number of summands in our case is 300. Hence, we must make sure that (in the worst case) the first 300 values of our perturbing vector do not sum up to ≥ 1 and do not amount to a higher sum than any other sum which was higher before the perturbation. For this, we could assume $\tau^a = \frac{1}{300}$ (or any smaller number). However, this might lead to rounding problems in the implementation because in the largest case, we would have $(\frac{1}{300})^{162,408}$ which might not be able to be stored in sufficient precision. To avoid this, we instead multiplied all original prices by 300 (which does not change the solution) so that then the smallest positive difference between a (sum of) price(s) and another is 300 and we must avoid the summed perturbing values to sum up to ≥ 300. Then, we can assume $\tau^a = 0.9999$ (or a smaller number) which can be stored in sufficient precision even if taken as the basis for an exponent many times. After the optimization we simply have to subtract the perturbing parameters from the prices and divide the prices by 300 to receive the original prices and payoffs. We tried different lexicographic orders for the products; "first alphabetic by O&D name, then increasing by fare class" and "increasing by fare class, then alphabetic by O&D name".

All in all, we constructed 1,800 instances (10 types of networks, three values for the competition intensity, two ways of ordering the products for the perturbation, and 10 pairs of demand for each of these cases, and 3 values of mean demands). We limited the computation time for each instance to three hours.

5.3.5.2 Results

The models and algorithms were implemented in C++ and version 5.6.3 of the Gurobi solver was used to solve the models. The study was done on a computer using an Intel Core i7-620M CPU (2.67 GHz; 2 cores), 8 GB memory and Windows 7 SP1 (64-bit).

The most important result we found was that a pure NE was found in 98.11% of our 1,800 instances within our time limit of 3 hours. Only in 34 cases (mostly with 2 hubs and 80 and 100 spokes, respectively) the algorithm timed out after three hours and did not find an NE. Thus, our algorithm is suited to compute pure NE in real-sized networks, even if additional binary variables and constraints are added to the model.

Strategies had to be forbidden in only 191 out of all instances where an NE was found. In 89 of these instances only one of the players had a repeating strategy. In most instances with forbidden strategies, only one (126 and 140 for player 1 and 2, respectively) or two (10 and 13 for player 1 and player 2, respectively) strategies were forbidden. At most 3 strategies were forbidden by both players. Mostly, the instances with a CI of 0.5 or 0.75 were affected by repeating strategies. In most cases, the instances with forbidden strategies were those with the highest number of iterations in the corresponding network with the corresponding CI.

In the remainder of this section we show only the results for the instances with the prices perturbed in order "increasing by fare class, then alphabetic by O&D name". The results for perturbation order "alphabetic by O&D name, then increasing by fare class" can be found in Appendix A.1. The results for both perturbation order are similar, though. Figures 5.4 – 5.6 show the average computation times in minutes needed for finding a pure NE in the different networks with different competition intensities (CI) as ratios for the products affected by competition.

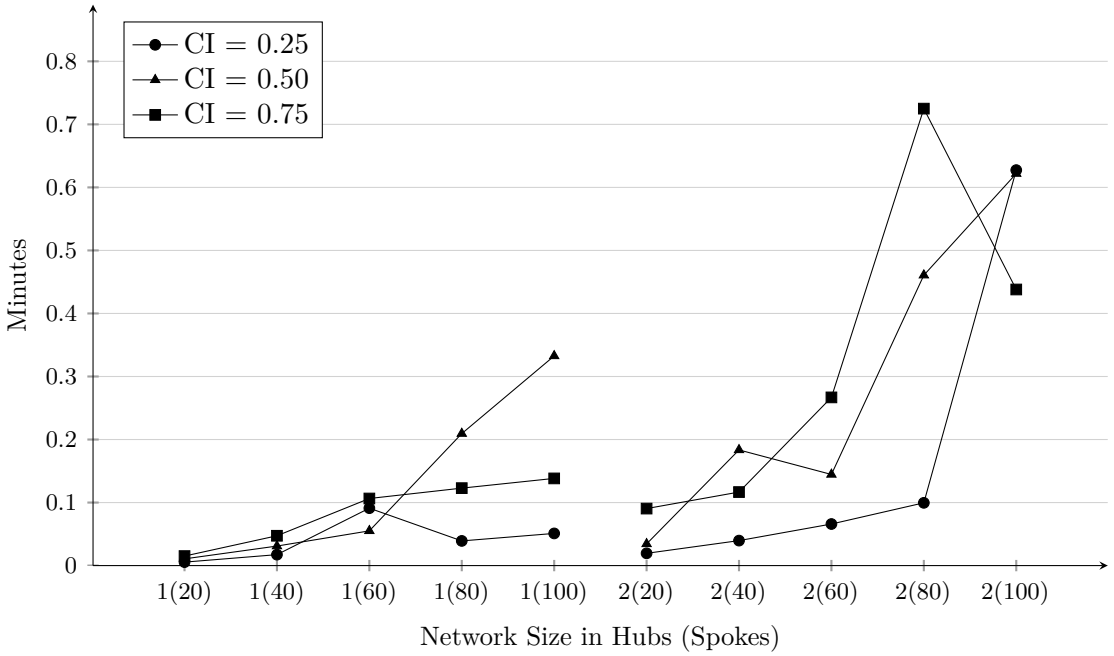


Figure 5.4: Average Computation Times in Minutes with $\mu = 2$

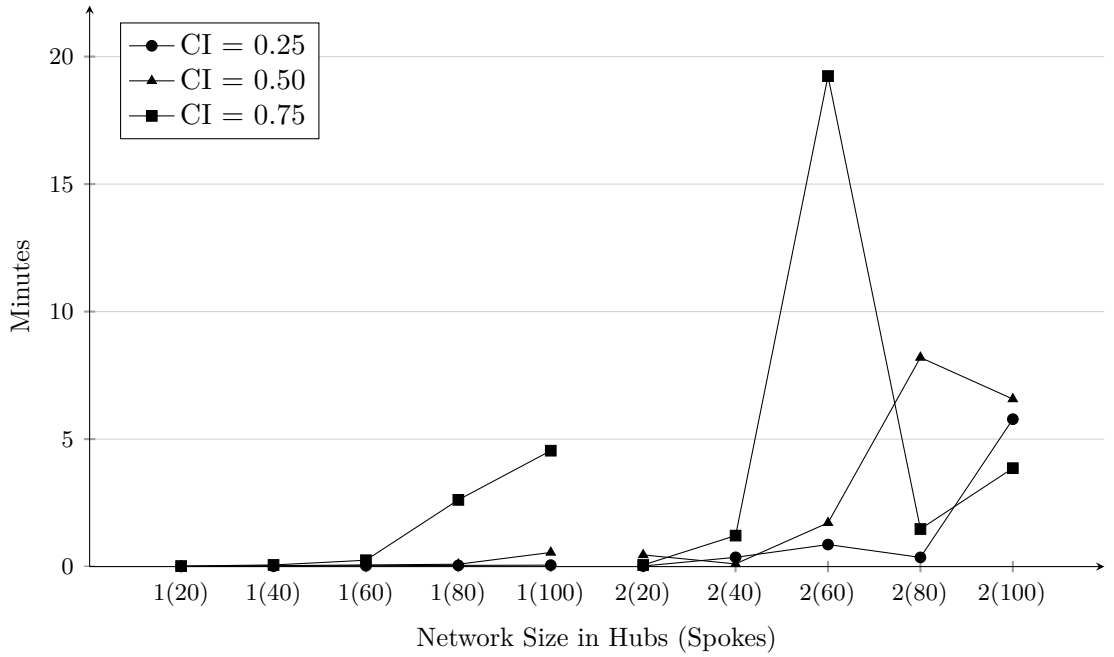


Figure 5.5: Average Computation Times in Minutes with $\mu = 4$

From the figures it becomes clear that the more products were affected by competition and the bigger the considered network, the more time was needed to find an NE. While in the smallest networks (less than 60 spoke airports) an NE was usually found within seconds, it took up to about 43.3 minutes on average with 2 hubs and 100 spokes, mean demand of $\mu = 6$ and a competition intensity of CI = 0.75. This increase of time is due to 1) a higher computational effort for solving the—ever larger—models and to 2) a higher number of iterations needed to terminate. The "outliers" in the graphs (e.g. the high computation times in the networks with

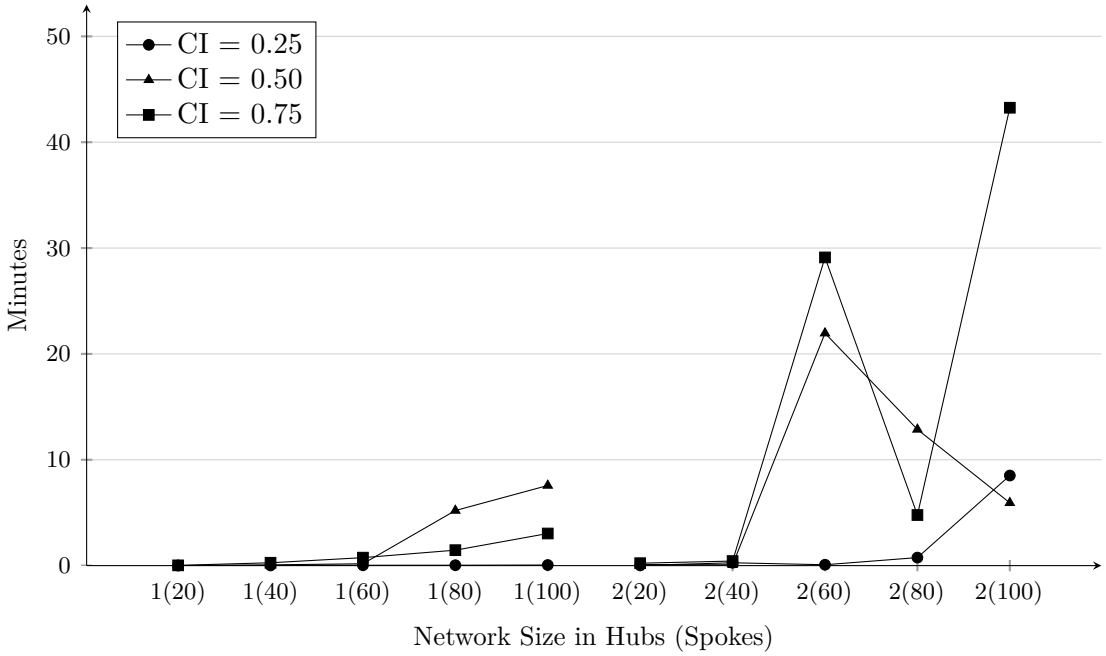


Figure 5.6: Average Computation Times in Minutes with $\mu = 6$

two hubs and 60 spoke airports) can also be explained by less timed out instances in these networks. While in these networks in almost all instances a pure NE was found within three hours (even when it took longer), the instances with the larger networks timed out more often which leads to mainly those instances with short computation times to play a role in the graphs.

Over all instances where an NE was found, the most time needed to find an NE was 151.7 minutes in an instance with the biggest network, $\mu = 6$, CI = 0.75, and perturbation order "alphabetic by O&D name, then fare class". The highest number of iterations needed to find an NE was 184 in an instance in a network with two hubs and 60 spoke airports and a mean demand of $\mu = 4$. The least number of iterations needed to find an NE was 3 iterations over all instances. Comparison of Figures 5.4 – 5.6 with their pendant Figures 5.7 – 5.9 showing the average numbers of iterations needed to find an NE reveals a positive correlation between the computation time and the number of iterations needed to terminate.

On average, the number of iterations—like the computation time—needed to find an NE rose with the competition intensity and the size of the network. The reason for this is that with a growing competition intensity and with a larger network there are more possible combinations for the booking limits of products employing a leg. The chosen combination, on the other hand, affects the spillover demand which has an influence on the competitor's demand and provides him with more possible combinations. The more spillover demand is exchanged, the more iterations are needed to settle down at an NE.

We compared the summed equilibrium payoffs to the cooperative (central) payoff and to the summed non-competitive payoffs when both airlines ignored competition. For the cooperative case the prices for the products affected by competition were set as average prices charged by the players. The other products' prices were taken directly as the player's respective price. The products' demands and the capacities on the legs used by these products were taken as the sum of the players' values. The results are shown in Tables 5.3 – 5.5. Here, C, NE, and NC stand for central, equilibrium summed, and non-competitive summed payoffs, respectively. The entries in

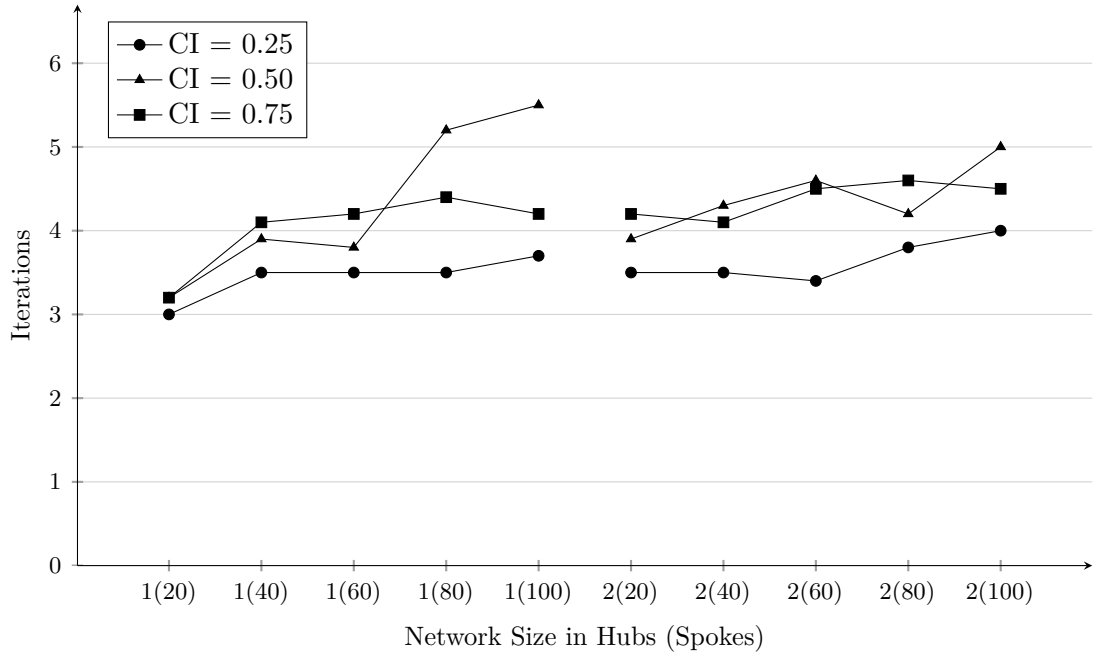


Figure 5.7: Average Numbers of Iterations with $\mu = 2$

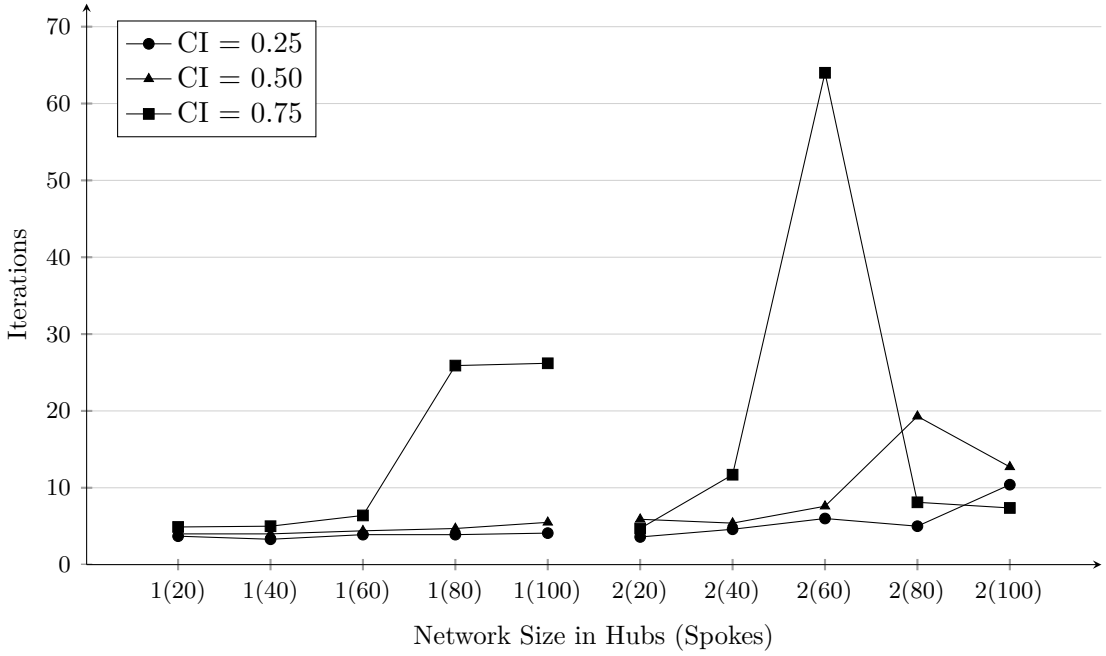


Figure 5.8: Average Numbers of Iterations with $\mu = 4$

the first row of columns 3 – 12 stand for the networks as "Hubs(Spokes)". The revenues were computed without the perturbing parameters τ. These were not needed for the cooperative case and for comparing the values with each other, we used the non-perturbed prices for the other settings as well.

As the tables show, the summed NE payoffs and non-competitive payoffs are lower than the cooperative payoffs in most cases. However, taking competition into account generates slightly higher payoffs than ignoring it. This is due to the non-negative spillover demand in constraint

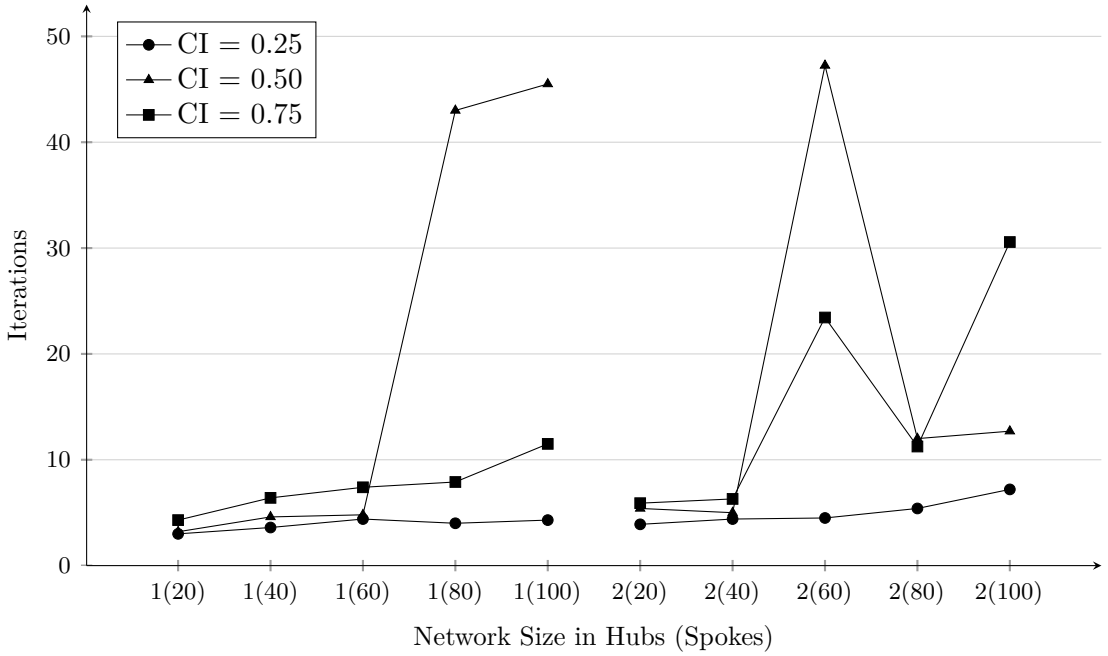


Figure 5.9: Average Numbers of Iterations with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	99.91%	99.97%	99.96%	99.97%	99.98%	100.01%	100.04%	100.02%	99.97%	99.95%
	NC/C	99.87%	99.92%	99.92%	99.93%	99.94%	99.94%	99.99%	99.97%	99.93%	99.91%
	NC/NE	99.97%	99.95%	99.96%	99.96%	99.96%	99.93%	99.95%	99.96%	99.96%	99.96%
0.50	NE/C	99.92%	100.00%	100.03%	100.04%	100.06%	100.20%	100.16%	100.09%	100.04%	100.01%
	NC/C	99.85%	99.91%	99.95%	99.97%	99.98%	100.06%	100.06%	100.00%	99.96%	99.93%
	NC/NE	99.93%	99.91%	99.92%	99.92%	99.92%	99.86%	99.90%	99.92%	99.92%	99.92%
0.75	NE/C	99.92%	100.03%	100.03%	100.07%	100.10%	100.31%	100.27%	100.15%	100.12%	100.06%
	NC/C	99.84%	99.93%	99.93%	99.96%	100.00%	100.14%	100.15%	100.05%	100.03%	99.97%
	NC/NE	99.92%	99.90%	99.90%	99.90%	99.90%	99.83%	99.88%	99.90%	99.91%	99.91%

Table 5.3: Average Payoff Ratios with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.12%	100.13%	100.12%	100.09%	100.12%	100.14%	100.05%	100.01%	99.97%	99.99%
	NC/C	100.02%	100.06%	100.04%	100.02%	100.04%	100.05%	99.98%	99.95%	99.92%	99.93%
	NC/NE	99.90%	99.93%	99.92%	99.92%	99.92%	99.91%	99.93%	99.94%	99.94%	99.95%
0.50	NE/C	100.28%	100.35%	100.32%	100.31%	100.31%	100.42%	100.18%	100.14%	100.09%	100.10%
	NC/C	100.09%	100.20%	100.16%	100.15%	100.15%	100.25%	100.05%	100.03%	99.98%	100.00%
	NC/NE	99.82%	99.85%	99.84%	99.84%	99.84%	99.83%	99.87%	99.89%	99.89%	99.90%
0.75	NE/C	100.32%	100.45%	100.40%	100.44%	100.44%	100.54%	100.28%	100.21%	100.16%	100.15%
	NC/C	100.09%	100.26%	100.19%	100.21%	100.21%	100.32%	100.09%	100.06%	100.01%	99.99%
	NC/NE	99.77%	99.81%	99.79%	99.78%	99.77%	99.78%	99.81%	99.84%	99.85%	99.85%

Table 5.4: Average Payoff Ratios with $\mu = 4$

(5.3) and leads to a tendentially higher demand and thus payoff when considering competition. This effect increases with a higher competition intensity because more products' demands are potentially raised through spillover demand. The higher the competition intensity (the more products are affected by spillover demand), the higher an airline's NE payoff and the lower the ratio between the NE payoffs and the non-competitive payoffs.

What stands out is that—except for a few exceptions—with a certain CI the increase in summed revenues in the NE compared to the non-cooperative summed payoffs is almost constant

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.22%	100.21%	100.23%	100.17%	100.22%	100.11%	100.10%	100.09%	100.08%	100.06%
	NC/C	100.14%	100.08%	100.10%	100.06%	100.09%	100.02%	100.02%	100.01%	99.99%	99.98%
	NC/NE	99.91%	99.87%	99.87%	99.89%	99.88%	99.91%	99.92%	99.92%	99.91%	99.92%
0.50	NE/C	100.45%	100.53%	100.54%	100.48%	100.54%	100.38%	100.31%	100.26%	100.24%	100.23%
	NC/C	100.30%	100.28%	100.28%	100.24%	100.29%	100.21%	100.16%	100.11%	100.07%	100.06%
	NC/NE	99.84%	99.75%	99.74%	99.76%	99.75%	99.83%	99.85%	99.85%	99.83%	99.84%
0.75	NE/C	100.53%	100.83%	100.81%	100.73%	100.82%	100.54%	100.46%	100.36%	100.36%	100.33%
	NC/C	100.34%	100.43%	100.42%	100.35%	100.43%	100.32%	100.25%	100.14%	100.13%	100.11%
	NC/NE	99.81%	99.60%	99.61%	99.63%	99.61%	99.78%	99.79%	99.79%	99.77%	99.77%

Table 5.5: Average Payoff Ratios with $\mu = 6$

over all networks with a similar structure (i.e. one hub/two hubs). This is indicated by the almost identical NC/NE ratios in the corresponding rows of Tables 5.3 – 5.5.

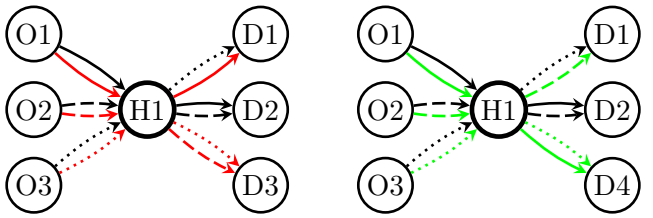
Note, however, that we only used the cooperative payoffs for relative comparison with the other values and that the cooperative payoffs cannot be compared to the other values directly for several reasons. First, the cooperative prices are taken as averages from the competitors' prices (which falsifies the cooperative payoff and leads to some of the entries in the tables being $> 100\%$) while the cooperative demand is the sum of the individual players' demands. Secondly, in case of competition not all denied customers turn to the competitor but the spillover demand is decreased by the parameter $\alpha_{pf}^{-a,a}$. Under cooperation this issue does not exist and all demand is considered. This also leads to those products not affected by competition being able to access the extended capacities in the cooperative case and raising the overall load factor by using the partner's capacities. With both competitors charging identical prices for a product, $\alpha_{pf}^{-a,a} = 1$ for all products, and CI = 1, the NE/C and NC/C ratios would turn out even closer to 1.

This case was considered in Grauberger and Kimms (2014b). There, in 105 of 120 considered instances, the summed payoffs in the NE were exactly as high as the cooperative payoff. If a difference did exist, it amounted to less than 1%. Apart from setting $\alpha_{pf}^{-a,a} = 1$ for all products, and CI = 1, Grauberger and Kimms (2014b) had a different setting like smaller networks, equal prices, and less instances. In order to be able to compare the results of the exact Algorithm 5.1 with the heuristic from Algorithm 5.2, we applied the heuristic with the same test bed used for the exact algorithm. The results are presented next.

5.4 Computing an Approximate Nash Equilibrium

Recall that we allowed for the exact algorithm from the previous section to terminate without finding an NE. This case can occur either if an imposed limit for the computation time is reached or if a game does not possess a pure NE. Indeed, Netessine and Shumsky (2005, Sec. 4) have proved that a pure NE might not exist in the case of the single-leg, two-class capacity control problem under competition between two airlines. In their example a high negative correlation between low-fare and high-fare demand was responsible for the absence of a pure NE. Since we consider more general network games, we next show that in such games a pure NE might not exist either under quite practical circumstances. It was already pointed out that a game does not possess a pure NE if the players' models have unique optimal solutions and for all starting strategies the iterative search ends in a loop with more than one iteration lying between the first and the second time a best response was chosen. Here, we illustrate with an example, which constellations of capacities and demands lead to such a result. Consider the following example for this.

Assume that two competing airlines each operate a network with one hub and six spoke airports as depicted in Figure 5.10. Each competitor has three origin airports O1, O2, and O3 as well as three destination airports D1, D2, and D3 (for player 1; D4 for player 2). We concentrate here on only six relevant itineraries per player to keep the example small. We also ignore fare classes and thus will use the words "product" and "itinerary" interchangeably in this example. The relevant six itineraries are O1–D1, O1–D2, O2–D2, O2–D3, O3–D3, and O3–D1 for player 1 as well as O1–D4, O1–D2, O2–D2, O2–D1, O3–D1, and O3–D4 for player 2, respectively. The other itineraries which could possibly be offered in these networks (e.g. O3–D2 or a product with the hub as a final destination) are considered irrelevant here, be it due to a lack of demand, too low revenues, or both. The relevant products are indicated by different colors and/or patterns of the arrows to distinguish them from each other and to distinguish the products affected by competition (black arrows) from those not affected by competition (red arrows for player 1 and green arrows for player 2, respectively). I.e. the product O1–D1, indicated by solid red arrows in player 1's network, is not affected by competition, while the product O1–D2, indicated by solid black arrows in both players' networks, is affected by competition. So are the products O2–D2 (dashed) and O3–D1 (dotted).



a) Network for Player 1 b) Network for Player 2

Figure 5.10: Networks for the Example without a Pure Nash Equilibrium

The networks in Figure 5.10 also make clear the complexity in airline network capacity control due to the interdependence of capacity occupied by different products. Assume that both players have only one (remaining) capacity unit on each of the aircraft employed for the relevant products. Then, each player can accept at most three of the relevant products; O1–D1, O2–D2, and O3–D3 or O1–D2, O2–D3, and O3–D1 for player 1 as well as O1–D4, O2–D2, and O3–D1 or O1–D2, O2–D1, and O3–D4 for player 2. This is due to a request accepted for one product (e.g. O1–D2) affecting the decision for two further products (O1–D1 and O2–D2 for player 1 or O1–D4 and O2–D2 for player 2). In practice, an accepted request might surely affect the decisions for more than only two products.

The players' demands, represented in Table 5.6 for player 1 and 5.7 for player 2, all occupy only one seat on each of the resources needed to serve the respective product. The products affected by competition are marked bold here. For these products, we assume that all passengers denied by a player request the same product from the competitor with certainty.

Product	O1–D1	**O1–D2**	**O2–D2**	O2–D3	O3–D3	**O3–D1**
Demand	1	0	0	1	1	1
Revenue	70	60	50	50	60	60

Table 5.6: Data for Airline 1 in the Example without a Pure Nash Equilibrium

Allocating capacities non-competitively, player 1 would accept the requests for products

Product	O1–D4	**O1–D2**	**O2–D2**	O2–D1	**O3–D1**	O3–D4
Demand	1	1	1	1	0	1
Revenue	60	60	50	50	60	50

Table 5.7: Data for Airline 2 in the Example without a Pure Nash Equilibrium

O1–D1 and O3–D3, while player 2 would accept the requests for the products O1–D2, O2–D1, and O3–D4. However, taking competition into account, player 2 would accept the requests for the products O1–D4, O2–D2, and O3–D1 with the request for the the product O3–D1 spilling from player 1. This would generate player 2 a revenue of 170; 10 more than accepting the other combination of products. As a consequence, the request for the product O1–D2 would be denied by player 2 and the customer would turn to player 1. This request, on the other hand, would be accepted by player 1 together with the requests for the products O2–D3 and O3–D1. Hence, player 1 would no longer deny the request for the product O3–D1 because he would need it for maximizing his own revenue. However, without the spillover demand for product O3–D1, player 2 would have to accept the same requests as in the non-competitive case (O1–D2, O2–D1, and O3–D4) denying the request for O2–D2. This, finally would close a circle with player 1 accepting this request and denying the request for his product O3–D1 again. Searching for best responses iteratively, the course of events would turn out as depicted in Figure 5.11.

Iteration	1	2
Requests accepted by Player 1	O1–D1, O2–D2, O3–D3	O1–D2, O2–D3, O3–D1
Requests accepted by Player 2	O1–D4, O2–D2, O3–D1	O1–D2, O2–D1, O3–D4

Figure 5.11: Course of Events in the Example without a Pure NE

In this example, the crucial factor responsible for the absence of a pure NE was that each player had a demand for a product affected by competition which was absent at the competitor. Further, the product O3–D1 can be seen as the decisive product since it increased the players' revenues in two different combinations (O1–D2 and O3–D1 for player 1 and O2–D2 and O3–D1 for player 2). Especially, it lead to a player accepting an originally denied request after the competitor denied a product which again was reversed after the first player accepted this spillover demand. This demand structure in combination with unique best responses for both players lead to no pure NE being existent in the above network RM game.

Having Algorithm 5.1 terminate without a pure NE would leave an airline facing the problem of optimizing its booking limits without any better suggestion than the non-competitive model which ignores competition. For this case, this section describes a heuristic to find at least approximate NE and be better off than in the non-competitive solution. The heuristic is considerably easier than the exact algorithm because it only consists of the forward search and does not need to manipulate the players' models or to do a backtracking procedure. Thus, it a least provides a help to be no worse off than in the non-competitive case within a short computation time.

In this context, the notion of equilibrium defined in Section 4.1 will be called *exact NE* in order to distinguish it from *approximate NE*. Finding the latter (also called ϵ-approximate NE) in bimatrix games nowadays makes up a considerable part of the computational research about NE since computing exact NE is computationally hard, see Chen et al. (2009). In order to measure the performance of algorithms computing ϵ-approximate NE and to be able to compare them, all entries in the payoff matrices are normalized to be in the range between 0 and 1. This can be done through multiplication and/or addition of appropriate (but equal) terms to all of a matrix's entries without changing the structure of the game or the structure or set of NE. See e.g. Kontogiannis et al. (2009, Sec. 2.4) for a formal proof. An ϵ-approximate NE can then be defined as a state of the game in which no player a has an incentive to deviate from his current strategy $s^a_{\epsilon-NE}$ if no other strategy s^a_i can yield him a larger utility than the current one plus a constant $\epsilon \geq 0$. Thus, $s^a_{\epsilon-NE}$ and $\mathbf{s}^{-\mathbf{a}}_{\epsilon-\mathbf{NE}}$ define an ϵ-approximate NE if $u^a(s^a_{\epsilon-NE}; \mathbf{s}^{-\mathbf{a}}_{\epsilon-\mathbf{NE}}) + \epsilon \geq u^a(s^a_i; \mathbf{s}^{-\mathbf{a}}_{\epsilon-\mathbf{NE}})$ for all players a and all strategies s^a_i, see e.g. Daskalakis et al. (2009b) as well as Kontogiannis et al. (2009). Following this definition, an exact NE can be described as an ϵ-approximate NE with $\epsilon = 0$. A motivation for $\epsilon-$approximate NE is that there might be a cost c involved with deviating from one strategy to another. If $c \geq \epsilon$ then deviating from the strategy $s^a_{\epsilon-NE}$ does not improve player a's net utility and leaves him with the current approximate NE.

Although ϵ-approximate NE are not necessarily exact, the algorithms to compute them only need polynomial running times opposed to exponential running times needed by algorithms that are proved to find exact NE (see e.g. Daskalakis et al., 2009b, Kontogiannis et al., 2009 as well as Tsaknakis and Spirakis, 2007). Hence, approximate NE constitute a compromise which sacrifices a part of the players' payoffs in an exact NE for a fast computation time. The best approximation to an exact NE in general two-player games known to date is an algorithm by Tsaknakis and Spirakis (2007) computing 0.3393-approximate NE. This means that, solving a general bimatrix game with their algorithm, every player's utility in the game's solution will be at most 33.93% worse than his utility in an exact NE. Daskalakis et al. (2007) as well as Daskalakis et al. (2009b) presented simpler algorithms in which the gap between the approximate and an exact NE is larger. Hémon et al. (2008) introduced approximations for multi-player games. See also Bosse et al. (2010), Kontogiannis et al. (2009) as well as Spirakis (2008) for reviews and simplifications of some of the mentioned results.

A more restrictive approximate NE is an *ϵ-well supported NE* which allows the players to play strategies that are only approximately best responses. See e.g. Daskalakis et al. (2009a) as well as Kontogiannis and Spirakis (2007a,b). Panagopoulou and Spirakis (2007) presented an algorithm for computing approximate and well-supported NE for bimatrix games in which the players' payoffs are random variables.

5.4.1 Our Notion of Approximate Nash Equilibria

In this section we will introduce our own notion of approximate NE because the ϵ-approximate NE described in the previous subsection have a further operational disadvantage—besides resulting in a possibly worse utility for the players than exact NE. In particular, the algorithms used for computing ϵ-approximate NE need the complete payoff matrices as input—like the algorithms used for computing exact NE. However, for the network revenue management problem with competition setting up the complete payoff matrices can be a very time and memory consuming task itself.

Alone the number of strategies available to a player in the real-sized games considered here amounts to several millions due to the extensive number of possible combinations. Even if one were to reduce the set of all possible strategies to very relevant ones (e.g. best responses, optimal solutions etc.), this number of strategies remains huge. Note that in the setting considered here the strategy sets are matrices themselves displaying the combinations of booking limits for the different products given airplane capacities. Not only setting up millions of strategies but also picking the relevant ones would take a considerable amount of time and memory. Setting up the payoff matrices afterwards would be exhausting as well since all combinations of the players' strategies would have to considered, i.e. millions of millions of entries would have to be computed and saved, once for each payoff matrix. However, even assuming this can be done in an acceptable time in practice, one would still need to apply an algorithm to compute an NE from the payoff matrices which would be beyond all constraints since the computational burden of such algorithm rises exponentially with the size of the game.

Although computing an ϵ-approximate NE would take less time than computing an exact NE, one would not even come so far in many cases due to the large amount of time and memory needed to set up the payoff matrices based on large amounts of strategies. Since, to the best of our knowledge, no efficient algorithm for computing NE in network revenue management games exists, we developed our own notion of approximate NE and our own heuristic to compute them.

Operationally, our heuristic is better than algorithms computing ϵ-approximate NE because it does not need the players' complete payoff matrices as input. In the course of our heuristic only the necessary entries of the matrices (the best responses) are computed "on demand" which is a lot more efficient and thus better suited to solve real-world problems. However, as opposed to ϵ-approximate NE that are proved to not bring a player a higher payoff than the current payoff plus ϵ by deviating to a different strategy, we cannot specify such a value for our results from the beginning.

In the course of our heuristic a part of the players' strategies is eliminated. Afterwards, only the game based on the strategies remaining available to the players after the elimination are considered. We will call this game *reduced game*. Finally, we compute the NE in the reduced game. We will show that pure NE in the reduced game are exact pure NE in the original game as well while the reduced game's mixed NE form the *approximate NE* for the original game. While mixed NE meet the conditions for being equilibria in the sense that none of the players can deviate and raise his utility, in practice only one definite vector of booking limits (i.e. one of the pure strategies entailed in the mixed one) can be implemented. Hence, with a mixed NE, the players must operate some generator of random values with the probability distribution from the mixed strategy to pick the actual pure strategy to implement. I.e. an airline implementing the heuristic proposed below in its RM system could employ a random value generator connected to the heuristic which, with this airline's probability distribution from the mixed NE, proposes one definite pure strategy to implement in practice.

Note, however, that the above definition of approximate NE holds only for this section and is only made for distinguishing the results from our heuristic from ϵ-approximate NE. In Chapter 6, we will propose an algorithm for approximating NE in RM games with simultaneous price and quantity competition. There, no reduction of the game will be performed, yet the results might still be non-exact NE.

5.4.2 Heuristic Approach

The reason we developed our heuristic was that to the best of our knowledge no algorithm exists that can find an exact NE in revenue management games with the large size considered in this thesis efficiently. Finding exact equilibria based on the complete payoff matrices of games with large strategy spaces and thus payoff matrices is a (computationally) hard and time consuming procedure. We provide a new heuristic for approximating NE. Existing approaches with the same purpose are not satisfying since they—like exact algorithms—rely on the whole set of strategies and on the whole payoff matrices.

Our heuristic starts with an iterative search for best responses like the approach by Jiang and Pang (2011). However, their algorithm ends with the termination of their iterative search which relies on a predefined stopping criterion. There, if the difference in booking limits of two consecutive iterations is at most as high as a predefined value, the players' most current pure strategies are taken to build an approximate NE. However, working with a predefined convergence criterion is risky because one has to take a guess about this value which can be too far away from or too close to the optimum. The consequence is a possible loss in revenue due to too early termination or the failure to terminate at all. In the former case the algorithm would stop although a higher payoff is realizable while in the latter case a (too tight) convergence criterion might lead to an endless loop because it will never be reached.

For this reason our terminating condition is different. Our iterative search terminates when a player's best response is chosen a second time. Also our approximate NE is not formed directly after the iterative search, as is the case with the algorithm by Jiang and Pang (2011). Instead, we reduce the game after the iterative search and compute an approximate NE for the original game within the reduced game. This makes up the essence of our heuristic. Jiang and Pang (2011, p. 316) pointed out that their algorithm converges for "normal" Nash games based on the PNLP "under suitable conditions". Note that such suitable conditions need not prevail for the PNLP in general, so it will not necessarily converge to an exact NE in general. Furthermore, the authors could not prove convergence of their algorithm for generalized Nash games based on the DLP and thus left this issue open for further research. Our heuristic is more general and does not depend on the type of model.

Reductions of the original game have been proposed before (see e.g. Gale et al., 1950 as well as Conitzer and Sandholm, 2006). The former authors gave conditions for reducing a zero-sum game payoff matrix. After computing the solution for the reduced game the solution for the original game could be given. Conitzer and Sandholm (2006) used a technique to reduce a bimatrix game and to find an exact NE for the original game by finding one in the reduced game. However, it was only applicable to certain games where, for a fixed strategy of a player, certain strategies give the opponent an equal payoff—like in some of the applications in Gale et al. (1950). In contrast to the mentioned reduction algorithms our reduction scheme is applicable to general games.

Our heuristic is displayed in Algorithm 5.2 and works as follows. First, the players search for best responses iteratively. In each iteration first one player chooses his best response to the other's best reaction from the previous iteration then the second player reacts to this best response. The iterative search stops when a player's best response is chosen a second time which would lead to a best response chosen before by the competitor. If a player's best response is chosen twice in two consecutive iterations, a pure, exact NE is found since the players' strategies are best responses to each other.

Note that until here, the heuristic functions exactly like the algorithm for computing exact pure NE. The approaches differ in their treatment of repeated strategies. While in the exact Algorithm 5.1 the backtracking procedure is initiated if a strategy is repeated after more than one iteration, in the heuristic we reduce the players' strategy sets to the strategies chosen as best responses repeatedly and set up a (reduced) bimatrix game based on these strategies. All other strategies are eliminated from the players' strategy sets. The original game's (approximate) NE is selected as one of the (possibly mixed) NE in the reduced game. Note that this NE can also be exact. Indeed, in the next section we reformulate the competitive DLP and show that with this reformulated model a pure NE in the reduced game is also an exact pure NE for the original game. However, this cannot be said for mixed NE in the reduced game since we only compute an NE in an excerpt of the whole game. Thus we call the mixed NE from the reduced game an approximate NE for the original game.

In the reduced game, the payoff r_{ij}^a for player a when he chooses his strategy from iteration i and the opponent chooses his strategy from iteration j is determined as in Equation (5.14). Here, $Loop^a$ and $Loop^{-a}$ stand for the sets of iterations in the loop, distinguished by players. This distinction is necessary because if the starting player finds a loop, the iterations involved are different (see e.g. Figure 5.1b) than if the second player finds one (as in e.g. Figure 5.2a).

$$r_{ij}^a = \sum_{p \in A} \sum_{f=1}^{\mathcal{F}} \pi_{pf}^a * \min\{B_{i,pf}^a; d_{pf}^a + \lfloor \alpha_{pf}^{-a,a} (d_{pf}^{-a} - B_{j,pf}^{-a})^+ \rfloor\} + \sum_{p \in NA^a} \sum_{f=1}^{\mathcal{F}} \pi_{pf}^a * B_{i,pf}^a$$
$$i \in Loop^a, j \in Loop^{-a} \quad (5.14)$$

A player's entry in the utility matrix of the reduced game is made up of the player's product price multiplied by the minimum of his booking limit in the corresponding iteration and his total demand for the respective product for products affected by competition. Recall here that B_i^a stands for player a's vector of booking limits in iteration i. The total demand is again the rounded down value of the original demand and the spillover demand from the competitor. For products not affected by competition, the price is simply multiplied by the player's booking limit.

The selection of the reduced game's final solution is based on the *payoff dominance* and *risk dominance* of certain NE similar to the dominance definitions by Harsanyi and Selten (1988). An NE *E1* payoff-dominates an NE *E2* if for both players $a \in \{1, -1\}$ the utility in *E1* is not worse than in *E2*, i.e. $u^a(E1) \geq u^a(E2)$ and at least one player's payoff in *E1* is strictly higher than in *E2*, i.e. $u^a(E1) > u^a(E2)$. For the risk dominance, assume that in the game formed by the matrices in Table 5.8 there are two pure (payoff dominant) NE (s_1^1, s_1^2) and (s_2^1, s_2^2) with payoffs (A,a) and (D,d). Here the capital letters and the lower-case letters stand for the first player's and the second player's payoffs, respectively. The latter equilibrium risk-dominates the former if $(C - D)(c - d) \geq (B - A)(b - a)$.

r^1	s_1^2	s_2^2
s_1^1	A	C
s_2^1	B	D

a) Payoff Matrix for Airline 1

r^2	s_1^2	s_2^2
s_1^1	a	b
s_2^1	c	d

b) Payoff Matrix for Airline 2

Table 5.8: Payoff Matrices for Risk Dominance

In other words, *E1* risk-dominates *E2* if deviation from the former by both players would lead to at least an equal loss to the players as deviating from the latter. Note that our dominance conditions are weaker than those in Harsanyi and Selten (1988) where the inequalities are required to be strict. Our heuristic is described in Algorithm 5.2.

```
1: k ← 1
2: choose a = a_0
3: choose B_0^{-a}
4: loop
5:     Update b^{-a}
6:     Solve M^a
7:     if r^a ∈ R^a then
8:         if b^a was chosen in an iteration j with j ∈ I^a(r^a) then
9:             Form the Best Response Matrices (BRMs)
10:            if The BRMs have one entry each then
11:                The last two chosen best responses constitute an exact pure NE
12:                Stop!
13:            else
14:                Compute all pure NE in the game based on the BRMs
15:                if No pure NE was found then
16:                    Compute all extreme mixed NE in the game based on the BRMs
17:                end if
18:                if More than 2 NE were found then
19:                    Sort the NE in an arbitrary but fixed order
20:                    Eliminate all payoff-dominated NE from the list
21:                    if More than 2 NE remain then
22:                        Eliminate all risk-dominated NE from the list
23:                    end if
24:                end if
25:                Pick the remaining NE as the original game's solution
26:                Stop!
27:            end if
28:        end if
29:    end if
30:    R_k^a ← r^a
31:    Add k to I^a(r^a)
32:    B_k^a ← b^a
33:    a ← a * (−1)
34:    if a == a_0 then
35:        k ← k + 1
36:    end if
37: end loop
```

Algorithm 5.2: Algorithm for Computing an Approximate Pure Nash Equilibrium

Note that with the heuristic, a player's model will be solved optimally with certainty, independent of the competitor's behavior, and hence every strategy of a player leads to a best response of the competitor. Hence, a best response will be chosen twice within finite time which means that our heuristic terminates within finite time. In order to include more than two players, the same extensions as with the exact algorithm from the previous section must be made in those parts of the heuristic which are identical with the exact algorithm. With more than two players, the reduced games will be based on more than tow matrices, though which must be set up appropriately and need appropriate algorithms to find NE in them. Particularly, for the matrix entries, Equation (5.14) will have to be extended to multiply the price of a product affected by competition with the minimum of the product's booking limit and the sum of spill

over demand in addition to the player's additional demand. Furthermore, the concept of risk dominance must be extended to cover three or more players.

However, we cannot say a priori what kind of Nash equilibrium—an exact or an approximate one—will be found with our heuristic because this result depends on the chosen starting strategy. See also the examples in Sections 5.4.3 and 5.4.4. Hence, we cannot prove the convergence to exact pure NE for our heuristic. However, even for algorithms that provably find an exact NE and terminate after finding only one NE (e.g. the algorithms described in Lemke and Howson, 1964 or McKelvey and McLennan, 1996) it cannot always be said a priori what type of Nash equilibrium they will find—a pure or a mixed one—because this result also depends on their starting strategy.

5.4.3 Numerical Example

In this section, we step through the heuristic, explain how the (reduced) payoff matrices come about, and how the approximate NE is computed. For the example we picked an instance for the computational study below. The mean demand was assumed to be $\mu = 6$, and the players were assumed to have networks with one hub and 40 spoke airports with 50% of their products affected by competition. The perturbation was "increasing by fare class, then alphabetic by O&D name". We chose a bimatrix representation of the game to visualize the growing matrices and the game's reduction. We point out the heuristic's behavior in some special games in the next section.

Lines 1 – 4 of the heuristic are identical to Algorithm 5.1. The next line is different, though. While in Algorithm 5.1 it was checked whether the opponent's booking limits were already updated in the current iteration, this need not be done in the heuristic because no backtracking procedure is performed. Hence, in line 5 the model must be updated with certainty followed by solving the model. Without the backtracking in the heuristic, there is also no need to add or drop any constraints. Without additional constraints, the model will be solved optimally with certainty so that we need not check this in the heuristic but can proceed with comparing the payoffs and strategies with the previous ones in lines 7 and 8, respectively. Since in the first iterations these do not match any previous value, the heuristic jumps to line 30 where the current payoff is saved in the vector $\mathbf{R}^a$ followed by adding the current iteration to the set of iterations $\mathbf{I}^a(r^a)$ with the same payoff and saving the best response in vector $\mathbf{B}^a$ in lines 31 and 32, respectively. Afterwards, the player is changed in line 33, the iteration count is augmented if necessary in line 35, and the loop repeats. This steps are also performed in the exact algorithm.

In this manner the heuristic iterates 5 times, until player 2 chooses the same strategy as in iteration 3. The payoff matrices after the first iteration are displayed in Table 5.9.

r^1	0
s_1^1	4,215,672

a) Payoff Matrix for Airline 1

r^2	0	s_1^2
s_1^1	0	4,171,614

b) Payoff Matrix for Airline 2

Table 5.9: Payoff Matrices After the First Iteration

Since we only search for best responses in each step, only one entry is added to each payoff matrix in each iteration. Note that the numbering of the strategies displayed says nothing about the order they would be set up if one wanted to enumerate all strategies. The index of a best

response simply shows the order (in this case the iteration) in which this strategy was chosen by a player. After the second iteration the matrices look as in Table 5.10.

r^1	0	s_1^2
s_1^1	4,215,672	–
s_2^1	–	4,181,923

a) Payoff Matrix for Airline 1

r^2	0	s_1^2	s_2^2
s_1^1	0	4,171,614	–
s_2^1	0	–	4,167,128

b) Payoff Matrix for Airline 2

Table 5.10: Payoff Matrices After the Second Iteration

Table 5.11 shows the players' payoff matrices after the third iteration. In this manner the matrices are filled with payoffs one iteration after the other.

r^1	0	s_1^2	s_2^2
s_1^1	4,215,672	–	–
s_2^1	–	4,181,923	–
s_3^1	–	–	4,181,424

a) Payoff Matrix for Airline 1

r^2	0	s_1^2	s_2^2	s_3^2
s_1^1	0	4,171,614	–	–
s_2^1	0	–	4,167,128	–
s_3^1	0	–	–	4,166,670

b) Payoff Matrix for Airline 2

Table 5.11: Payoff Matrices After the Third Iteration

In iterations 4 and 5 the optimal values of the objective functions approach their equilibrium values ever closer, so we omit showing the growing matrices in these iterations. In this iteration, player 2 chooses s_3^2 as his best response to strategy s_5^1, so the heuristic proceeds from line 8 to line 9 where the best response matrices (BRMs) are set up. The BRMs consist of the payoffs with the strategies player 1 chose in iterations 4 and 5 and player 2 chose in iterations 3 and 4 and are shown in Table 5.12. The entries that were not found by the iterative search were added as well.

r^1	s_3^2	s_4^2
s_4^1	4,181,417	4,180,679
s_5^1	4,181,323	4,181,323

a) BRM for Airline 1

r^2	s_3^2	s_4^2
s_4^1	4,166,670	4,166,824
s_5^1	4,166,670	4,165,422

b) BRM for Airline 2

Table 5.12: Best Response Matrices

In line 11 of the heuristic it is checked whether the BRMs have only one entry each which would mean that an exact pure NE was found leading to terminating the procedure right afterwards. Since the BRMs consist of more than one entry each, a pure NE was not found in the iterative search. Hence, we first seek all pure NE of the reduced game (e.g. with the Gambit software by McKelvey et al., 2014) in line 14. The reduced game does not have any pure NE, so we proceed to line 16 to compute all extreme mixed NE of the reduced game (recall from Section 4.3 the definition of extreme NE as being extreme points of convex combinations of a set of NE).

Note that the two steps of computing all pure NE and all mixed NE were treated as one in Grauberger and Kimms (2014b) who proposed that all NE (pure and mixed) were seeked right

away in the reduced game. However, since we prefer pure NE over mixed ones, we can reduce the computational burden by looking for the former first. The Gambit software provides specified algorithms for this. Only if no pure NE can be found, we search for mixed NE in the reduced game. Note also that an exact pure NE found in the iterative approach is expressed here as the payoff matrices having only one entry each. While the exact algorithm differentiates pure NE explicitly, the heuristic relies on the BRMs as its core elements and hence we can describe the same aspect in a different manner.

The reduced game has only one, mixed NE as Table 5.13 shows. The entries in the second through fourth columns show the ratios with which the players' strategies are played in the respective NE.

NE	s_4^1	s_5^1	s_3^2	s_4^2	Payoff Player 1	Payoff Player 2
1	$624/701$	$77/701$	$322/369$	$47/369$	4,181,323	4,166,670

Table 5.13: Nash Equilibrium in the Reduced Game

Since this NE is the only one in this game, line 16 of the heuristic leads to line 25 where this NE is chosen as the original game's approximate solution. The heuristic terminates in the next line. In this approximate NE, merely the payoff for the first player is a little lower than the original game's exact NE computed with Algorithm 5.1 ($r^1 = 4,181,397$ and $r^2 = 4,166,670$). The second player's payoffs in the approximate NE is even as high as the payoff in the exact one.

If more than two NE were found in the reduced game, though, the procedure would follow lines 19 – 23. Here, the NE would be sorted in an arbitrary but fixed order before the payoff dominated and, if necessary, the risk dominated NE would be eliminated so that only one NE with the highest payoffs for both players remains to be chosen as the original game's solution. Note for the elimination of risk-dominated NE that since the NE are sorted in an arbitrary order, every pair i, j of NE must be checked. Here, $i \in \{1, \ldots, n-1\}$, $j \in \{i+1, \ldots, n\}$, $i < j$, and n denotes the number of NE in the reduced game.

5.4.4 Special Types of Games

Consider again the game described above and replicated in Table 5.14. While the exact Algorithm 5.1 finds the pure NE with certainty (possibly after forbidding one or more of the "wrong" starting strategies), with the heuristic from Algorithm 5.2, the pure NE will only be found with a probability of $\frac{1}{3}$. This probability corresponds to the relation of the two pure strategies leading to the pure NE if chosen as starting strategies in the iterative search to the total number of strategies (six). Thus, our heuristic might fail to find a game's exact pure NE although one exists. The mixed NE found with the heuristic is an exact mixed NE, though.

However, our heuristic is not the only procedure to suffer from this issue. As was pointed out in Section 5.4.2, even algorithms that provably find an exact NE and terminate after finding only one NE would end up finding the mixed NE in the above game if one of the strategies constituting the mixed NE is chosen as the starting strategy for these algorithms. In order to raise the probability of finding an exact pure NE, one could use the exact Algorithm 5.1 or rerun the heuristic with different starting strategies. However, since one does not know the payoff matrices' structure at the beginning, rerunning the procedures might not lead to an

r^1	s_1^2	s_2^2	s_3^2
s_1^1	2	1	0
s_2^1	1	2	0
s_3^1	0	0	3

a) Payoff Matrix r^1

r^2	s_1^2	s_2^2	s_3^2
s_1^1	1	2	0
s_2^1	2	1	0
s_3^1	0	0	3

b) Payoff Matrix r^2

Table 5.14: Payoff Matrices for the Exemplary Game

improvement. This is e.g. the case when there are very many strategies but only few of them are best responses that lead to a pure NE like in the game from Table 5.14 or when no pure NE exists at all like in the game from Section 5.4. The payoff matrices for this game are displayed in Table 5.15. Here, the strategy s_1^1 stands for player 1 accepting the requests for O1–D1, O2–D2, and O3–D3, while s_2^1 stands for this player accepting the requests for O1–D2, O2–D3, and O3–D1. The strategy s_1^2 by player 2 accepts requests for products O1–D4, O2–D2, and O3–D1, while his strategy s_2^2 accepts requests for products O1–D2, O2–D1, and O3–D4. These are the relevant strategies forming the loop.

r^1	s_1^2	s_2^2
s_1^1	130	180
s_2^1	170	110

a) Payoff Matrix for Player 1

r^2	s_1^2	s_2^2
s_1^1	170	160
s_2^1	110	160

b) Payoff Matrix for Player 2

Table 5.15: Payoff Matrices for the Example without Pure NE

Since in this game both players each have only two strategies, the heuristic would find the only (exact) mixed NE with Player 1 playing his first and second strategy with a probability of $\approx 83.33\%$ and $\approx 16.67\%$, respectively and Player 2 using his first and second strategy in $\approx 63.64\%$ and $\approx 36.36\%$ of the cases, respectively. However, there are also games in which our heuristic would even fail to find an exact mixed NE. Such a game is displayed in Table 5.16.

r^1	s_1^2	s_2^2
s_1^1	1	5
s_2^1	3	4
s_3^1	5	2

a) Payoff Matrix for Player 1

r^2	s_1^2	s_2^2
s_1^1	3	1
s_2^1	3	2
s_3^1	3	4

b) Payoff Matrix for Player 2

Table 5.16: Exemplary Payoff Matrices

In the course of the heuristic the first player's strategy s_2^1 would be eliminated because it is never a best response to any of the second player's strategies. However, in the only exact NE it is played with a probability of 50%, as Table 5.17 shows. Here, the entries in the parentheses in the column "Payoff Player 1" stand for the decimal values of the fractions to make these comparable.

The approximate NE leads to a lower payoff for player 1 than in the exact one. Yet, a lower payoff in an inexact NE is still acceptable in very large games such as ours if the players'

NE	s_1^1	s_2^1	s_3^1	s_1^2	s_2^2	Payoff Player 1	Payoff Player 2
Exact NE	0	1/2	1/2	1/2	1/2	7/2 (3.5)	3
Approximate NE	1/3	–	2/3	3/7	4/7	23/7 ($\approx$ 3.3)	3

Table 5.17: Exact and Approximate Nash Equilibria in a Game without a Pure Equilibrium

strategies can be reduced after the iterative search and the heuristic terminates within a short time. This way, an airline in practice can at least count on an approximate result which is computed in an acceptable time and never leads to lower payoffs than the non-competitive single-airline DLP (2.2) – (2.5). The computational study in the next section will reveal this advantage of the heuristic in cases where no pure NE was found by the exact algorithm. The payoffs in these cases are still higher than those from the single-airline model (2.2) – (2.5) while the computation times are very low so that the heuristic can indeed be employed right after the exact algorithm if the latter terminates without an exact pure NE. However, if a game's original payoff matrices have a "diagonally shifted" structure—as the extreme example in Table 5.18 shows—no significant reduction of the original game can be performed in line 9 of the heuristic in Algorithm 5.2.

r^1	s_1^2	$\cdots$		s_n^2
s_1^1	1	0	0	0
$\vdots$	0	1	0	0
	0	0	1	0
s_m^1	0	0	0	1

a) Payoff Matrix for Player 1

r^2	s_1^2	$\cdots$		s_n^2
s_1^1	0	1	0	0
$\vdots$	0	0	1	0
	0	0	0	1
s_m^1	1	0	0	0

b) Payoff Matrix for Player 2

Table 5.18: Diagonally Shifted Payoff Matrices

In the worst such cases, every strategy is a best response to exactly one of the competitor's strategies and the best-response matrices correspond to the original payoff matrices. Depending on the size of the BRMs, one could compute the (exact) mixed NE from them or artificially reduce them by picking a subset of the chosen best responses during the iterative search.

5.4.5 Reformulated Competitive DLP

Note that the game based on the model (5.1) – (5.5) is a so-called generalized Nash game in which, according to Restriction (5.3), the strategies (booking limits) available to the players depend on the competitor's most recent behavior. Thus, if one strategy of player $-a$ leads to a spillover demand of, say 2 for a product, then player a's total demand for this product is $d_{pf}^a + 2\alpha_{pf}^{-a,a}$. If, however this spill amounts only to 1 due to player $-a$ playing a different strategy, player a's total demand for this product shrinks to $d_{pf}^a + \alpha_{pf}^{-a,a}$. That means that in a generalized game the strategies available to the players are not necessarily always the same leading to the game based on the BRMs possibly including players' strategies that might not be available in combination with certain strategies of the competitor. Strictly speaking, this eventually leads to possible pure NE in the reduced game to be only approximate NE since the corresponding pure strategies might not exist as a combination.

In order to avoid this issue and make sure that a possible pure NE in the reduced game based on the BRMs is indeed an exact pure NE for the original game, we can reformulate the above model for a "normal" game. For, in reality RM games are not generalized games because the players always have the same strategies available. In reality, nothing hinders an airline to set the booking limit for a product higher than its total demand. It can do so, but must take into account that in this case $b^a_{pf} - (d^a_{pf} + \alpha^{-a,a}_{pf}(d^{-a}_{pf} - b^{-a}_{pf})^+)$ seats remain empty.

In order to model a "normal" game in which the players' strategies always remain the same, we redefine the bounds for the booking limits to a fixed lower bound LB^a_{pf} and a fixed upper bound UB^a_{pf}. These are the same parameters used for the indexes in Constraints (5.6) and (5.7) where the additional binary variables are defined for forbidding repeated strategies in the exact algorithm. In this case, one has to make sure that in the objective function only booking limits at most as high as the total demand for a product, i.e. $\min\{b^a_{pf}; d^a_{pf} + \alpha^{-a,a}_{pf}(d^{-a}_{pf} - b^{-a}_{pf})^+\}$ contribute to the total revenue. However, writing this term in the objective function would make it non-linear and the model hard to solve.

In order to keep the objective function linear, we introduce an auxiliary variable v^a_{pf} for player a's excess supply for the products affected by competition. This excess supply is punished in the new objective function (5.15) by multiplying it with a big number N^a and subtracting it from the total revenue. This formulation effectively avoids excess supply in an optimal solution. Airline a's model $\mathcal{M}^a_{Normal}$ for the normal game looks as follows:

$$\mathcal{M}^a_{Normal}: \max \quad r^a(b^a_{pf}, b^{-a}_{pf}) = \sum_{p=1}^{\mathcal{P}^a}\sum_{f=1}^{\mathcal{F}} \pi^a_{pf} b^a_{pf} - N^a \sum_{p\in A}\sum_{f=1}^{\mathcal{F}} v^a_{pf} \tag{5.15}$$

$$\text{subject to} \quad b^a_{pf} \le d^a_{pf} \qquad p \in NA^a, f \in F \tag{5.16}$$

$$b^a_{pf} \le UB^a_{pf} \qquad p \in A, f \in F \tag{5.17}$$

$$v^a_{pf} \ge b^a_{pf} - \lfloor (d^a_{pf} + \alpha^{-a,a}_{pf}(d^{-a}_{pf} - b^{-a}_{pf})^+) \rfloor \qquad p \in A, f \in F \tag{5.18}$$

$$\sum_{p=1}^{\mathcal{P}^a}\sum_{f=1}^{\mathcal{F}} M^a_{lpf} b^a_{pf} \le C^a_l \qquad l \in L^a \tag{5.19}$$

$$b^a_{pf} \ge LB^a_{pf} \ge 0 \qquad p \in P^a, f \in F \tag{5.20}$$

$$v^a_{pf} \ge 0 \qquad p \in A, f \in F \tag{5.21}$$

Restriction (5.16) is the same as Restriction (5.2) and limits the booking limit for a product not affected by competition to its expected demand. Restriction (5.17) limits the booking limit for a product affected by competition to UB^a_{pf}. With this formulation of an upper bound for the booking limits, we make sure that the strategies available to the players always remain the same. In Restriction (5.18) the player's excess supply is determined. The variable v^a_{pf} takes on a positive value when more seats are reserved than there is total demand for a product, i.e. if $b^a_{pf} > (d^a_{pf} + \alpha^{-a,a}_{pf}(d^{-a}_{pf} - b^{-a}_{pf})^+)$. This is prevented by the high value of N^a, though. Due to Restriction (5.21) the excess supply cannot be negative. Restriction (5.19) is the same as (5.4) and keeps track of a leg's capacity. In Restrictions (5.20) and (5.21) the variables b^a_{pf} and v^a_{pf} are assumed to be continuous, non-negative variables. Note that this model is only a different way of modeling the decisions in the model $\mathcal{M}^a$ (5.1) – (5.5). Hence, with equal parameters in both models, the same optimal solution in terms of booking limit allocations will be found by both models.

With the reformulated model, we can show that a pure NE in the reduced game based on the BRMs is also an exact pure NE for the original game since the combinations of strategies

in the reduced game exist with certainty. For a proof consider the following game based on an instance from the computational study with a pure NE in the reduced game. In the instance, the demand was $\mu = 4$, the players employed networks with two hubs and 20 spoke airports connected to each hub, respectively. The perturbation order was "by fare class, then alphabetic by O&D name" and the competition factor amounted to 0.5. Since the building of the BRMs was illustrated in an earlier example, we skip this part of the heuristic here. It is only relevant to note that in iteration 6 player 1 chose the same strategy as in iteration 4 with the payoffs shown in Table 5.19.

r^1	s_4^2	s_5^2
s_4^1	–	3,586,942
s_5^1	3,586,956	–

a) Payoff Matrix for Player 1

r^2	s_4^2	s_5^2
s_4^1	3,589,382	–
s_5^1	–	3,589,382

b) Payoff Matrix for Player 2

Table 5.19: Payoff Matrices after Iteration 5

The course of this iterative procedure would look like the one in Figure 5.12. Here, the dotted arrow indicates the repeating strategy and closes the loop.

Iteration	4	5
Strategy Player 1	s_4^1	s_5^1
	↓	↗ ↓
Strategy Player 2	s_4^2	s_5^2

Figure 5.12: Exemplary Iterative Course

The BRMs for this game are depicted in Table 5.20. Here, one can see that the combination $(s_5^1; s_4^2)$ yields player 2 the same payoff as the combination $(s_5^1; s_5^2)$ which was chosen in the iterative search. This means that strategy s_4^2 is also a best response to strategy s_5^1.

r^1	s_4^2	s_5^2
s_4^1	3,586,414	3,586,942
s_5^1	3,586,956	3,586,688

a) Payoff Matrix for Player 1

r^2	s_4^2	s_5^2
s_4^1	3,589,382	3,588,852
s_5^1	3,589,382	3,589,382

b) Payoff Matrix for Player 2

Table 5.20: Exemplary Best Response Matrices

Hence, the strategy combination $(s_5^1; s_4^2)$ constitutes a pure NE in this reduced game. With the original model $\mathcal{M}^a$ made up of Equations (5.1) – (5.5), we could not be sure right away whether this is also an exact pure NE for the original game since the strategy s_4^2 might not be available to player 2 with player 1 playing strategy s_5^1. However, with the reformulated model $\mathcal{M}^a_{Normal}$ made up of Equations (5.15) – (5.21) we know that this strategy exists, irrespective of the competitor's behavior. Knowing that it yields the same payoff to player 2 like the "truly" chosen best responses during the iterative search, we know that this strategy is also a best response for the original game. Hence, the combination $(s_5^1; s_4^2)$ is also a pure NE for the original game based on the reformulated model $\mathcal{M}^a_{Normal}$. Thus, whenever two (or more) of player a's pure

strategies yield him the same maximal payoff in the reduced game, given one of the competitor's strategies, a pure NE involving one of player a's strategies in the reduced game is also an exact pure NE in the original game. This also applies to games with more than two players.

5.4.6 Computational Study

We next present results from the computational study with model (5.15) – (5.21) employed in the heuristic from Algorithm 5.2. In order to ensure comparability, the same test bed as for the exact algorithm from Section 5.3.5.1 was used. Later we show that our heuristic also allows for a derivation of a range ϵ in which the NE payoffs lie.

5.4.6.1 Results

Figures 5.13 – 5.15 show the average computation times for the heuristic. One can see that the results for a mean demand of $\mu = 2$ are similar to those from the exact algorithm. Yet, the times needed by the heuristic show a smoother increase without any "outliers" compared to the times for the exact algorithm.

The picture changes with larger values of mean demand, though. While here the average computation times in the small networks are still similar to those from the previous section, the larger networks (60 and more spoke airports) show drastically shorter average computation times. Especially the instances with the largest networks could be solved much faster. The results for the heuristic again show a smoother behavior.

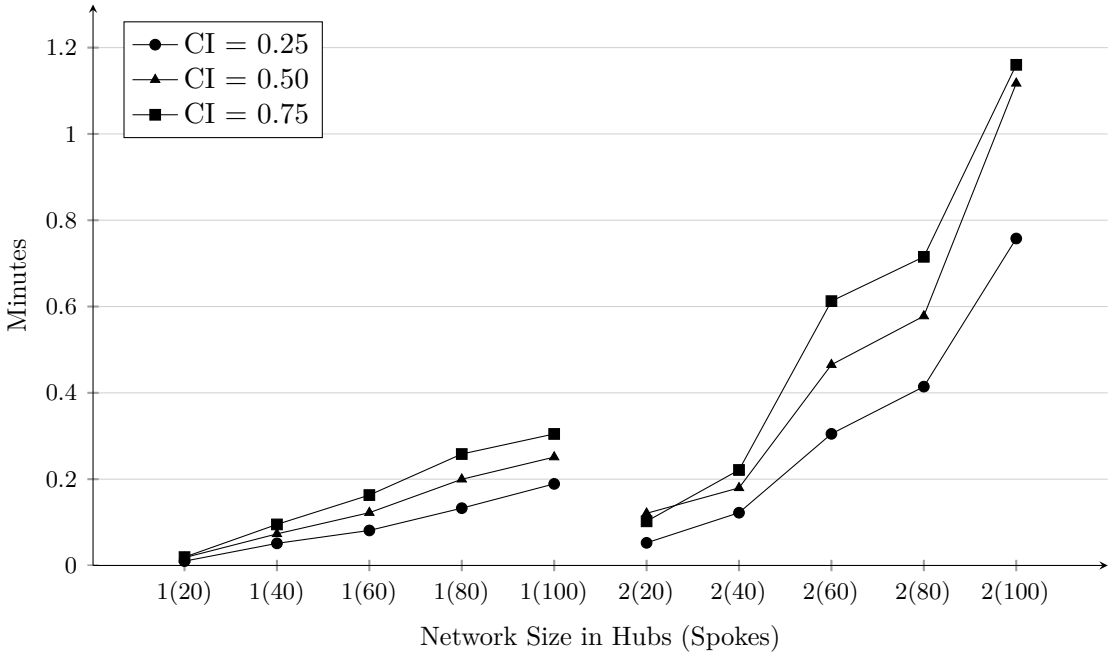


Figure 5.13: Average Computation Times in Minutes with $\mu = 2$

The large differences in computation times are mainly due to the fact that in the exact Algorithm 5.1 forbidding a strategy takes time to augment the model and leads to more iterations for the termination. The heuristic, on the other hand terminates almost as soon as a strategy is chosen for the second time and thus takes much less time. Figures 5.16 – 5.18 displaying

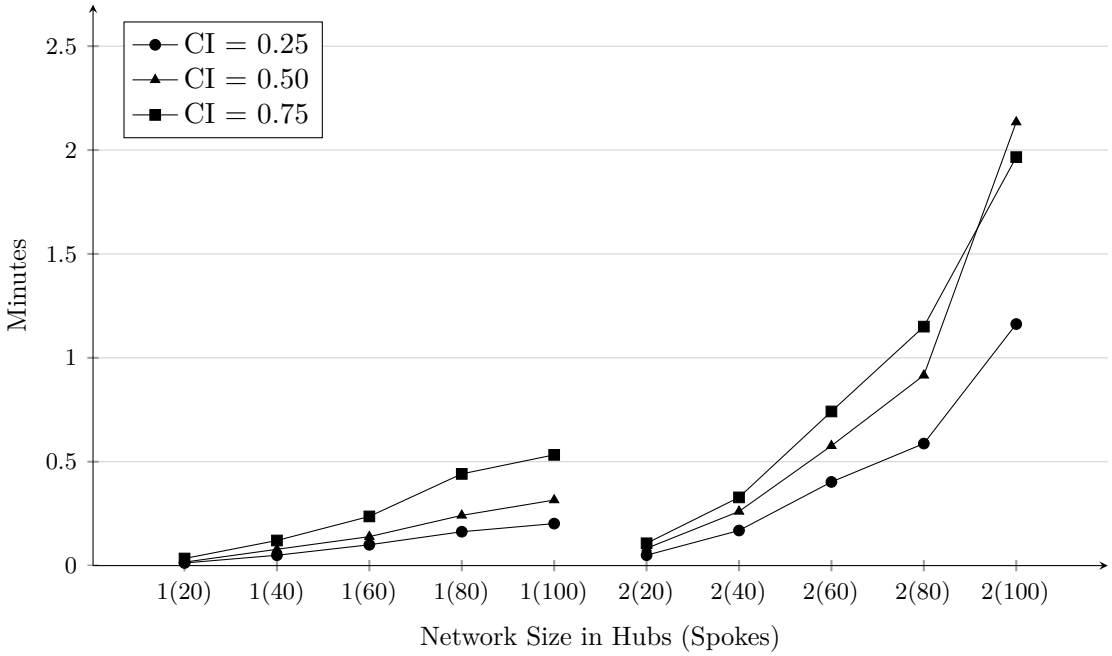


Figure 5.14: Average Computation Times in Minutes with $\mu = 4$

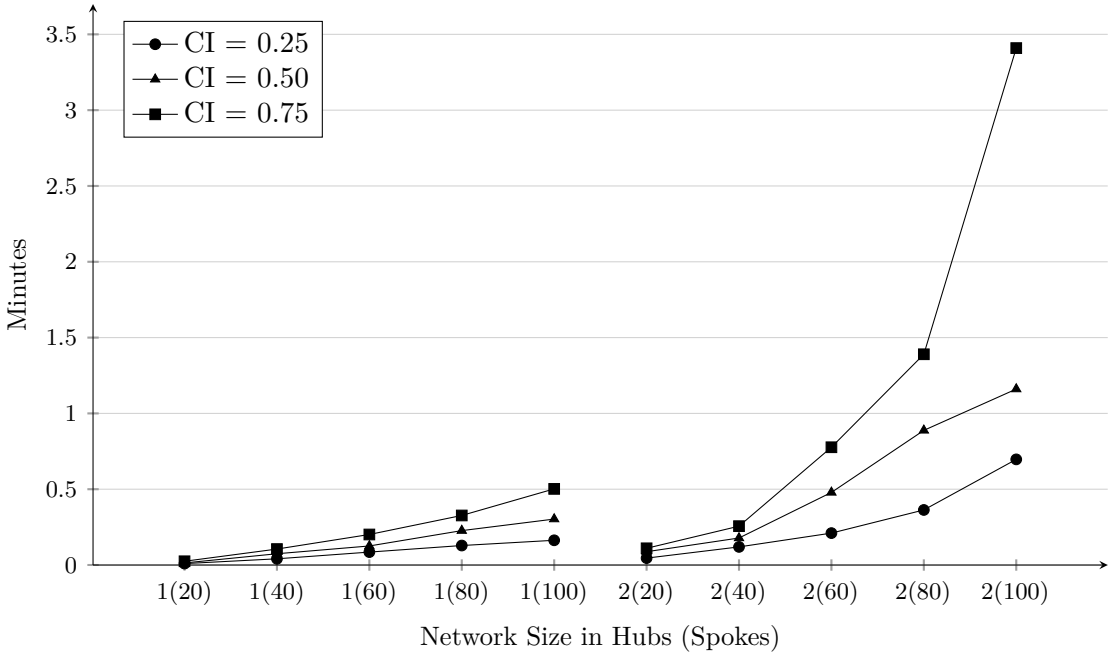


Figure 5.15: Average Computation Times in Minutes with $\mu = 6$

the average numbers of iterations needed by the heuristic to terminate show a similar behavior as the computation times, although the reductions are not quite as large. While with a mean demand of $\mu = 2$ the number of iterations are similar by both methods, the larger values of mean demands did not lead to such large increases as with the exact algorithm. Again, the graphs for the heuristic show a smoother behavior.

Tables 5.21 – 5.23 display the payoff ratios for the heuristic. The equilibrium payoffs in the

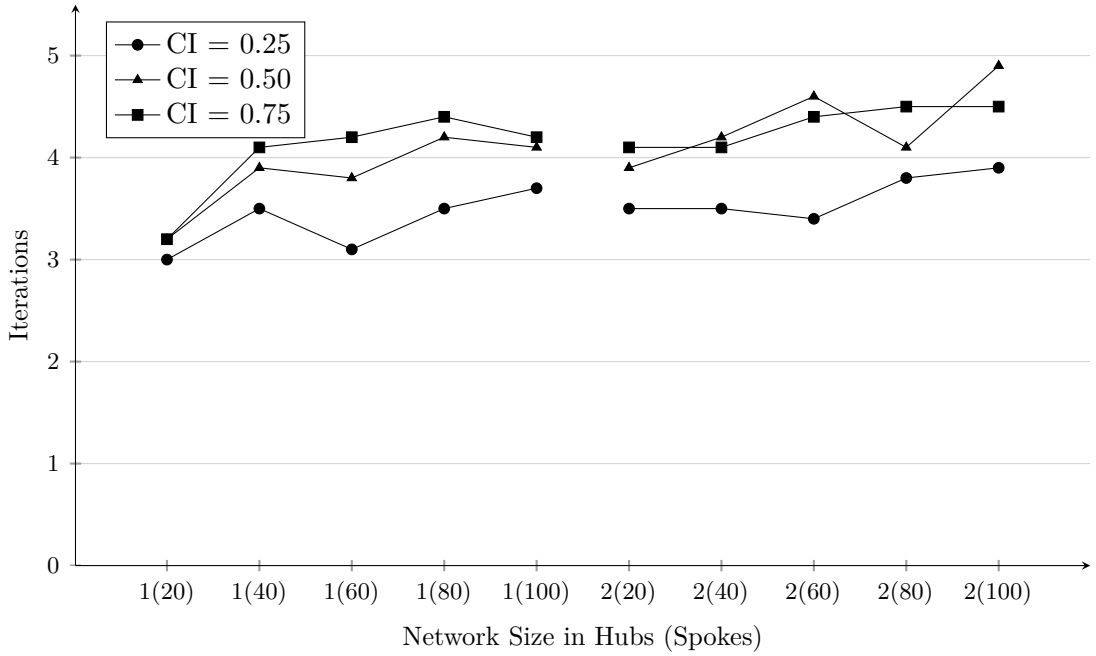


Figure 5.16: Average Numbers of Iterations with $\mu = 2$

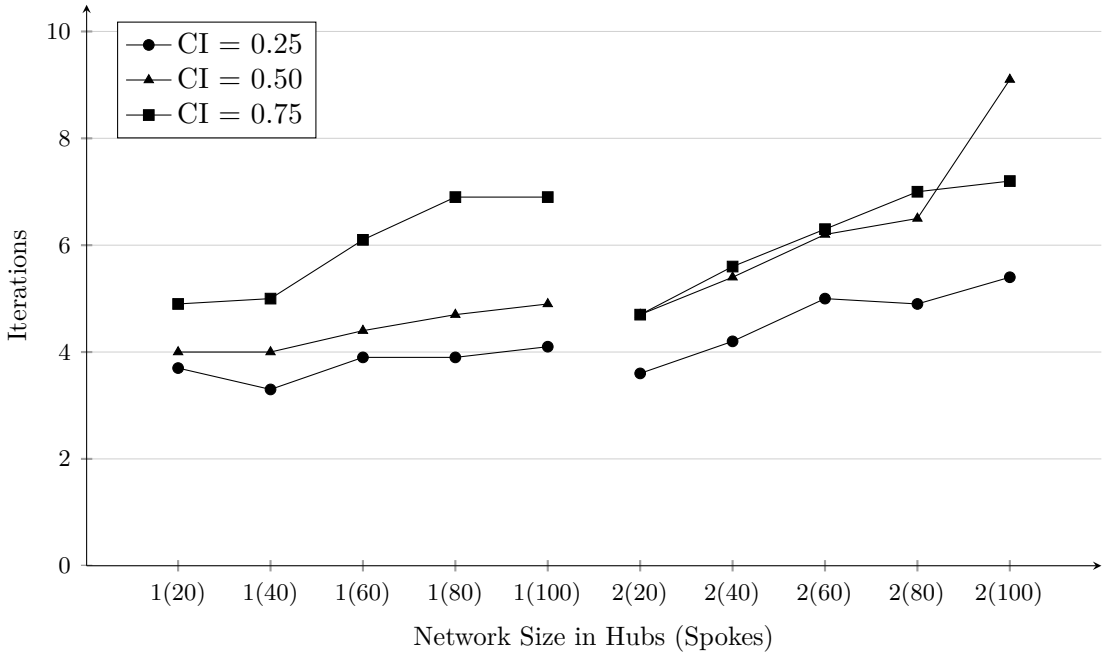


Figure 5.17: Average Numbers of Iterations with $\mu = 4$

heuristic have similar values as those with the exact algorithm. I.e. with an increasing CI the NE payoff ratios increased but decreased with the network size.

The direct comparison of the payoffs in the exact and the approximate NE shows an even clearer picture. The second, fourth, and sixth rows of Tables 5.24 – 5.26 show this direct comparison of summed equilibrium payoffs. Here, the entries were only made for those networks in which an exact pure NE could not be found with the simple iterative search for best responses. For the cases the ratios would turn out to 100% since both, the exact Algorithm 5.1 and the

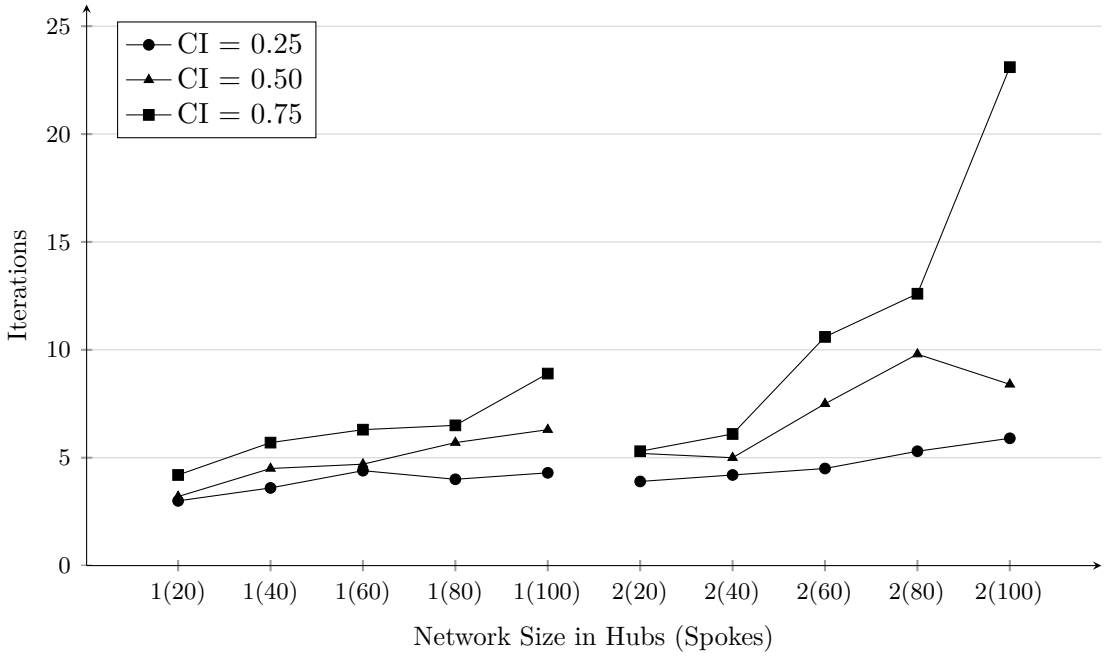


Figure 5.18: Average Numbers of Iterations with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	99.91%	99.97%	99.96%	99.97%	99.98%	100.01%	100.04%	100.02%	99.97%	99.95%
	NC/C	99.87%	99.92%	99.92%	99.93%	99.94%	99.94%	99.99%	99.97%	99.93%	99.91%
	NC/NE	99.97%	99.95%	99.96%	99.96%	99.96%	99.93%	99.95%	99.96%	99.96%	99.96%
0.50	NE/C	99.92%	100.00%	100.03%	100.04%	100.06%	100.20%	100.16%	100.09%	100.04%	100.01%
	NC/C	99.85%	99.91%	99.95%	99.97%	99.98%	100.06%	100.06%	100.00%	99.96%	99.93%
	NC/NE	99.93%	99.91%	99.92%	99.92%	99.92%	99.86%	99.90%	99.92%	99.92%	99.92%
0.75	NE/C	99.92%	100.03%	100.03%	100.07%	100.10%	100.31%	100.27%	100.15%	100.12%	100.06%
	NC/C	99.84%	99.93%	99.93%	99.96%	100.00%	100.14%	100.15%	100.05%	100.03%	99.97%
	NC/NE	99.92%	99.90%	99.90%	99.90%	99.90%	99.83%	99.88%	99.90%	99.91%	99.91%

Table 5.21: Average Payoff Ratios with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.12%	100.13%	100.12%	100.09%	100.12%	100.14%	100.05%	100.01%	99.97%	99.99%
	NC/C	100.02%	100.06%	100.04%	100.02%	100.04%	100.05%	99.98%	99.95%	99.92%	99.93%
	NC/NE	99.90%	99.93%	99.92%	99.92%	99.92%	99.91%	99.93%	99.94%	99.94%	99.95%
0.50	NE/C	100.28%	100.35%	100.32%	100.31%	100.31%	100.42%	100.18%	100.14%	100.09%	100.09%
	NC/C	100.09%	100.20%	100.16%	100.15%	100.15%	100.25%	100.05%	100.03%	99.98%	99.99%
	NC/NE	99.82%	99.85%	99.84%	99.84%	99.84%	99.83%	99.87%	99.89%	99.89%	99.90%
0.75	NE/C	100.32%	100.45%	100.40%	100.44%	100.44%	100.54%	100.28%	100.21%	100.16%	100.15%
	NC/C	100.09%	100.26%	100.19%	100.21%	100.21%	100.32%	100.09%	100.06%	100.00%	100.00%
	NC/NE	99.77%	99.81%	99.79%	99.78%	99.77%	99.78%	99.81%	99.84%	99.85%	99.85%

Table 5.22: Average Payoff Ratios with $\mu = 4$

heuristic from Algorithm 5.2 would generate identical payoffs. The tables show that with the heuristic the players could receive almost the same payoffs as in the exact algorithm. However, while in the networks with only one hub the heuristic lead to slightly higher expected payoffs in the respective mixed approximate NE, the exact algorithm performed better in the larger networks and with a higher mean demand.

The sizes of the reduced payoff matrices were encouragingly small in most cases. In about

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.22%	100.21%	100.23%	100.17%	100.22%	100.11%	100.10%	100.09%	100.08%	100.06%
	NC/C	100.14%	100.08%	100.10%	100.06%	100.09%	100.02%	100.02%	100.01%	99.99%	99.98%
	NC/NE	99.91%	99.87%	99.87%	99.89%	99.88%	99.91%	99.92%	99.92%	99.91%	99.92%
0.50	NE/C	100.45%	100.53%	100.54%	100.48%	100.54%	100.38%	100.31%	100.25%	100.24%	100.23%
	NC/C	100.30%	100.28%	100.28%	100.24%	100.29%	100.21%	100.16%	100.10%	100.07%	100.06%
	NC/NE	99.84%	99.75%	99.74%	99.76%	99.75%	99.83%	99.85%	99.85%	99.84%	99.84%
0.75	NE/C	100.53%	100.83%	100.81%	100.73%	100.82%	100.54%	100.46%	100.36%	100.36%	100.33%
	NC/C	100.34%	100.43%	100.42%	100.35%	100.43%	100.32%	100.25%	100.15%	100.14%	100.11%
	NC/NE	99.81%	99.60%	99.61%	99.63%	99.61%	99.78%	99.79%	99.79%	99.77%	99.78%

Table 5.23: Average Payoff Ratios with $\mu = 6$

70% of all instances with approximate NE (157 out of 225), only two strategies per player were available in the reduced games. Three and four strategies were available in 22 and 20 cases, respectively. The highest number of strategies to make up the BRMs was 28. These have the same structure as the game in the introductory Section 5.4. However, in the games in the computational study, several "crucial" products existed with each iteration affecting the booking limits for two such products, only one of which was identical in both iterations.

Out of all reduced games, only 14 (about 6%) had more than one equilibrium. Merely three reduced games had a pure NE. The number of NE increased with the game size. For the largest games (one with 24 and one with 28 strategies) the enumeration of all mixed NE was not possible within three hours which shows the huge computational burden of these algorithms based on bimatrix games. Both of these games did not have any pure NE, though. For comparing the results from these instances, we used the payoffs from one sample mixed equilibrium which was computed in less than one second.

Comparing the results of the exact Algorithm 5.1 and the heuristic in Algorithm 5.2, one might argue for the latter since it leads to very similar results as the former in a shorter computation time. However, the exact algorithm is still preferable to the heuristic for two reasons. First, in practice, airlines begin accepting bookings for certain flights as long as up to one year in advance. Considering the somewhat longer computation times for the exact algorithm and the increase of revenue that it generates for large networks, one can accept the computation time in favor of the revenue increase if the exact algorithm is applied to allocate booking limits for all offered flights throughout the year. 30 minutes of computation time (the average times of exact algorithm in the largest networks) in a booking period of one year is surely acceptable. Considering this, one might even set a larger limit for the computation time to reduce the number of instances without an NE. Second, the exact algorithm yields a game's pure NE with certainty if one exists while the heuristic—if it cannot find a pure exact NE—almost certainly yields a mixed, inexact NE which is of a worse quality. This aspect is also valuable in practice where certainty about a game's outcome plays a crucial role in the interpretation and implementation of solutions.

5.4.6.2 Bounds for the Players' Equilibrium Payoffs

As the sizes of games increase, the number of NE tends to increase as well. For this reason it would be convenient to be able to evaluate the payoffs in the NE. One evaluation approach is to deduce bounds for the solutions. We hence now present how we deduce bounds for the players' equilibrium payoffs after the computation of NE.

As pointed out in Section 5.2, a player's payoff in the competitive setting cannot be lower than in the non-competitive setting when the model (5.1) – (5.5) is solved optimally. Thus, as Equation (5.22) shows, a lower bound $\mathcal{LB}^a_{NE}$ for a player's payoff in the competitive setting is his optimal non-competitive payoff $r^{a*}(b^a_{pf})$ calculated from solving the single-airline DLP (2.2) – (2.5).

$$\mathcal{LB}^a_{NE} = r^{a*}(b^a_{pf}) \tag{5.22}$$

Given this information and the payoffs $r^*_{central}$ from the central case as well as the summed payoffs from the NE r^*_{SumNE}, we can derive an upper bound $\mathcal{UB}^a_{NE}$ for a player's payoff in Equation (5.23) by subtracting the competitor's lower bound from the higher value of the central payoff and the summed NE payoffs. For, If one player receives only his lower bound in the NE, the rest is available for the competitor.

$$\mathcal{UB}^a_{NE} = max\{r^*_{central}; r^*_{SumNE}\} - \mathcal{LB}^{-a}_{NE} \tag{5.23}$$

With this in mind, we are after all able to provide a range in which the equilibrium payoffs will lie. With Equation (5.24) we can also describe which proportion ϵ^a of the upper bound the lower bound amounts to or, in other words, how much lower the worst-case payoff is compared to the best-case payoff in the competitive setting. From this equation, we also see why the distinction $max\{r^*_{central}; r^*_{SumNE}\}$ is necessary to determine the upper bound. As we have seen in Section 5.3.5.2, the NE payoffs and even the summed non-competitive payoffs might turn out higher than the cooperative payoff for several reasons. In this case, the value for ϵ^a would turn out negative which would make no sense.

$$\epsilon^a = \frac{\mathcal{UB}^a_{NE} - \mathcal{LB}^a_{NE}}{\mathcal{UB}^a_{NE}} \tag{5.24}$$

But since this value depends on the individual game's data, it is variable as opposed to the ϵ values in the respective ϵ-approximate NE mentioned in Section 5.4. This means that this value can also amount to 1 for a player if his demand for all products is 0 and the other player's total demand exceeds his capacities. This would mean a lower bound of 0, an upper bound of > 0, and a value $\epsilon = 1$ for the player without any demand because of a higher cooperative (central) payoff than the sum of non-competitive payoffs.

The values for ϵ^a in our setup are shown in the third, fifth, and seventh rows of Tables 5.24 – 5.26. Like for the comparison of payoffs generated by the exact algorithm and the heuristic, the values for ϵ^a are only entered for those instances where approximate NE were found. The values for ϵ^a in the exact NE would turn out to 0 in most cases since $\mathcal{UB}^a_{NE} = \mathcal{LB}^a_{NE}$ leading to the enumerator to be 0 in equation (5.24). Hence, those cells of Tables 5.24 – 5.26 without values implicate exact pure NE in all 10 instances of the respective setting. The entries in the tables show average values for ϵ. The bounds for the payoffs in the NE were very tight with $\epsilon^a < 0.01$ (1%) in all cases. On average, ϵ^a amounted to 0.0019 over all networks with $\mu = 2$, 0.0028 over all networks with $\mu = 4$, and 0.0045 over all networks with $\mu = 6$ for both players.

Considering the relatively low values of ϵ and the relatively low revenue increases that the algorithms proposes so far generate, one might consider the effort too high connected with implementing these procedures. However, as was mentioned in Section 3.1, the airline business is one with an intense competition but with low profit margins. E.g. the German airline Lufthansa had 14.57 billion € in earnings in 2014 with accumulated profits amounting to 0 € (Lufthansa,

2014, p. 17, 19). With such a balance, even additional 0.03% of earnings (the lowest difference between the non-competitive and the NE payoff in the computational studies above) would lead to additional 4.37 million € in earnings. If we consider a modest increase of 0.05%, the additional earnings increase to about 7.3 million € and climb up to 14.57 million € if the revenue increase amounts to 0.1%. Given that an airline cannot end up worse than in the non-competitive situation when it employs the competitive models proposed here (due to the non-negativity of spill over demand) and the relatively short computation times, the effort necessary to increase their earnings is acceptable.

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Heuristic/Exact	–	–	100.0037%	–	–	–	–	–	–	99.9992%
	ϵ	–	–	0.0027	–	–	–	–	–	–	0.0020
0.50	Heuristic/Exact	–	–	–	100.0018%	100.0016%	–	99.9978%	–	99.9993%	99.9993%
	ϵ	–	–	–	0.0014	0.0015	–	0.0019	–	0.0015	0.0019
0.75	Heuristic/Exact	–	–	99.9945%	–	–	99.9992%	–	99.9985%	99.9979%	–
	ϵ	–	–	0.0014	–	–	0.0032	–	0.0021	0.0016	–

Table 5.24: Average Payoff Ratios between the Heuristic and the Exact Algorithm with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Heuristic/Exact	–	–	–	–	–	–	100.0029%	100.0028%	99.9981%	100.0080%
	ϵ	–	–	–	–	–	–	0.0014	0.0015	0.0020	0.0016
0.50	Heuristic/Exact	–	–	–	–	100.0024%	100.0008%	–	100.0006%	100.0011%	100.0004%
	ϵ	–	–	–	–	0.0032	0.0034	–	0.0022	0.0022	0.0021
0.75	Heuristic/Exact	–	–	100.0013%	100.0028%	99.9976%	–	99.9953%	100.0003%	99.9985%	99.9998%
	ϵ	–	–	0.0040	0.0045	0.0044	–	0.0036	0.0031	0.0029	0.0030

Table 5.25: Average Payoff Ratios between the Heuristic and the Exact Algorithm with $\mu = 4$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Heuristic/Exact	–	–	–	–	–	–	100.0014%	–	99.9989%	100.0005%
	ϵ	–	–	–	–	–	–	0.0015	–	0.0016	0.0016
0.50	Heuristic/Exact	–	99.9999%	99.9992%	100.0046%	99.9986%	99.9980%	–	99.9984%	99.9974%	99.9968%
	ϵ	–	0.0051	0.0053	0.0046	0.0050	0.0032	–	0.0030	0.0032	0.0031
0.75	Heuristic/Exact	99.9939%	99.9937%	99.9931%	99.9948%	99.9971%	99.9905%	99.9971%	99.9968%	99.9991%	99.9947%
	ϵ	0.0034	0.0085	0.0075	0.0074	0.0076	0.0042	0.0042	0.0043	0.0047	0.0045

Table 5.26: Average Payoff Ratios between the Heuristic and the Exact Algorithm with $\mu = 6$

Chapter 6

Simultaneous Price and Quantity Competition

While Chapter 5 dealt only with capacity competition, price competition is also a crucial factor in RM industries. Yet, the simultaneous competition over prices and capacities is seldom considered in RM literature. In this chapter, we want to help closing the gap between independent price and quantity competition and investigate the problem of two airlines that simultaneously compete within a network RM setting via both prices and quantities. There, an airline's demand is assumed to be a function of prices and the competitor's behavior (prices and booking limits). This chapter is based on Grauberger and Kimms (2014a).

Although a product's price and demand are interdependent and affect each other, they still are seldom optimized jointly although this tendentially leads to increased revenues (Jacobs et al., 2000). In the RM literature it is still often assumed that prices are fixed and capacity control is performed under this assumption or the capacity allocation is assumed as fixed and only the price is optimized. Reasons for this are that the models become more complex with both, prices and booking limits being variables and that coordinating the so far independent systems for pricing and capacity allocation requires big efforts which might not be worth it. Recently, Kocabiyikoğlu et al. (2014) have shown under which circumstances coordinating these decisions is worth the effort and under which the effort would exceed its benefit.

Although antitrust immunity might be granted to alliances, this is not relevant here. For, following antitrust immunity, the partners can set prices and control capacities jointly which eliminates the need for competitive pricing and capacity control since a centralized solution can be implemented. Further, antitrust immunity would turn the situation at hand into a cooperative pricing and capacity allocation game. For, in order to implement a centralized solution in practice, the players must decide about how to share the revenue in a fair way, so that each of them is not worse off than in the non-cooperative (and non-competitive) situation. Also, antitrust immunity may be granted only under certain conditions. These conditions would have to be included in the models as well. All in all, tools from cooperative game theory, as presented in e.g. Kimms and Çetiner (2012) and Çetiner and Kimms (2013), would have to be applied in case of antitrust immunity, which is not subject of this thesis. Hence, the game under simultaneous price and quantity competition within alliances is the same as without alliances. Cooperative aspects of airline alliances will be taken into account in the next chapter in form of code sharing agreements.

In the next section, we first mention some publications dealing with setting prices and

booking limits simultaneously without considering competition and then sum up the few publications we found dealing with simultaneous price and quantity competition. Section 6.2 presents the competitive model used in the further approach, while Section 6.3 presents an algorithm to compute an approximate NE under simultaneous price and capacity competition. Section 6.4 presents the computational study.

6.1 Related Literature

Optimizing prices and allocating capacities independently thus can leave "money on the table" and does not exploit all possibilities for maximizing total revenues. Furthermore, simultaneous price and quantity competition is seldom considered which leaves even more room for optimization.

Simultaneous price and quantity competition in a revenue management setting is even less investigated than simultaneous price and quantity optimization without competition. Weatherford (1997) was one of the first authors to recognize the importance of jointly optimizing prices and seat allocations. He proposed models to account for $\mathcal{F}$ price classes on a single flight leg. His models were solved directly using a commercial solver for non-linear models. In Feng and Xiao (2006) it was assumed that there is only a pre-specified set of prices from which a subset was chosen in their optimal solution. They also assumed a dynamic setting in which the available set of prices is time-dependent. Chew et al. (2009) proposed a dynamic program for one product with a two-period lifetime. They assumed that the price for a product increased as the time of its perishment approached. For lifetimes longer than two periods, three heuristics were proposed to approximate the optimal payoff. Further extensions were also introduced. Cizaire and Belobaba (2013) assumed two fare classes but these were set in a two-period time frame and the decisions in the earlier time frame affected the decisions in the later one. Their model combined the approaches by Weatherford (1997) and Chew et al. (2009).

However, in these publications only one product or one single leg was assumed. One of the first to consider the joint pricing and seat allocation optimization in a network was de Boer (2003). Her model was based on a probabilistic, non-linear problem and was solved via an iterative non-linear optimization algorithm. She also provided heuristics for a multi-period problem. See also Bitran and Caldentey (2003) for a literature overview and several proposed models. They also addressed the problem within a dynamic pricing setting.

Côté et al. (2003) were among the first authors to consider competition in a joint pricing/allocation problem. They proposed a bilevel model to optimize these decisions. However, in their setting, there is a leader airline and follower airlines constituting the aggregate competition. Raza and Akgunduz (2008) introduced a two-player model for simultaneous price and seat allocation competition on a single flight leg for two price classes. They also provided a simple iterative algorithm to determine the (approximate) equilibrium prices and booking limits.

Grauberger and Kimms (2014a) have picked up many of these ideas and extended them further. Their approach built the foundation for this chapter. While Côté et al. (2003) described a Stackelberg game and did not provide an algorithm for solving the proposed problem, in Grauberger and Kimms (2014a) a Nash game was introduced together with an algorithm to approximate an NE under simultaneous price and quantity competition. Further, the authors optimized the players' decisions over a network with more than two classes, unlike Raza and Akgunduz (2008).

In Zhao and Atkins (2008) the problem of N newsvendors under simultaneous price and inventory competition was investigated. The authors proved the existence of a pure NE and showed under which circumstances this NE is unique. However, a competitive newsvendor problem is more related to a one-leg, two-class RM problem under competition, as Netessine and Shumsky (2005) showed. We, on the other hand, assume larger airline networks with more than two fare classes, but only two players.

6.2 Model Formulation

As the previous section showed, a very limited amount of research has been done on simultaneous price and seat allocation competition in revenue management. The mentioned publications are also very restricted in use since only one leg or two fare classes are considered or no algorithm is provided for solving the model. To the best of our knowledge, prior to Grauberger and Kimms (2014a) no model has been presented together with an algorithm for searching for NE under simultaneous price and quantity competition on a network with $\mathcal{F}$ classes. We intend to fill this gap. Moreover, we propose an algorithm to solve the game on hand. As before, we assume the competitors to interact in a game of complete information, i.e. both have complete knowledge about all relevant information concerning the competitor and both are aware of this fact. This is a common assumption also for games with simultaneous price and quantity competition made also in e.g. Côté et al. (2003), Raza and Akgunduz (2008) as well as Zhao and Atkins (2008).

We assume demands to be linear functions of the players' prices which are simple and allow for a first examination of the model's interdependencies. While linear demand functions are not necessarily realistic, they are used quite commonly to approximate demand as was also done in the publications mentioned in the previous section. See also Talluri and van Ryzin (2004b, Sec. 7.3) as well as Phillips (2007, Ch. 3). $D^a_{pf} \geq 0$ is the demand with both players' prices being equal to zero and $\beta^a_{pf} \geq 0$ shows how this demand reacts to a price change of the considered airline a. The order of request arrivals is not relevant for our model formulation. The demand player a expects to receive for a product during the booking period is a function of his price π^a_{pf} as well as, if $p \in A$, possibly the competitor's price π^{-a}_{pf} and the spillover demand of customers denied by the competitor.

For the quantity competition, $\alpha^{-a,a}_{pf} \in [0,1]$ is again the proportion of customers requesting a product from the competitor a, if they are denied a ticket by their preferred airline $-a$. We assume this proportion to be a function of the players' prices to incorporate the pricing decisions in the quantity competition as well. However, as will be shown in Section 6.4.1, this parameter is determined differently in this case than in Section 5.3.5.1. For the price competition, $\gamma^a_{pf} \geq 0$ shows how the competitor's price affects a player's demand. It is assumed that $\gamma^a_{pf} \leq \min\{\beta^a_{pf}; \beta^{-a}_{pf}\}$ in order 1) to avoid the competitors raising demands by simultaneously raising prices and 2) to make sure that a player's demand is not raised more than the competitor's demand is reduced through his price.

The decision variables for airline a are its booking limits b^a_{pf} and prices π^a_{pf} for its offered products. All in all, airline a solves model $\mathcal{M}^a_{NL}$ consisting of equations (6.1) – (6.6) in response to the competitor's behavior:

$$\mathcal{M}^a_{NL}\text{: max} \quad r^a(b^a_{pf}, \pi^a_{pf}) = \sum_{p=1}^{\mathcal{P}^a} \sum_{f=1}^{\mathcal{F}} \pi^a_{pf} b^a_{pf} \tag{6.1}$$

$$\text{subject to} \quad b^a_{pf} \leq D^a_{pf} - \beta^a_{pf}\pi^a_{pf} \qquad p \in NA^a, f \in F \quad (6.2)$$

$$b^a_{pf} \leq D^a_{pf} + \alpha^{-a,a}_{pf}((D^{-a}_{pf} - \beta^{-a}_{pf}\pi^{-a}_{pf}) - b^{-a}_{pf})^+ - \beta^a_{pf}\pi^a_{pf} + \gamma^a_{pf}\pi^{-a}_{pf} \qquad p \in A, f \in F \quad (6.3)$$

$$\sum_{p=1}^{\mathcal{P}^a}\sum_{f=1}^{\mathcal{F}} M^a_{lp} b^a_{pf} \leq C^a_l \qquad l \in L^a \quad (6.4)$$

$$b^a_{pf} \geq 0 \qquad p \in P^a, f \in F \quad (6.5)$$

$$\pi^a_{pf} \geq 0 \qquad p \in P^a, f \in F \quad (6.6)$$

The objective function (6.1) maximizes the total revenue summing over the prices for the products times their booking limits. Restrictions (6.2) and (6.3) limit the player's products' booking limits to their expected demands. In the former, products not affected by competition are considered while the latter considers the products affected by competition. Here, the total expected demand is a function of both players' prices and the spillover demand from the competitor. Note that here the spillover demand $((D^{-a}_{pf} - \beta^{-a}_{pf}\pi^{-a}_{pf}) - b^{-a}_{pf})^+$ is also a function of the competitor's price and booking limit for a product affected by competition. Restriction (6.4) is the capacity restriction requiring the sum of all booking limits on a leg not to exceed the airplane's capacity on this leg. Finally, restrictions (6.5) and (6.6) define the booking limits and prices to be non-negative continuous numbers.

In order to model more than two competitors, again the index $-a$ would have to stand for all players in set $\mathcal{A}$ except the current player a and the set of products affected by competition would have to be extended by an index a to A^a to be differentiated by players. Constraint (6.3) would have to be extended to cover the sum of all demand denied by all competitors. Furthermore, the parameter γ^a_{pf} would have to be extended to $\gamma^{-a,a}_{pf}$ to take into account, which competitor's price affects and airline's own demand to which extent. The sum of all parameters $\gamma^{-a,a}_{pf}$ would then have to be at most 1 in order to not (in sum) add more demand to the competitors than is turned away by the considered airline through its own price. Here, also the sum over all competitors would have to be taken.

6.3 Computing an Approximate Nash Equilibrium

Note that the model $\mathcal{M}^a_{NL}$ presented in the previous section is not linear which makes it hard to solve because optimality is harder to achieve in non-linear constrained problems than in linear ones (see e.g. Bazaraa et al., 2006). Although solving such a model is not impossible, the algorithms necessary for it could take a very long time for realistic problems. E.g. the commercial solver MINOS for non-linear problems could not solve a single model of the type (6.1) – (6.6) for the largest networks considered in our computational studies within three hours. In the algorithm we present below, two models (one for each airline) will have to be solved iteratively to find an NE. This would render a non-linear model impractical to use. For this reason, we split the competitive aspects for our algorithm and compute equilibrium prices and booking limits separately thus avoiding non-linearity. If we find an exact quantity-based NE and, based on that, exact competitive prices, our algorithm returns an exact NE under simultaneous price and quantity competition. If however, either an exact quantity-based NE cannot by computed or the prices cannot by set optimally, even after several reoptimizations, our algorithm returns an approximate NE under simultaneous price and quantity competition.

If a player a solves model $\mathcal{M}^a$ optimally, his booking limit for a product affected by competition will amount exactly to this product's total demand, i.e. we will have $((D_{pf}^{-a} - \beta_{pf}^{-a}\pi_{pf}^{-a}) - b_{pf}^{-a})^+ = 0$ for all $p \in A$ and all $f \in F$. For, if an airline would want to set a product's booking limit lower than its demand, it could instead set a higher price for this product which would lower the demand to the level of the desired booking limit and generate a higher revenue with the same number of sold tickets. Hence, although each airline optimizes prices and booking limits jointly, competition only takes place on the price level. For this reason, we can reduce the competitive aspect to a price competition and compute NE prices in a first step. Here, $((D_{pf}^{-a} - \beta_{pf}^{-a}\pi_{pf}^{-a}) - b_{pf}^{-a})^+ = 0$ is the central equilibrium condition around which we build the following algorithm.

Step 1: Computing NE Prices

In the first step of the algorithm we focus on the price competition and ignore capacity restrictions. For this, we assume the product demand functions to be independent of one another. Although this is a simplifying assumption, it is quite common in RM theory and practice (see e.g. de Boer et al., 2002, p. 91, Talluri and van Ryzin, 2004a, p. 16 as well as Talluri and van Ryzin, 2004b, Sec. 7.1).

Since we assume products' demands to be independent of each other, we can set them separately. Player a's revenue function for a product not affected by competition, i.e. for all $p \in NA^a$ and all $f \in F$ reduces to $R_{pf}^a(\pi_{pf}^a) = \pi_{pf}^a(D_{pf}^a - \beta_{pf}^a\pi_{pf}^a)$. For these products, the players are assumed to set the prices according to the monopolistic optimal prices as in Equation (6.7). These are formed from deriving the revenue function, setting it to 0 and isolating π_{pf}^a.

$$\pi_{pf}^{a*} = \frac{D_{pf}^a}{2\beta_{pf}^a} \qquad p \in NA^a, f \in F \tag{6.7}$$

With the first derivative being set to 0 and the second derivative of the revenue function for products being ${R_{pf}^a}''(\pi_{pf}^a) = -2\beta < 0$ is a sufficient condition for Equation (6.7) to yield a product's optimal price for maximizing the revenue function for products in the set NA^a if capacity restrictions are ignored. To determine the optimal prices for products affected by competition, we need the players's reaction functions. Recall that a player's reaction function is the first derivative of his revenue function and shows his best response as a function of the competitor's strategy. If only price competition is considered, player a's revenue function for a product reduces to $R_{pf}^a(\pi_{pf}^a) = \pi_{pf}^a(D_{pf}^a - \beta_{pf}^a\pi_{pf}^a + \gamma_{pf}^a\pi_{pf}^{-a})$. Taking the first derivative of this revenue function, setting it to 0, and isolating π_{pf}^a yields a player's product's reaction function which depends on the opponent's price π_{pf}^{-a}. This function is given in Equation (6.8).

$$\pi_{pf}^a = \frac{D_{pf}^a + \gamma_{pf}^a\pi_{pf}^{-a}}{2\beta_{pf}^a} \qquad p \in A, f \in F \tag{6.8}$$

Here, the second derivative is also ${R_{pf}^a}''(\pi_{pf}^a) = -2\beta < 0$. Together with the first derivative of the revenue function of products in the set A being equal to 0 this is a sufficient condition for Equation (6.8) to determine the price for maximizing the revenue of a product affected by competition. Note that the only difference between Equations (6.7) and (6.8) lies in the enumerator. Since $\gamma_{pf}^a\pi_{pf}^{-a} \geq 0$, the optimal price from the latter cannot be lower than the optimal price from the former. Thus, considering price competition for a product will tendentially

allow for a higher price than ignoring it. Recall that a Nash equilibrium can be expressed as an intersection of the players' reaction functions. To determine this intersection and thus to find a player's NE price for a product affected by competition, the players' reaction functions must equal each another. This can be achieved by plugging in one player's price from Equation (6.8) in the other player's reaction function. After simplifying the resulting expression, one can determine the product's equilibrium price ignoring capacity constraints with Equation (6.9).

$$\pi_{pf}^{a*} = \frac{2D_{pf}^{a}\beta_{pf}^{-a} + \gamma_{pf}^{a}D_{pf}^{-a}}{4\beta_{pf}^{a}\beta_{pf}^{-a} - \gamma_{pf}^{a}\gamma_{pf}^{-a}} \qquad p \in A, f \in F \tag{6.9}$$

The prices determined by equations (6.9) and (6.7) are taken to determine the players' total expected demands $D_{pf}^{a} - \beta_{pf}^{a}\pi_{pf}^{a} + \gamma_{pf}^{a}\pi_{pf}^{-a}$ and $D_{pf}^{a} - \beta_{pf}^{a}\pi_{pf}^{a}$ for products affected and not affected by competition, respectively. If the summed demands achievable with these prices do not exceed any capacity, the products' equilibrium booking limits are simply taken as the rounded down values of the demands achievable with these prices. In this case, Step 2 of the algorithm is skipped and Step 3 is performed to update the prices.

If, however a capacity is exceeded through the summed demands of the products using this capacity, some of the booking limits must be set lower than the demand rates to satisfy the capacity constraints. This is done in the next step.

Step 2: Computing NE Booking Limits

The prices computed with Equation (6.9) do not consider any capacity restrictions and may lead to the summed demands on a leg to exceed its capacity which necessitates optimization of booking limits. For this reason, in a second step we fix the prices in the used model to the equilibrium prices and reduce it to a pure quantity-based DLP in which capacity restrictions are taken into account. This enables us to use Algorithm 5.1 from Section 5.3 to compute a pure NE as mutual best responses computed from this quantity-based DLP. In order to receive integer equilibrium booking limits, we again use the demands' rounded down values in Restrictions (6.2) and (6.3) to make sure that the values on the right-hand sides are integers. This again allows us to keep the booking limit variables continuous and still receive integer booking limits in the optimal solution as in the previous chapter.

The algorithm might terminate without an NE, though, either because none exists or because of restricted computation time. In case the algorithm terminates without an NE, the capacities are allocated non-competitively. I.e. the term for the spillover demand $((D_{pf}^{-a} - \beta_{pf}^{-a}\pi_{pf}^{-a}) - b_{pf}^{-a})^{+}$, which is not equal to zero yet after Step 1, is removed from restriction (6.3) in the players' models and they are solved anew. We motivate in the next step the decision of using non-competitive booking limits instead of the booking limits from the mixed approximate NE generated by heuristic in Algorithm 5.2.

Step 3: Updating the Prices

Total demand and booking limits may not be identical after Step 1 and/or Step 2 which would violate the equilibrium condition $((D_{pf}^{a} - \beta_{pf}^{a}\pi_{pf}^{a}) - b_{pf}^{a})^{+} = 0$. E.g. the booking limits from the previous step might turn out lower or higher than the demand based on the equilibrium prices from Step 1, depending on the capacities as well as the competitor's behavior and demand

spill for other products. In order to match demand and booking limits and ensure the equilibrium condition, we update the players' prices with the LP (6.10) – (6.12) where the parameter b_{pf}^{a*} is player a's equilibrium booking limit from Step 2. Both players' prices are updated simultaneously to capture their interdependent effects on the players' demands.

$$\max \quad \sum_{a\in\{1;-1\}} \sum_{p=1}^{\mathcal{P}^a} \sum_{f=1}^{\mathcal{F}} (D_{pf}^a - \beta_{pf}^a \pi_{pf}^a + \gamma_{pf}^a \pi_{pf}^{-a}) \tag{6.10}$$

$$\text{subject to} \quad D_{pf}^a - \beta_{pf}^a \pi_{pf}^a + \gamma_{pf}^a \pi_{pf}^{-a} \leq b_{pf}^{a*} \qquad a \in \{1;-1\}, p \in P^a, f \in F \tag{6.11}$$

$$\pi_{pf}^a \geq 0 \qquad a \in \{1;-1\}, p \in P^a, f \in F \tag{6.12}$$

With the booking limits given, we need to make sure that the demand values do not fall too far below the booking limits in order to be able to keep the latter at their current levels and at the same time they are not to be exceeded by the demand values to ensure the equilibrium condition. We take care of the first point by maximizing the players' summed demands in the objective function (6.10). Constraint (6.11) limits a player's total demand to his equilibrium booking limit from Step 2. Since $\gamma_{pf}^a \pi_{pf}^{-a} \geq 0$, this takes care of the second point and ensures the equilibrium condition. In Constraint (6.12) the players' prices are assumed to be non-negative continuous variables. In this model, we need not distinguish after affected and non-affected products but can simply set $\gamma_{pf}^a = 0$ for the non-affected products which eliminates the price competition aspect for these products. This has the same effect as differentiating after A and NA^a.

Note that instead of using the above LP, we could determine the prices of the non-affected products with simple equations and use a pair of linear equations for each O&D pair in the set A. The linear equations would be set up like Restriction (6.11) where equality holds between the left and right hand sides with certainty. However, depending on the parameters, a system of linear equations may not have a solution. With the above LP we need not worry about this issue and receive the best possible solution. Where a system of linear equations would return no solution, in the LP's optimal solution the respective product's demand would turn out lower than the booking limit from Step 2.

Here it becomes clear why we propose to use non-competitive booking limits in step 2 in case no pure NE can be found instead of using the solution from the heuristic in Algorithm 5.2. Applying this heuristic might yield mixed NE here which would complicate the further procedure of the algorithm presented in this section because the model in this step depends on pure strategies as inputs. Allowing mixed strategies here would mean to set up decision trees with the number of branches corresponding to the number of pure strategies entailed in the mixed strategy and the corresponding probabilities assigned to the branches. Afterwards, new optimal prices would have to be determined for each of the pure strategies with the result that each product might end up with several prices; each with a certain probability with which the corresponding pure strategy used as input for the price optimization is played. Such an approach would complicate the procedure and leave uncertainty about which strategy (as a combination of a product's price and it booking limit) to play in the end. This uncertainty would even be amplified if the steps 2 and 3 must be repeated which is explained in the next step. This is why we use the non-competitive booking limits as input for the new prices in this step.

If in the optimal solution of the LP (6.10) – (6.12) the prices are set so that for all of both players' products the demands amount exactly to the booking limits from Step 2, we are done and Step 4 can be skipped. In this case, equality would hold for all constraints (6.11). The

booking limits from Step 2 and the prices from this step would constitute the equilibrium values. However, if the demand for a product turns out lower than the booking limit, Step 4 needs to be performed.

Step 4: Repeating Steps 2 and 3

The updated prices might lead to lower demand rates than the ones used in Step 2 for seeking the capacity-based NE. This, on the other hand, makes necessary a revision of the booking limits necessitating again price updates in Step 3. Hence, if a product's demand turns out lower than the booking limits from Step 2, Steps 2 and 3 are repeated until the difference in the summed revenues (equilibrium booking limits from Step 2 times updated prices from Step 3) in two consecutive iterations does not amount to more than a pre-defined value δ.

Step 4 makes sure that the players' revenues converge to a steady state because updating the prices in Step 3 leads to ever lower effective demands which in the equilibrium do not exceed the capacities and thus in equilibrium the booking limits will exactly amount to the demands. Since we cannot predict how long this process may take in the worst case, we let the algorithm terminate when the equilibrium summed revenues are reached approximately. In this case our procedure finds an approximate NE. An approximate NE is also found if the algorithm employed in Step 2 terminates without a capacity-based NE which would make necessary to take the players' non-competitive booking limits as input to Step 3 leading to different prices in this step than in an NE. Thus, we cannot guarantee that the eventual result of our approach will be an exact NE. However, the approach terminates with an exact NE if Step 2 yields an exact capacity-based NE and afterwards equality holds in all Constraints (6.11) of Step 3. Then, both players' models (6.1) – (6.6) are solved optimally and no further optimization is necessary by repeating Step 2.

Note that although the demand functions were assumed to be linear here, the approach would also be applicable with non-linear functions. The advantage of linear demand functions is the twofold differentiability of the revenue functions based on them. This allows for checking the sufficient optimality conditions and for the direct determination of the optimal product prices in Step 1 by plugging in the corresponding parameters in Equations (6.7) and (6.9). The same is true for non-linear demand functions which lead to twice differentiable revenue functions. However, while the prices in Step 1 could be determined relatively easily with non-linear, twice differentiable revenue functions, the model (6.10) – (6.12) for updating the prices in Step 3 would become non-linear as well and thus possibly more difficult to solve optimally as was mentioned above. In Step 1, it is possible to determine the prices which maximize non-linear revenue functions which cannot be differentiated twice. However, this would necessitate own algorithms to determine each product price individually, e.g. the golden section method (for the products in NA^a) or the cyclic coordinate method (for the products in set A), see e.g. Bazaraa et al. (2006, Ch. 8). Further, these approaches would only narrow down the intervals in which the optimal prices would lie and would retain some form of uncertainty. The model (6.10) – (6.12) would likewise be more difficult to optimize. All in all, this would complicate the simultaneous optimization of product prices and booking limits and might render it less inapplicable in practice. In order to extend the heuristic to cover more than two players, one would merely have to perform Steps 1 – 4 for the appropriate number of players.

6.4 Computational Study

6.4.1 Test Bed

The networks were assumed to have the same structure as in the previous Chapter; one or two hubs with numbers of 20, 40, 60, 80, and 100 airports to be connected with each hub, respectively. The assumptions about capacities were also the same; the capacities on half of the flights going in and out of the hubs were set to 100 seats and 200 each and the capacities on the legs connecting the hubs to 300.

Out of all possible products, we again chose randomly so many that each leg was demanded by at most 60 products. We also assumed the competition intensity CI to amount to 25%, 50%, and 75% of an airline's total offered products. Concerning the original demand D^a_{pf}, we generated randomly ten pairs of values for each player from the Poisson distribution with mean $\mu = 10$ for each network setting. For the quantity competition, the proportion of denied customers making a request at the competitor was set to $\alpha^{-a,a}_{pf} = 0.5 - \phi^a_{pf}(\pi^a_{pf}; \pi^{-a}_{pf})$ with

$$\phi^a_{pf}(\pi^a_{pf}; \pi^{-a}_{pf}) = \begin{cases} \dfrac{0.5\beta^a_{pf}}{D^a_{pf}}(\pi^a_{pf} - \pi^{-a}_{pf}) & \text{if } \pi^a_{pf} > \pi^{-a}_{pf} \\ \dfrac{0.5\beta^{-a}_{pf}}{D^{-a}_{pf}}(\pi^a_{pf} - \pi^{-a}_{pf}) & \text{otherwise.} \end{cases}$$

If the price difference for a product is 0, half of an airline's denied passengers request a ticket at the competitor in the quantity-based situation because then $\alpha^{-a,a}_{pf} = 0.5$. If airline a sets a lower price than the competitor, it receives more requests and vice versa. The lowest value $\phi^a_{pf}(\pi^a_{pf}; \pi^{-a}_{pf})$ could amount to was -0.5 while its highest value was 0.5. This makes sure that all together, the proportion of spillover demand ($\alpha^{-a,a}_{pf}$) can at most amount to 1 if airline a sets a price of 0 and the competitor sets his price to his maximum for a product. On the other hand, it can drop to 0 in the opposite case. The values for β^a_{pf} and γ^a_{pf} depended on the number of legs used by the product and on the fare class. The values were set to $\beta^a_{pf} = 0.02 * f$ and $\gamma^a_{pf} = 0.01 * f$, $\beta^a_{pf} = 0.015 * f$ and $\gamma^a_{pf} = 0.0075 * f$, and $\beta^a_{pf} = 0.01 * f$ and $\gamma^a_{pf} = 0.005 * f$ for single-leg, two-leg and three-leg products, respectively. This way we could manipulate the prices and make sure that the more legs a product used the higher its price was. With these parameter values, the demand-to-capacity ratios corresponded approximately to those with $\mu = 4$ in the previous chapter.

The perturbing parameter τ^a had again to be set so that the (unique) optimal solution of the perturbed problem is also an optimal solution in the original one. Since the price difference now amounted to 0.01 in the smallest case, we had to make sure that the perturbing values added to the smaller sum's summands did not add up to ≥ 0.01. Assuming 300 as the highest capacity on a plane and all products only occupying one seat, the highest number of summands in our case was 300. Hence, we had to make sure that (in the worst case) the first 300 values of our perturbing vector did not sum up to ≥ 0.01. In order to avoid problems of storing the exponentiated perturbing value in sufficient precision, we multiplied all prices by 30,000 (which led to the smallest positive difference between two prices to amount to 300) and set $\tau^a = 0.9999$ for both players. After the optimization we again subtracted the perturbing parameters from the prices and divided the prices by 30,000 to receive the original prices and payoffs. The perturbation orders for the products were again "alphabetic by O&D name, then increasing by fare class" and "increasing by fare class, then alphabetic by O&D name".

All in all, we constructed 1,200 instances—10 types of networks, three values for the competition intensity, two ways of perturbation order, and 10 pairs of demand for each of these cases, all with competitive prices and monopolistic prices. The competitive prices were set as explained in Section 6.3, while for the monopolistic prices we ignored the price competition completely and set all prices in Step 1 according to Equation (6.7) and set all $\gamma_{pf}^{a} = 0$ for the updated prices in the linear program (6.10) – (6.12). We limited the computation time for each instance to three hours and set δ, the difference in summed revenues from two consecutive iterations of the algorithm to 5%.

6.4.2 Results

In the remainder of this section we again show only the results for the instances with the prices perturbed in order "increasing by fare class, then alphabetic by O&D name". The results for perturbation order "alphabetic by O&D name, then increasing by fare class" can be found in Appendix A.3. The results for both perturbation orders are similar, though. Figures 6.1 and 6.2 show the average computation times in minutes needed for finding a pure NE in Step 2 of our algorithm in the different networks with different competition intensities (CI) as ratios for the products affected by competition. The time for updating the prices was not noticeable. An NE was found in 1,194 of our 1,200 instances (99.5%) within the time limit of three hours in Step 2 of our algorithm and we did not have to reoptimize the booking limits in Step 4 after optimizing the prices in Step 3.

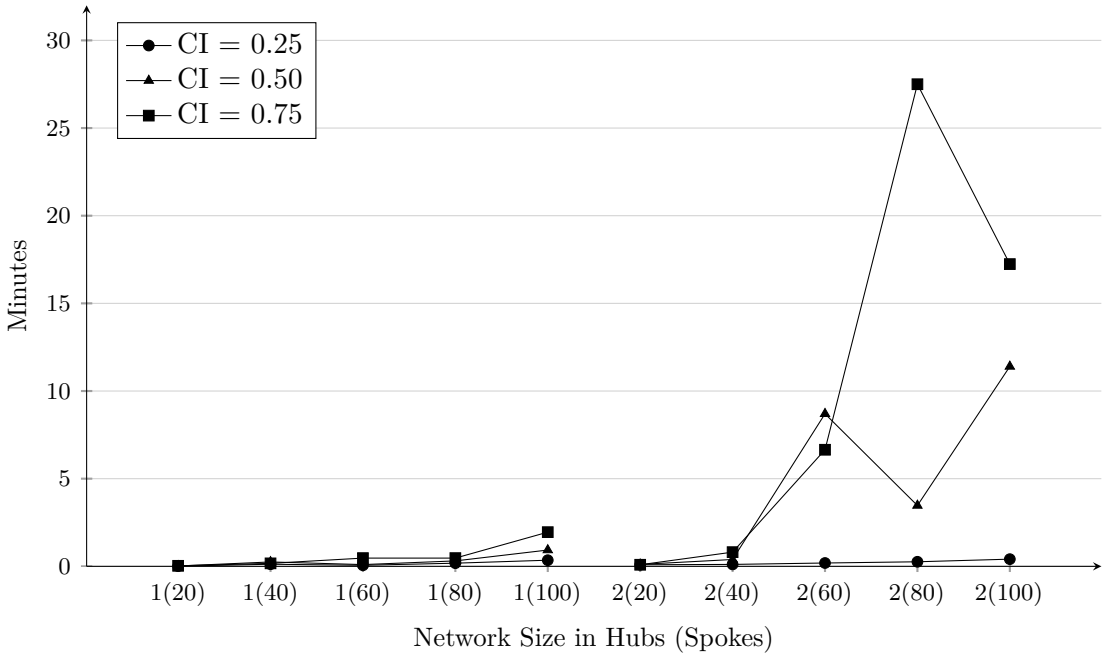


Figure 6.1: Average Computation Times in Minutes with Competitive Prices

From the figures it becomes clear that the more products were affected by competition and the bigger the considered network, the more time was needed to find an NE. While in the smallest networks (less than 60 spoke airports) an NE was usually found within seconds, it took up to about 17.85 minutes on average with 2 hubs and 100 spokes and a competition intensity of CI = 0.75. This increase of time is due to 1) a higher computational effort for solving the—ever larger—models and to 2) a higher number of iterations needed to terminate. Computing NE

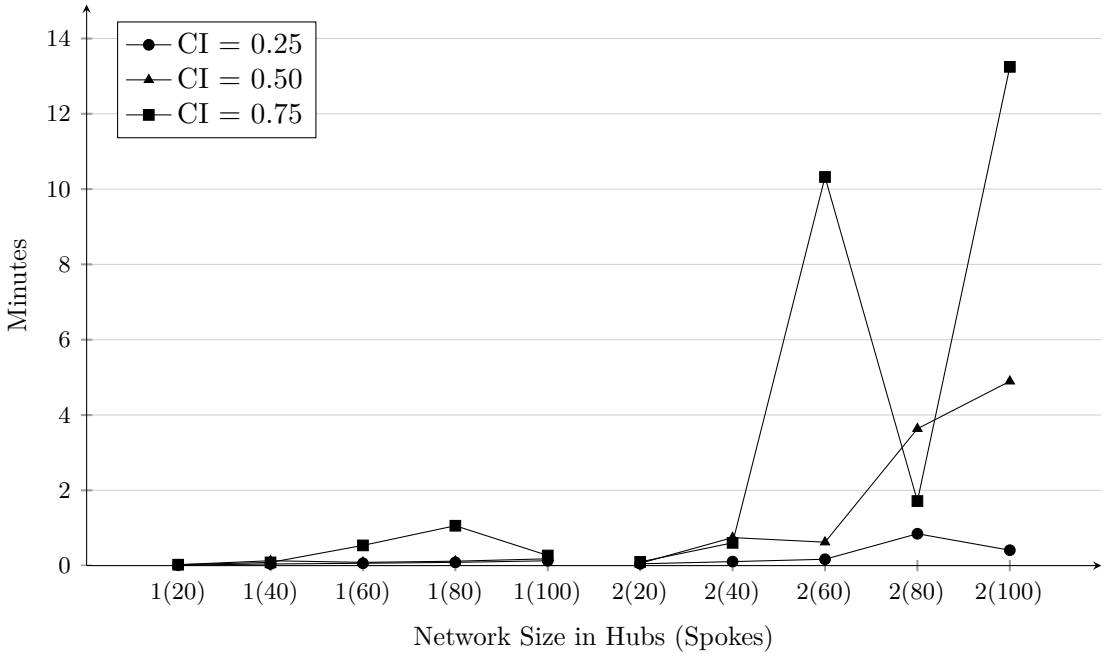


Figure 6.2: Average Computation Times in Minutes with Monopolistic Prices

with monopolistic or competitive prices did not make a big difference on the computation time. Only the times for CI = 0.75 with two hubs were higher in average in the competitive case, while the monopolistic prices led to a higher computation time for one hub and 100 spoke airports. Still, the times were fairly short amounting to less than 20 minutes on average.

The numbers of iterations depicted in Figures 6.3 and 6.4 draw a similar picture like the computation times. Not only are the graphs' paths similar but also the average numbers increase with demand and network sizes. With monopolistic prices less iterations were needed to compute an NE with Algorithm 5.1 in Step 2 of the approach for computing approximate NE under simultaneous price and quantity competition.

We compared the players' revenues after Step 2 to those after Step 3 of our algorithm to compare the results from a mere quantitative competition in Step 2 to those with optimized prices. With our parameters, we noticed a significant revenue increase of up to 11.89% after reoptimizing the prices in Step 3 for player 2 when price competition was taken into account. Player 1 received a higher revenue of up to 11.35% in Step 3. Ignoring price competition led to an increase of up to 3.97% for player 1 and 3.95% for player 2 in the second-largest networks. The least increases were 0.63% for player 1 and 0.75% for player 2 in the monopolistic case. Figures 6.5 – 6.8 show the average percentage increases in revenues with reoptimized prices in Step 3 of the algorithm compared to the revenues after finding a quantitative NE in Step 2. The values were computed with the formula $\frac{\text{Revenue after Step 3}}{\text{Revenue after Step 2}} * 100 - 100$.

One can say that the payoff increases were about proportional to the competition intensity. What stands out is that while with competitive prices the payoffs increased with increasing competition intensity, the payoffs increments shrank with increasing competition intensity with monopolistic prices. In the six instances without an NE in Step 2 the payoff differences (between the monopolistic booking limit allocation and the optimized price after Step 3) were similar to those in the other instances, so we do not show those explicitly here.

These results can also be seen in Tables 6.1 and 6.2 which show the average ratios between

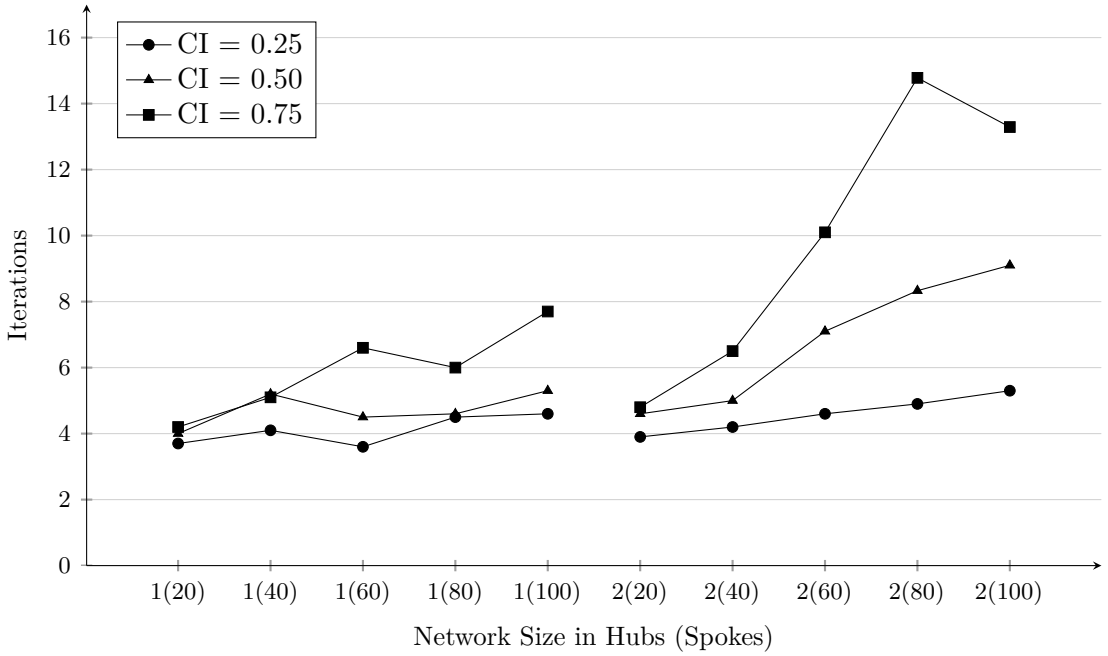


Figure 6.3: Average Numbers of Iterations with Competitive Prices

the players' summed non-competitive payoffs (NC), as well as their summed payoffs after Step 2 and Step 3 of the algorithm, respectively. The entries in the first row of columns 3 – 12 stand for the networks as "Hubs(Spokes)". The Tables' entries are based on the players setting prices competitively (Table 6.1) and monopolistically (Table 6.2), respectively. The smaller percentages in the former table reflect the fact that with competitive prices a price update in Step 3 allows for a larger payoff increase than with monopolistic prices. Also, in the equilibrium of Step 2 the revenues were higher in the competitive case than in the monopolistic case. The entries in rows 3 and 4, 6 and 7 as well as 9 and 10 of Table 6.1 allow for a comparison of the results form exact NE to the results from the approximate NE in which the players set their booking limits non-competitively and with which our procedure might terminate.

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Step 2/Step 3	93.49%	93.07%	93.06%	93.13%	93.15%	93.16%	93.18%	93.21%	93.22%	93.27%
	NC/Step 3	93.19%	92.74%	92.71%	92.81%	92.82%	92.83%	92.89%	92.90%	92.91%	92.95%
	NC/Step 2	99.68%	99.65%	99.62%	99.65%	99.65%	99.65%	99.69%	99.66%	99.66%	99.66%
0.50	Step 2/Step 3	91.68%	91.13%	91.17%	91.16%	91.21%	91.26%	91.35%	91.41%	91.50%	91.62%
	NC/Step 3	91.35%	90.83%	90.85%	90.85%	90.92%	90.97%	91.06%	91.12%	91.20%	91.35%
	NC/Step 2	99.64%	99.67%	99.65%	99.66%	99.68%	99.68%	99.68%	99.68%	99.68%	99.70%
0.75	Step 2/Step 3	90.28%	89.68%	89.60%	89.55%	89.53%	89.65%	89.77%	89.89%	89.98%	90.16%
	NC/Step 3	89.88%	89.29%	89.23%	89.17%	89.15%	89.24%	89.36%	89.48%	89.56%	89.77%
	NC/Step 2	99.56%	99.57%	99.59%	99.57%	99.58%	99.54%	99.54%	99.54%	99.54%	99.57%

Table 6.1: Average Payoff Ratios with Competitive Prices

Comparing the players' payoffs with competitive prices to those with monopolistic prices reveals even higher gains for our parameters as Figures 6.9 – 6.12 show. Again, a higher competition intensity leads to higher gains. The increases stem from tendentially higher prices in the price competition based on Equation (6.9) versus Equation (6.7) and tendentially higher demands in the quantitative competition due to Constraint (6.3). Similar results were found by Zhao and Atkins (2008) for newsvendors under simultaneous price and quantity competition.

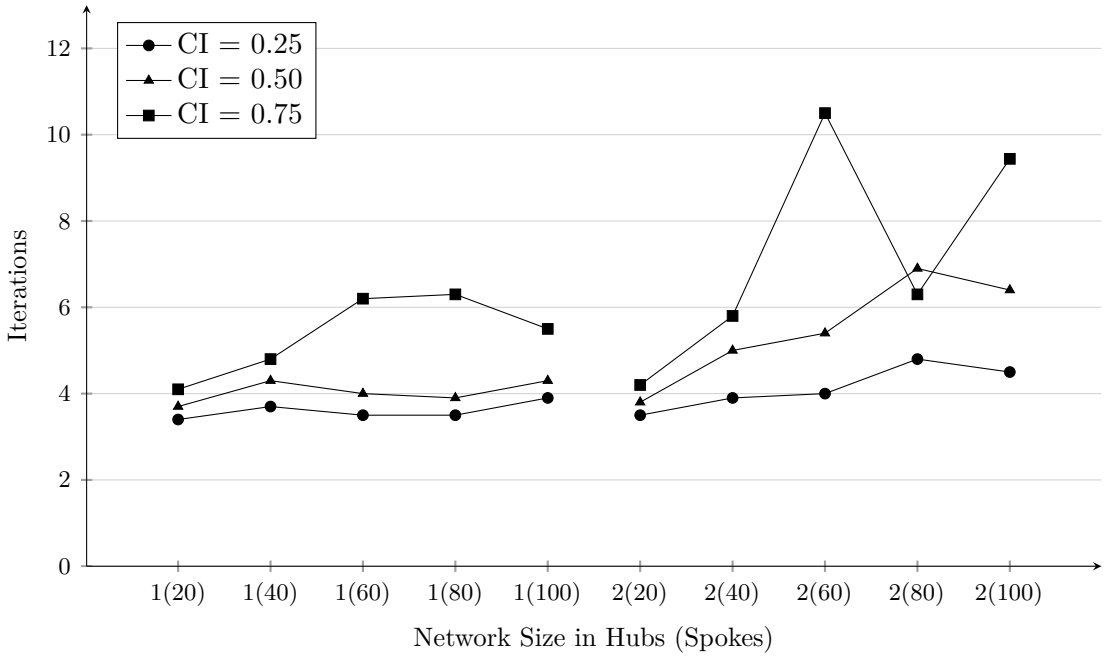


Figure 6.4: Average Numbers of Iterations with Monopolistic Prices in Step 2

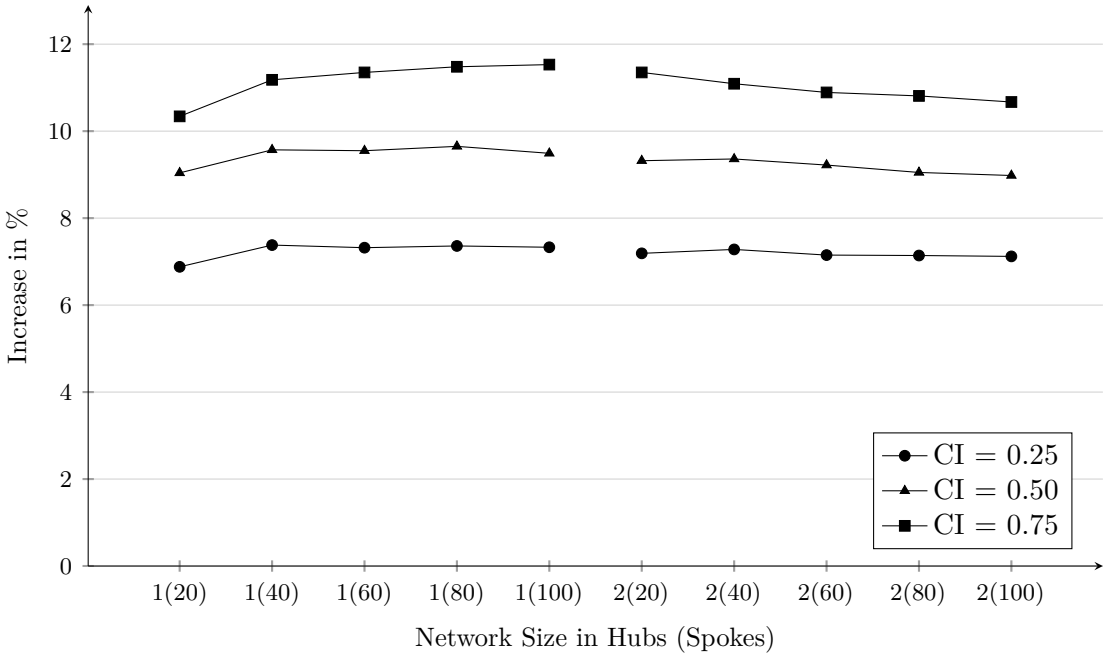


Figure 6.5: Average Payoff Increases From Step 2 to Step 3 with Competitive Prices for Player 1

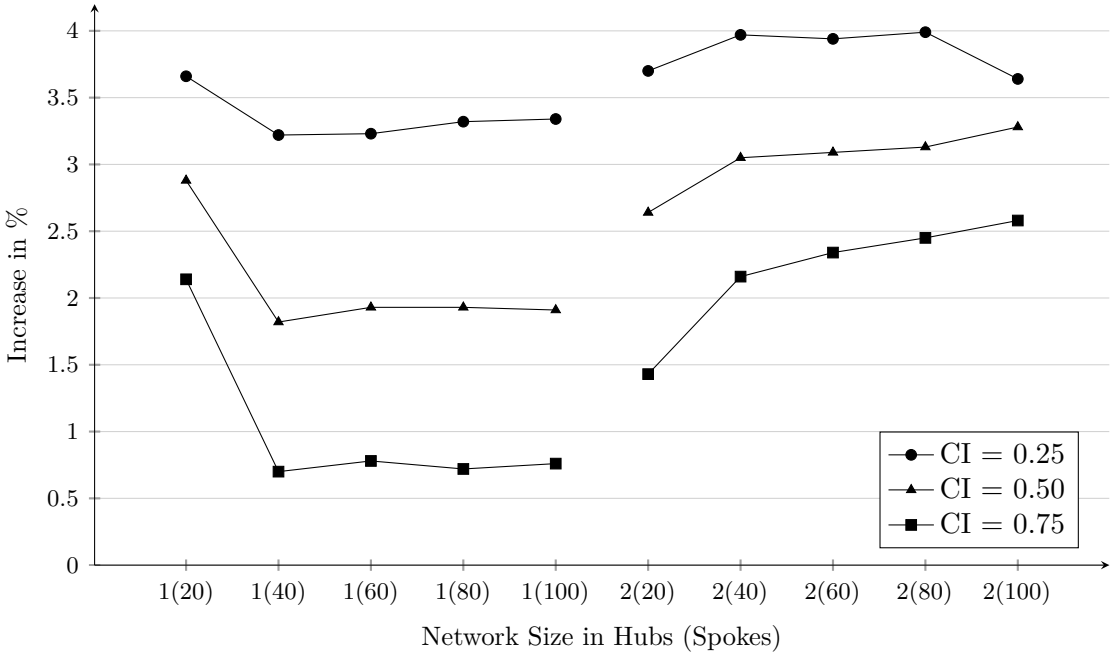


Figure 6.6: Average Payoff Increases From Step 2 to Step 3 with Monopolistic Prices for Player 1

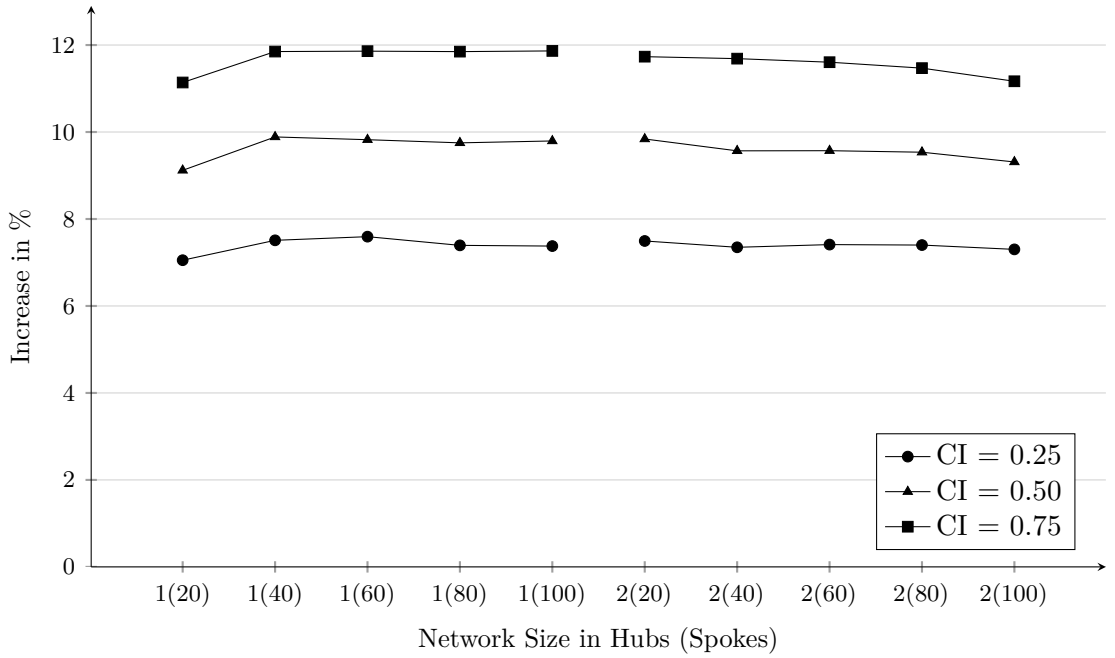


Figure 6.7: Average Payoff Increases From Step 2 to Step 3 with Competitive Prices for Player 2

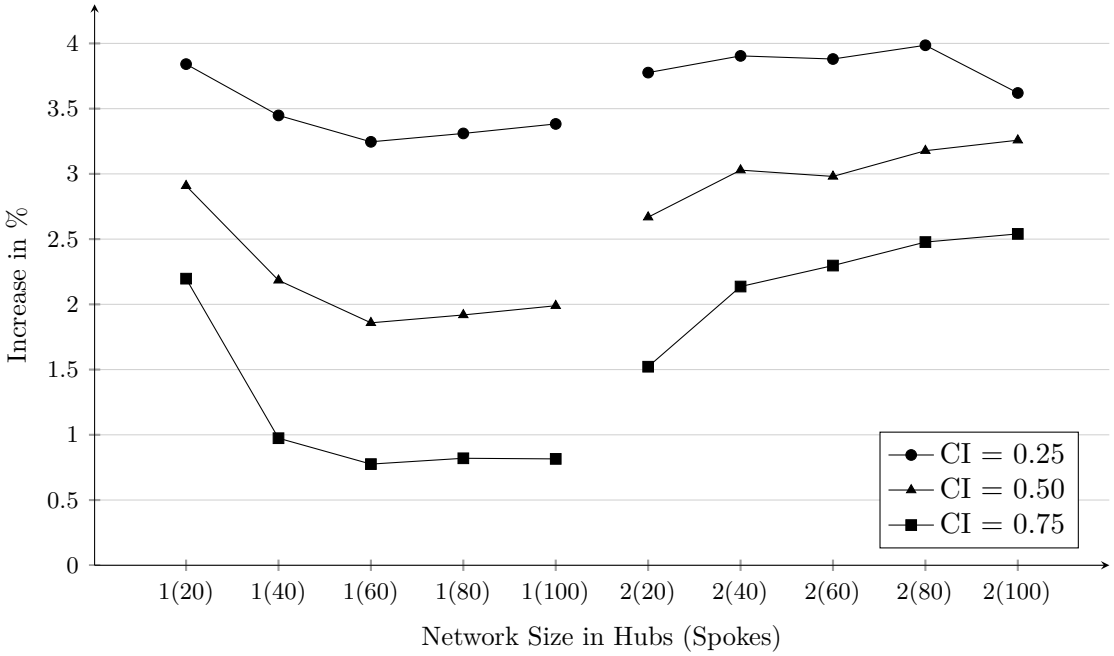


Figure 6.8: Average Payoff Increases From Step 2 to Step 3 with Monopolistic Prices for Player 2

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Step 2/Step 3	96.39%	96.78%	96.86%	96.79%	96.75%	96.39%	96.21%	96.24%	96.17%	96.49%
	NC/Step 3	96.24%	96.56%	96.64%	96.59%	96.54%	96.26%	96.10%	96.12%	96.04%	95.99%
	NC/Step 2	99.85%	99.78%	99.77%	99.79%	99.79%	99.86%	99.89%	99.87%	99.86%	99.49%
0.50	Step 2/Step 3	97.19%	98.04%	98.14%	98.11%	98.09%	97.42%	97.05%	97.05%	96.94%	96.83%
	NC/Step 3	97.03%	97.83%	97.94%	97.90%	97.87%	97.28%	96.93%	96.93%	96.82%	96.71%
	NC/Step 2	99.84%	99.79%	99.80%	99.78%	99.78%	99.86%	99.88%	99.87%	99.87%	99.87%
0.75	Step 2/Step 3	97.88%	99.17%	99.23%	99.23%	99.22%	98.55%	97.90%	97.73%	97.59%	97.51%
	NC/Step 3	97.75%	98.97%	99.03%	99.03%	99.03%	98.27%	97.77%	97.63%	97.50%	97.40%
	NC/Step 2	99.87%	99.80%	99.80%	99.79%	99.80%	99.72%	99.88%	99.90%	99.90%	99.89%

Table 6.2: Average Payoff Ratios with Monopolistic Prices

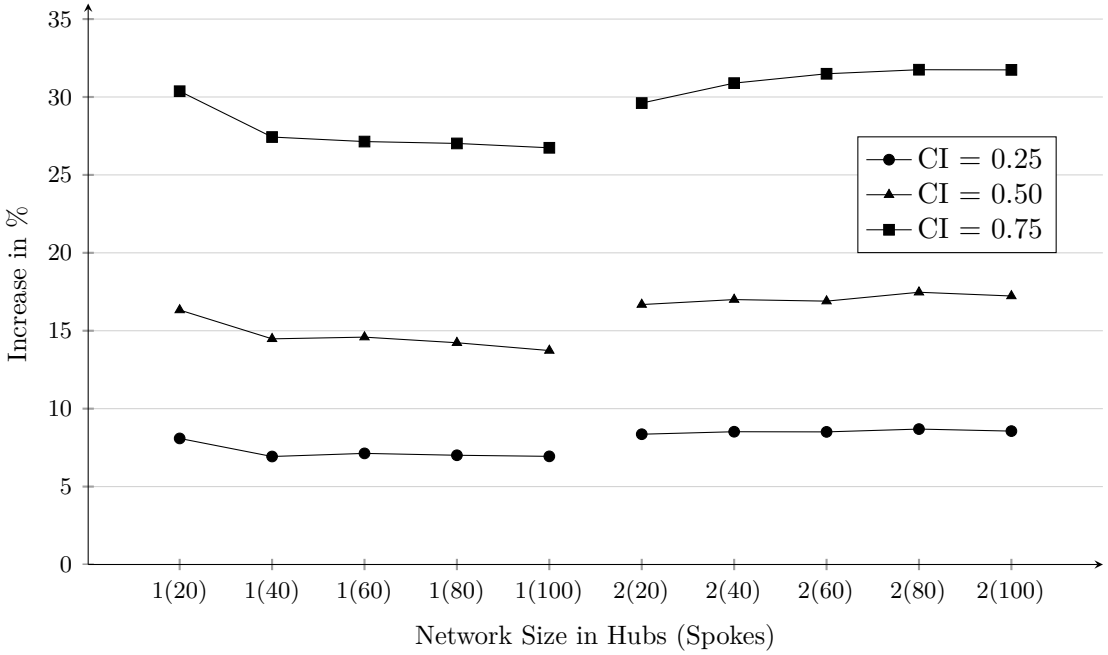


Figure 6.9: Average Payoff Increases in Step 2 with Competitive vs. Monopolistic Prices for Player 1

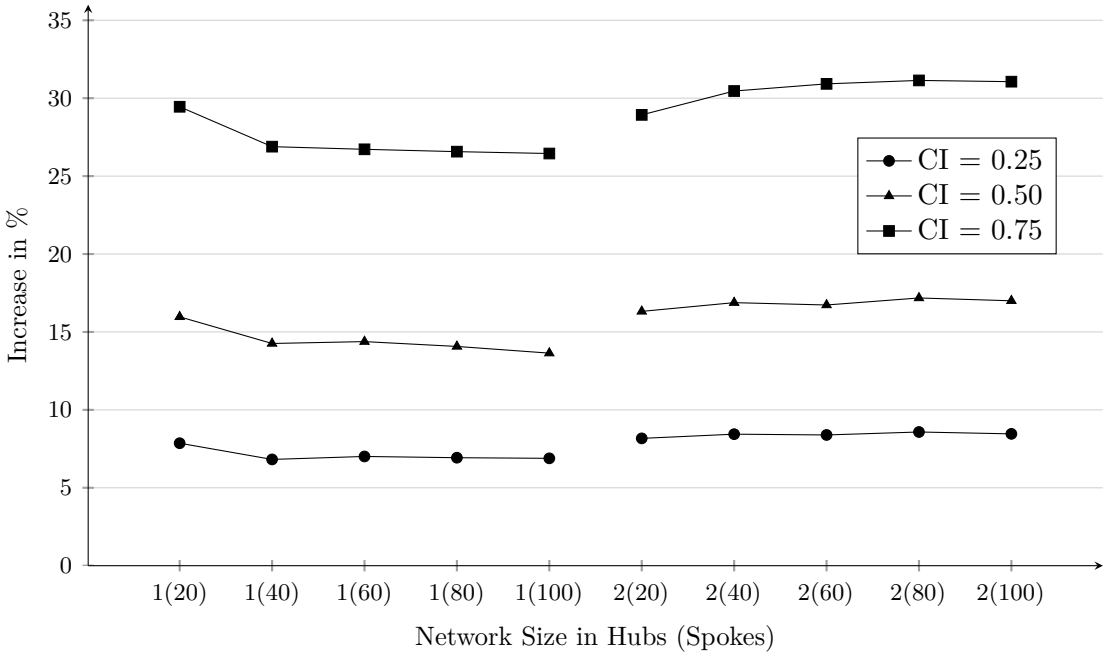


Figure 6.10: Average Payoff Increases in Step 2 with Competitive vs. Monopolistic Prices for Player 2

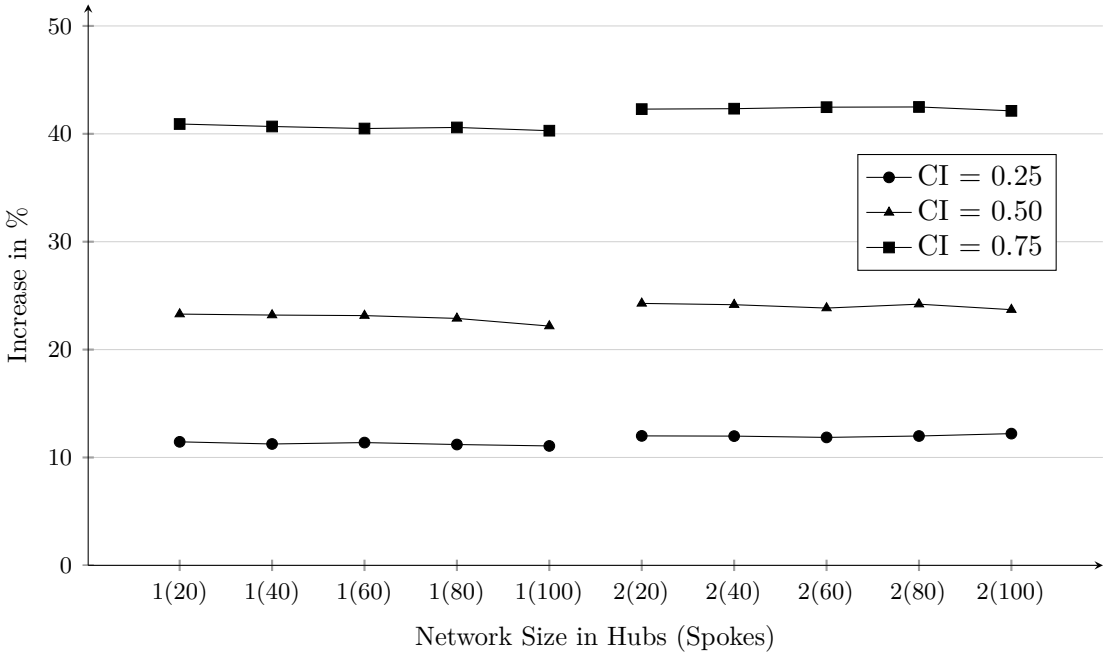


Figure 6.11: Average Payoff Increases in Step 3 with Competitive vs. Monopolistic Prices for Player 1

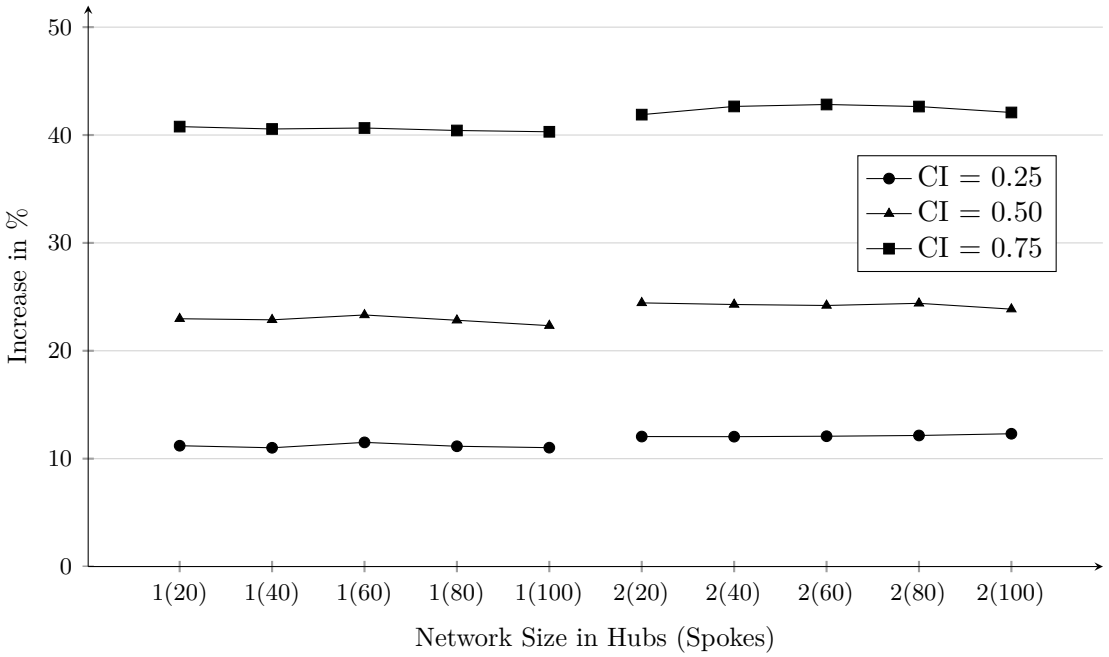


Figure 6.12: Average Payoff Increases in Step 3 with Competitive vs. Monopolistic Prices for Player 2

Chapter 7

Competition within Strategic Alliances

As mentioned in the introductory chapters, the deregulation of the airspace in the US, followed by deregulations in other countries in the past two decades, allowed airlines to enlarge their networks (Oum and Park, 1997). In order to avoid many of the efforts connected to the entry into new markets, airlines formed alliances. On the one hand, these cooperations brought along many benefits, on the other hand they lead to a new form of competition, namely the competition between partners.

This chapter, based on Grauberger and Kimms (2014d), models such competition and provides insights into the equilibrium outcomes of such situations. We will first give an overview of the relevant literature in Section 7.1 before we present the competitive model in Section 7.2. Section 7.3 presents the computational study. See e.g. Oum and Park (1997) and Park (1997) for further reasons for alliance formation as well as its (economic) effects. Chapter 3.2 in Çetiner (2013) provided a detailed treatment of this topic as well as a thorough literature overview.

7.1 Related Literature

A key characteristic of airline alliances are code sharing agreements that allow the airlines to sell products which involve utilization of partners' capacities as if they were their own (Oum et al., 2001, p. 57). Code sharing allows the partners to extend their networks, improve customer service, and raise their efficiency through a higher capacity load factor, among other things. With code sharing agreements in use, the problem is not only to allocate capacity to different products, but also to divide the available capacity among the partners within the alliance.

O'Neal et al. (2007) presented a mixed-integer problem to select those flights which should be made available for code sharing. Given these decisions, it must be decided how much of an airline's capacity should be made available to the alliance partners. Graf and Kimms (2011, 2013) have developed procedures based on real options to solve this problem for a two-airline alliance. However, with more than one airline involved in the sale of tickets, the problem of how to divide the profit amongst them arises. Kimms and Çetiner (2012) as well as Çetiner and Kimms (2013) introduced procedures to allocate the alliance's revenue among the partners in fair ways so that none of them has an incentive to leave the alliance. Their procedures are based on the nucleolus concept from cooperative game theory and turned out to be very effective. Topaloglu

(2012) proposed a decomposition approach for determining alliance booking limits and transfer prices based on a centralized DLP. Belobaba and Jain (2013) described the technical difficulties involved in the information sharing process faced by alliance RM and proposed information sharing mechanisms to overcome these.

All these authors assumed a cooperative attitude on the side of the partners. However, despite cooperating in certain aspects, the alliance members often remain competitors in other aspects and strive for revenue maximization. In this chapter, we investigate the problem of two airlines that on the one hand cooperate within an alliance but on the other hand continue to compete for customers. In this setting, two types of competition arise which are called horizontal and vertical competition in Netessine and Shumsky (2005) who first took this circumstance into account. We have already defined these concepts in Chapter 5.1 and re-state these definitions here for the sake of readability.

In *vertical competition* two airlines have to decide on the number of seats to reserve for connecting (code shared) passengers changing planes at a stopover city. Different legs of a multi-leg itinerary are operated by different airlines. Thus, one airline can sell tickets for products which occupy the partner's aircraft. In this setting the airlines must choose how many seats to protect for local and connecting passengers in absence of cooperation or coordination. An airline's booking limit for code-shared tickets is thus affected by the partner's booking limit for them. In *horizontal competition*, on the other hand, the competitors offer identical substitutable products and customers can request a ticket from the competitor if they are denied by their preferred airline.

Wright et al. (2010) also investigated some competitive issues that arise within an alliance, but their approach is different. Namely, they ignored horizontal competition and focused only on the vertical competition the airlines face. Secondly, they did not compute the players' booking limits but rather assumed that the airlines decided for every request whether it should be accepted or not within a Markov game. In this case, the authors provided rules based on bid prices for accepting or denying a request. All in all, the authors focused on examining which type of revenue sharing agreements, static or dynamic, generated the highest revenues to the alliance while providing incentives for the airlines to stay in the alliances. Wright (2014) implied incomplete information for the case that the alliance partners were unable or unwilling to share certain information concerning code sharing. He introduced a decomposition rule for a central dynamic program to determine approximate bid prices for the individual alliance partners.

The approach taken in Grauberger and Kimms (2014d) and presented in this chapter is more related to that of Netessine and Shumsky (2005) who were the first ones to use non-cooperative game theory for determining optimal booking limits for code shared products in an alliance. However, they considered only horizontal or only vertical competition and focused on one-leg and two-leg itineraries only, respectively. Further, the authors considered only two fare classes. Hu et al. (2013) used the same setup, yet they described the alliance formation and operation process as a two-stage game. In the first stage, a cooperative game was used to determine optimal revenue sharing rules for code shared products, while the second stage uses a non-cooperative game to determine optimal booking limits for all products using the airlines' legs. The authors focused on revenue sharing mechanisms that lead to maximal revenues for the complete alliance and modeled the individual partners' decisions so that their models incorporated the central solution. We, on the other hand, consider simultaneous horizontal and vertical competition in $\mathcal{F}$ classes. To the best of our knowledge, we are the first to address both of these types of competition simultaneously within airline alliance networks. Decisions about revenue sharing

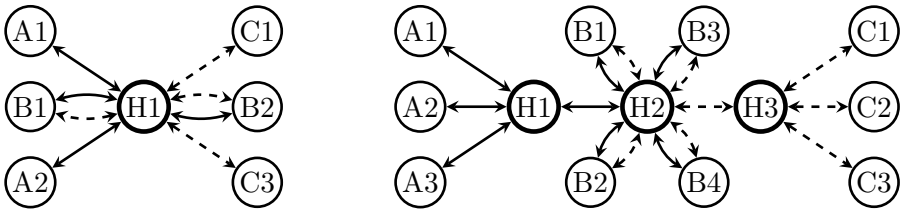
scheme are not treated here.

Transchel and Shumsky (2012) introduced a closed-loop dynamic pricing game for alliance partners that operate a parallel and substitutable flight. On the one hand the competitors are assumed to compete horizontally on this flight while on the other hand they have to set prices for their local and for the their code-shared products. We do not consider the situation in which both partners operate the same route and at the same time share codes on it. Instead, we assume that products subject to horizontal competition are not subject to vertical competition and vice versa.

7.2 Model Formulation

As the previous sections showed, the amount of research considering competition in airline alliances is close to nothing. The mentioned publications are very restricted in use since the competitors operate different legs. To the best of our knowledge, no publication so far considered simultaneous horizontal and vertical competition within alliances on a network with $\mathcal{F}$ classes. We intend to fill this gap here. The model we present here is another extension of the DLP.

We will define the further notions with the help of Figure 7.1 which shows simplified versions of the networks used in the computational study below. The solid and dashed arcs represent the different airlines' networks L^a, respectively. We do not show connecting routes, but these are certainly available in the networks, e.g. a flight from A1 to A2 with a stopover in H1. As the figure shows, in order for code-sharing to be applicable, the networks have to have a common connecting airport at which the code shared customers can switch from one airline's plane to the other's. In our case the connecting hubs are the hubs H1 in Figure 7.1a) and H2 in Figure 7.1b).



a) Network with One Hub b) Network with Three Hubs

Figure 7.1: Simple Alliance Network Structures

The O&D pairs for which the players' demand is affected by horizontal competition are contained in the set $A = \{P^a \cap P^{-a}\}$. These are made up of itineraries with the same origin and destination offered by both competitors, e.g. itinerary B1–H1 in Figure 7.1a). Since we have only two competitors, we need no superscript denoting the player for this set. We assume that both players offer the same fare classes and that competition and code sharing thus affects all classes of an O&D pair.

O&D pairs with demand not affected by horizontal competition make up the set $NA^a = \{P^a \backslash P^{-a}\}$ and are indicated by arcs without overlapping legs by the competitor. I.e. considering a player's itinerary O – D, the competitor does not offer a substitute itinerary with the same origin and destination. These itineraries are available for vertical competition. E.g. O&D pairs A1–H1 in Figure 7.1a) and A1–H2 in Figure 7.1b) belong to the set NA^a with the solid arcs

belonging to player a's network. This set includes the set $CS^a \subseteq NA^a$ of itineraries available for code-shared products. One part of a code-shared itinerary is carried out by one airline and another part is carried out by the competitor.

To simplify modeling, we further divide the set CS^a into the sets of inbound itineraries $In^a \subseteq CS^a$ originating in the competitor's network and outbound itineraries $Out^a \subseteq CS^a$ originating in player a's own network. E.g. the itineraries from H1 to A1 or to A2 in Figure 7.1b) belong to the set In^a of player a if the journey originated in the competitor's network, while the itinerary from A1 or from A2 to H1 belongs to this player's set Out^a if the final destination lies in the competitor's network. A code-shared itinerary is a combination of one player's outbound and the other player's inbound itinerary.

The initial demand player a expects to receive for a non-code-shared product during the booking period is d^a_{pf}. The demand for code-shared products is $\hat{d}_{pqf}$ and is not differentiated after players. It is differentiated by the itinerary p served by one player, the itinerary q served by the competitor and the fare class f. Player a's revenue for a non-code shared product is π^a_{pf} and the revenue for a code-shared-product is $\hat{\pi}^a_{pqf}$. For horizontal competition, the proportion of customers requesting a product from player a if they are denied a ticket by their preferred airline $-a$ is $\alpha^{-a,a}_{pf} \in [0,1]$ which is determined in the same was as in Section 5.3.5.1 in which only horizontal competition was taken into account.

The decision variables for airline a are its booking limits b^a_{pf} for its non-code-shared products and $\hat{b}^a_{pqf}$ for its code shared products. All in all, airline a solves model $\mathcal{M}^a_{CS}$ consisting of (7.1) – (7.9) in response to the competitor's behavior:

$$\mathcal{M}^a_{CS}\text{: max } r^a(b^a_{pf}, \hat{b}^a_{pqf}) = \sum_{p=1}^{\mathcal{P}^a}\sum_{f=1}^{\mathcal{F}} \pi^a_{pf} b^a_{pf} + \sum_{p\in In^a}\sum_{q\in Out^{-a}}\sum_{f=1}^{\mathcal{F}} \hat{\pi}^a_{pqf}\hat{b}^a_{pqf} + \sum_{p\in Out^a}\sum_{q\in In^{-a}}\sum_{f=1}^{\mathcal{F}} \hat{\pi}^a_{pqf}\hat{b}^a_{pqf} \tag{7.1}$$

$$\text{subject to } b^a_{pf} \le d^a_{pf} \qquad p\in NA^a, f\in F \tag{7.2}$$

$$b^a_{pf} \le d^a_{pf} + \lfloor \alpha^{-a,a}_{pf}(d^{-a}_{pf} - b^{-a}_{pf})^+ \rfloor \qquad p\in A, f\in F \tag{7.3}$$

$$\hat{b}^a_{pqf} \le \min\{\hat{d}_{pqf}; \hat{b}^{-a}_{pqf}\} \qquad p\in In^a, q\in Out^{-a}, f\in F \tag{7.4}$$

$$\hat{b}^a_{pqf} \le \min\{\hat{d}_{pqf}; \hat{b}^{-a}_{pqf}\} \qquad p\in Out^a, q\in In^{-a}, f\in F \tag{7.5}$$

$$C^a_l \ge \sum_{p=1}^{\mathcal{P}^a}\sum_{f=1}^{\mathcal{F}} M^a_{lp} b^a_{pf} + \sum_{p\in In^a}\sum_{q\in Out^{-a}}\sum_{f=1}^{\mathcal{F}} M^a_{lp}\hat{b}^a_{pqf} + \sum_{p\in Out^a}\sum_{q\in In^{-a}}\sum_{f=1}^{\mathcal{F}} M^a_{lp}\hat{b}^a_{pqf} \qquad l\in L^a \tag{7.6}$$

$$b^a_{pf} \ge 0 \qquad p\in P^a, f\in F \tag{7.7}$$

$$\hat{b}^a_{pqf} \ge 0 \qquad p\in In^a, q\in Out^{-a}, f\in F \tag{7.8}$$

$$\hat{b}^a_{pqf} \ge 0 \qquad p\in Out^a, q\in In^{-a}, f\in F \tag{7.9}$$

The objective function (7.1) maximizes the total revenue of airline a summing over the revenues for the products times their booking limits. Constraint (7.2) limits the booking limit for a product which is not available for code sharing and not affected by horizontal competition to this product's demand. Restriction (7.3) limits the booking limit for a product affected by

horizontal competition to its rounded down total expected demand. Rounding down the values again leads to all parameters on the right hand side being integer-valued which allows for defining the variables continuously and still receiving integer solutions.

Constraints (7.4) and (7.5) restrict the booking limits for the code-shared products which are required to not exceed the minimum of the demand for this product and the competitor's booking limit for it. Restriction (7.6) is the capacity restriction requiring the sum of all booking limits for products utilizing a player's leg not to exceed the airplane's capacity on this leg. Finally, Restrictions (7.7) – (7.9) define the booking limits to be non-negative continuous numbers.

Extending this model to consider more then two players is slightly more complicated than with the models in the previous chapters. As before, the index $-a$ would have to stand for all competitors in set $\mathcal{A}$ except for the considered player a and the set A would have to be extended to A^a in order to differentiate the products affected by horizontal competition by players. Constraint (7.3) would have to cover the sum of spill over horizontal demand denied by all competitors. Assuming that all alliance partners have access to all others' network through a common airport, Constraints (7.4) and (7.5) would have to be extended to cover all competitors's outbound or inbound products affected by vertical competition, respectively

7.3 Computational Study

We used the model from the previous section to compute pure NE with Algorithm 5.1 presented in Section 5.3 and compared the results with those from the case when a central decision maker controlled the capacity. Although assuming complete information for competition within alliances is somewhat unrealistic due to technical or legal aspects concerning alliance cooperation (see e.g. Boyd, 1998, Shumsky, 2006, and Vinod, 2005), it simplifies the analysis and allows focusing on the main issues. This assumption was also made in e.g. Netessine and Shumsky (2005), Li et al. (2008).

7.3.1 Test Bed

We assumed two airlines with similar, overlapping networks as in Figure 2.1 to form an alliance. The number of hubs in the players' networks was identical, i.e. either both had one hub or both had two. Both airlines' networks were again assumed to have numbers of 20, 40, 60, 80, and 100 airports to be connected with each hub. In this case, the competition intensity CI (25%, 50%, and 75% of an airline's total offered itineraries) again stood for the percentage of O&D pairs affected by horizontal competition. We assumed that products affected by horizontal competition were not subject to code shares and vice versa. The code shared products had to originate in one player's network and had to end in the competitor's network and affected only those itineraries which were not affected by horizontal competition and were available for code sharing. Furthermore, if an itinerary entailed a part served by both players and a part served by only one airline (like e.g. B1–C1 in the above figures), we assumed that the whole itinerary was served by the latter airline. I.e. the mentioned itinerary B1–C1 was not affected by competition but was completely carried out by the airline with the dashed marked network. With two hubs in each of the players' networks, we assumed the networks to be connected only in hub H2.

Out of all possible non-code-shared products, we chose randomly so many that each leg was demanded by at most 60 products. For these products again ten pairs of original demand d^a_{pf} for

non-code-shared products were generated randomly from the Poisson distribution with means of $\mu \in \{2, 4, 6\}$ for each network setting. Since these are the same assumptions as in Section 5.3.5.1, the demand-to-capacity ratios for these products were the same as above. The demand $\hat{d}_{pqf}$ for the code shared products was assumed to have a mean of $\mu = 1$. Depending on the value of CI, the competition intensity, the code shared products raised the demand-to-capacity ratios by at most $\frac{60*(1-CI)}{C_l}$.

The assumptions concerning the product prices for the non-code-shared products and the ratios for the spillover demand in the horizontal competition were the same as in Section 5.3.5.1. We assumed that a code-shared product's revenue contributed to the operating airlines' total revenue as much as if the airline sold the corresponding single-leg or two-leg flight to local passengers, which was also assumed in Netessine and Shumsky (2005). This allowed for less differentiation of a player's inbound code shared products since they all generated him the same revenue, no matter where they originated in the competitor's network. Hence, we needed only one variable per class of an inbound code shared itinerary which enabled us to replace Constraint (7.4) with Constraint (7.10).

$$\hat{b}^a_{pf} \leq \min \left\{ \sum_{q \in Out^{-a}} \hat{d}_{pqf}; \sum_{q \in Out^{-a}} \hat{b}^{-a}_{pqf} \right\} \qquad p \in In^a, f \in F \tag{7.10}$$

This lead to a reduction of the number of variables needed to model a player's booking limits for his inbound code shared products from the combination $|Out^{-a}|*|In^a|*\mathcal{F}$ to the combination $|In^a| * \mathcal{F}$. This bundling is motivated by the player operating the inbound itineraries merely communicating the number of seats available for code sharing and the competitor having to decide with which of his outbound itineraries he wants to use these seats. The booking limits for the outbound code-shared products could not be bundled in this manner (i.e. irrespective of their final destination) because they had different revenues connected to them and the players had to be able to differentiate between the individual outbound products in case the competitor's inbound booking limit for a product was lower than the total demand for it.

Having the same setup relevant for the perturbing parameter τ^a, it again was set to 0.9999 and the prices were were multiplied by 300 before the optimization. All in all, we considered the same 1,800 instances as in Section 5.3.5 with the difference being the additional vertical competition.

7.3.2 Results

The computational results show that the algorithm presented in Chapter 5.3 is suited for real-sized airline networks given in two-partner alliances and can compute NE with simultaneous horizontal and vertical competition. All in all, an NE was found in 99.33% (1,788 out of 1,800) instances. Next, we again only show the results for the instances with the prices perturbed in order "increasing by fare class, then alphabetic by O&D name". The results for perturbation order "alphabetic by O&D name, then increasing by fare class" can be found in Appendix A.4. The results for both perturbation orders are similar, though. Figures 7.2 – 7.4 show the average computation times in minutes needed for finding a pure NE in the different networks with different competition intensities (CI) as ratios for the products affected by horizontal competition.

The computation time needed to find an NE rose with the network size, the mean demand, and the competition intensity. While an NE was found in seconds in the smallest networks with

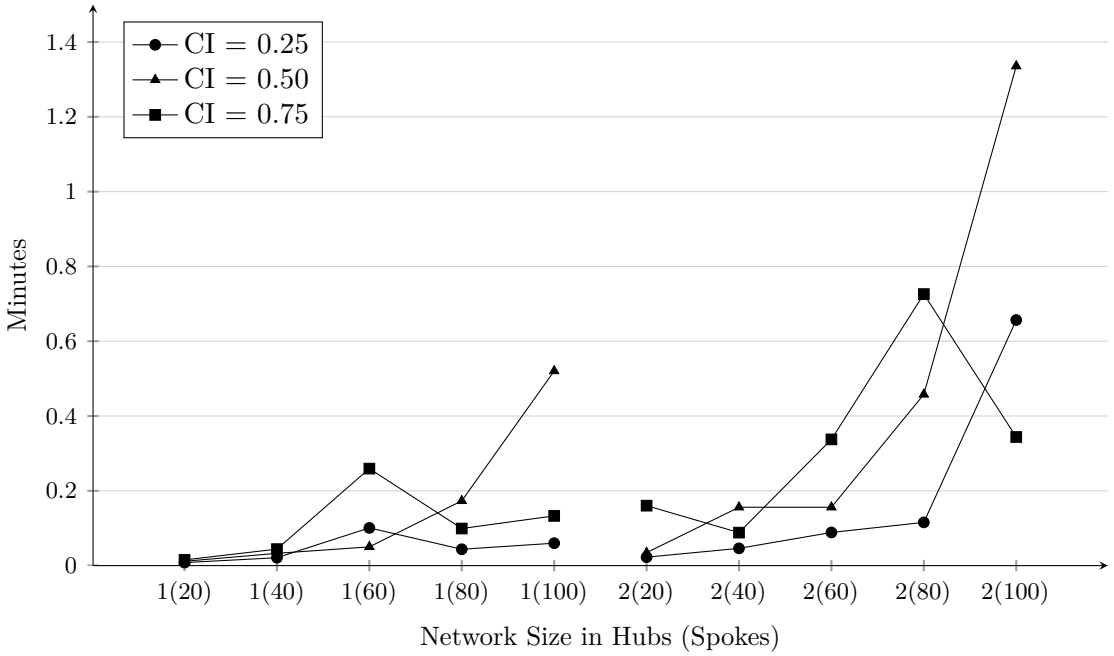


Figure 7.2: Average Computation Times in Minutes with $\mu = 2$

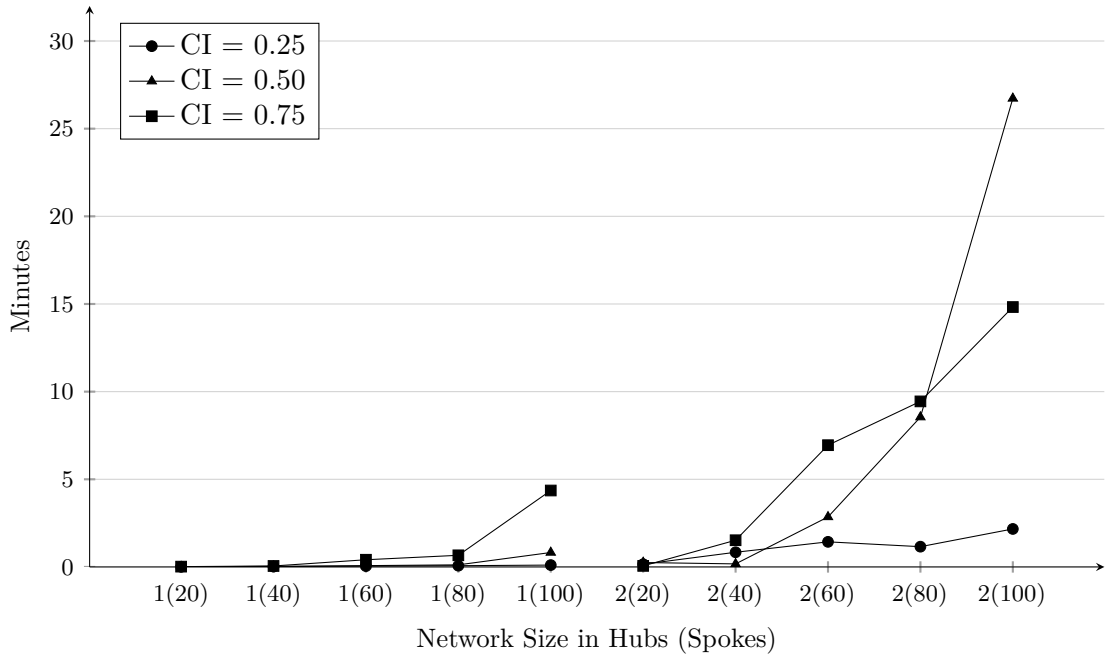


Figure 7.3: Average Computation Times in Minutes with $\mu = 4$

a mean demand of $\mu = 2$, in the largest networks with a mean demand of $\mu = 6$, and CI = 0.75 about 39 minutes were needed on average to find an NE. The most time needed by the algorithm to terminate with an NE was 161 minutes. Such "outliers" are responsible for the graphs being not monotone since they raise the average. All in all, the computation times are acceptable even for large networks and are similar to the times without vertical competition. Yet, vertical competition adds complexity to the problem since more products need to be allocated to

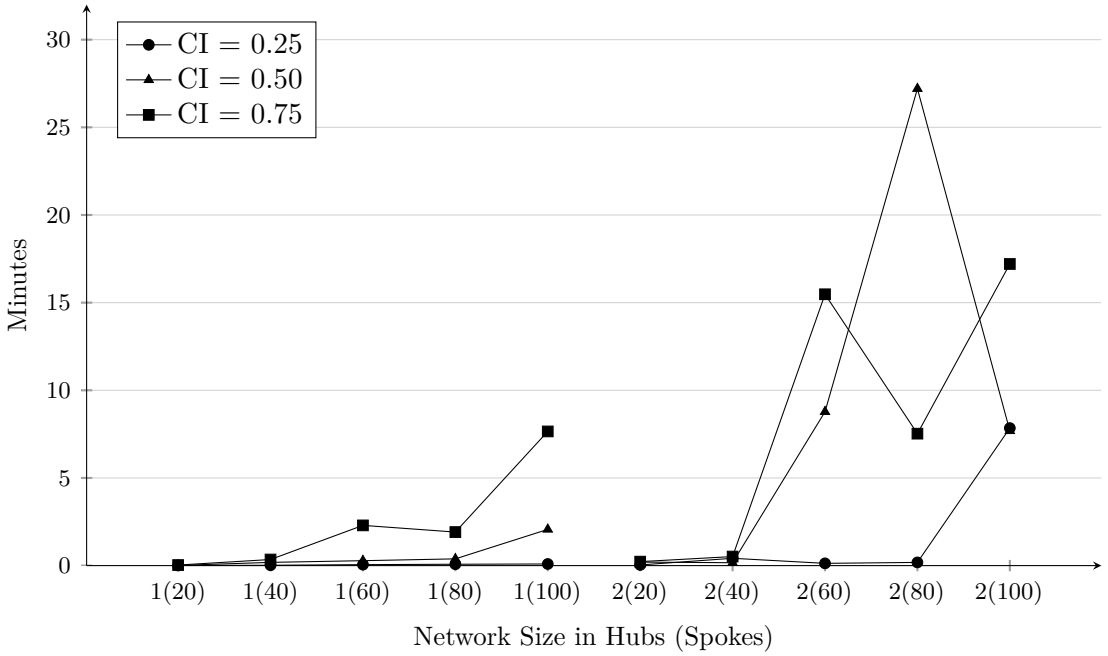


Figure 7.4: Average Computation Times in Minutes with $\mu = 6$

the individual resources. Hence, the computation times needed to find NE under simultaneous horizontal and vertical competition are higher.

Looking at the average numbers of iterations needed for Algorithm 5.1 to terminate with simultaneous horizontal and vertical competition again makes evident the similarity with the results without vertical competition. These results are shown in Figures 7.5 – 7.7. The number of iterations needed to terminate again increased with a higher mean demand, larger networks and higher competition intensity. Even the paths of the graphs are similar to those in Section 5.3.5.2. However, again the additional vertical competition is responsible for the increased average number of iterations. Merely the maximal numbers are higher in Section 5.3.5.2.

We compared the summed equilibrium payoffs to the cooperative (central) payoff and to the summed non-competitive payoffs, i.e. when both airlines ignored competition and did not take into account neither spill over nor code shared demand. For the cooperative case the prices for the products affected by competition were set as average prices charged by the players. The other products' prices were taken directly as the player's respective price. The products' demands and the capacities on the legs used by these products were taken as the sum of the players' values. The results are shown in Tables 7.1 – 7.3. Here, C, NE, and NC stand for central, equilibrium summed, and non-competitive summed payoffs, respectively. The entries in the first row of columns 3 – 12 stand for the networks as "Hubs(Spokes)". The revenues were computed without the perturbing parameters τ. These were not needed for the cooperative case and for comparing the values with each other, we used the non-perturbed prices for the other settings as well.

As the tables show, the summed competitive and non-competitive payoffs were usually lower that the central payoffs. The payoffs in the NE were slightly higher than those in the non-competitive situations, though. This is due to Constraints (7.3) – (7.5) which take into account a player's spillover and code shared demand and increase a player's demand and thus payoff. This effect increases with a higher competition intensity because more products' demands are

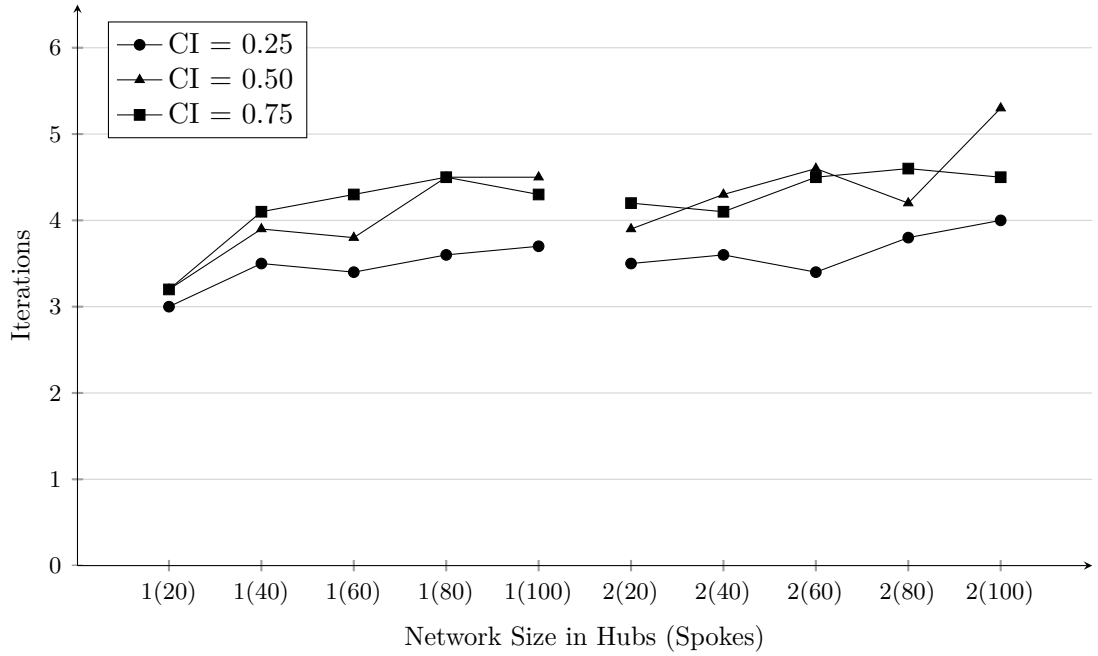


Figure 7.5: Average Numbers of Iterations with $\mu = 2$

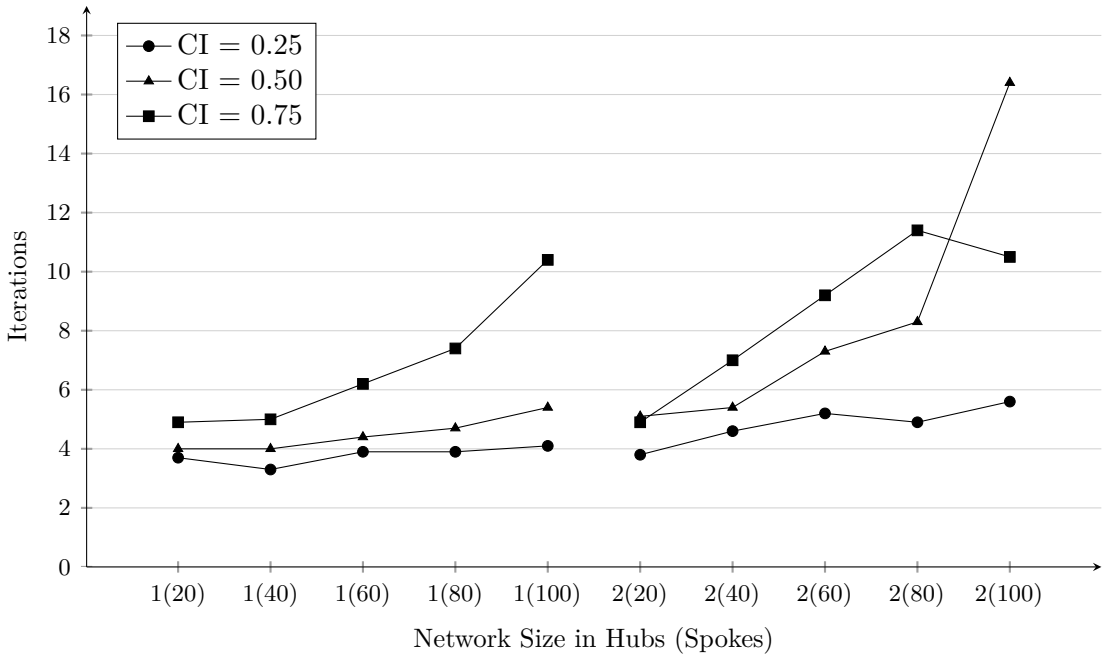


Figure 7.6: Average Numbers of Iterations with $\mu = 4$

potentially raised through spillover and code shared demand. The higher the competition intensity (the more products are affected by spillover demand), the higher an airline's NE payoff and the lower the ratio between the NE payoffs and the non-competitive payoffs. However, the difference between the central payoff compared to the competitive and non-competitive ones decreases with a higher demand and CI indicating that with a high demand and/or competition intensity, the revenues in the decentralized decisions are almost as high as those in the centralized case.

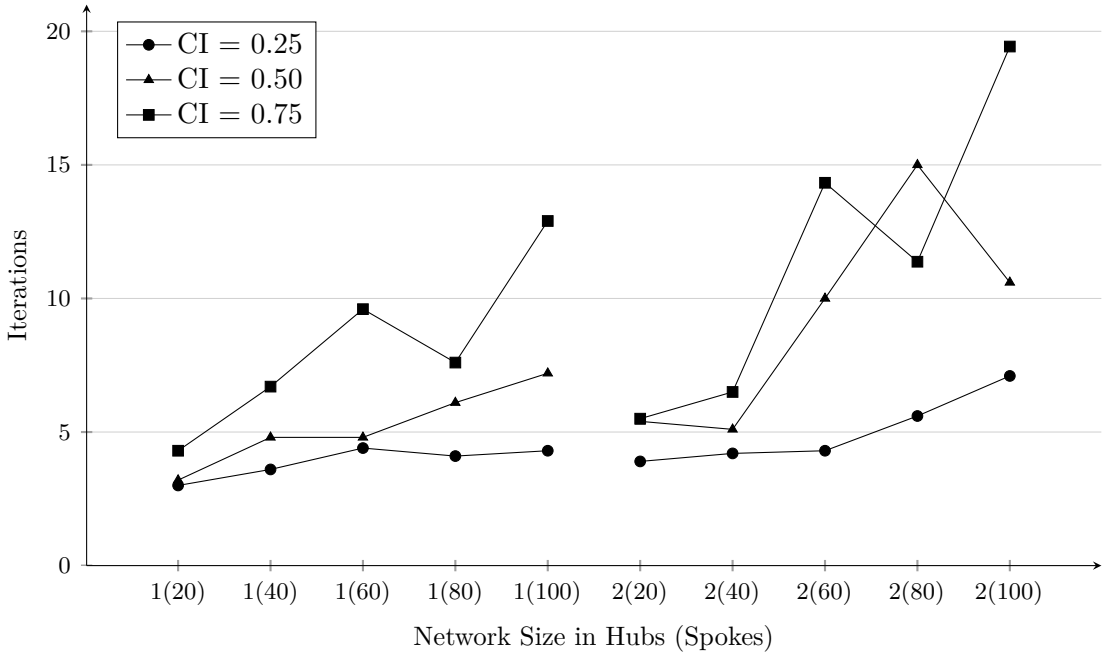


Figure 7.7: Average Numbers of Iterations with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	91.44%	91.65%	95.39%	95.07%	97.10%	97.46%	97.95%	99.33%	99.37%	98.92%
	NC/C	81.27%	91.04%	95.13%	94.16%	96.75%	87.27%	92.23%	96.22%	96.08%	96.91%
	NC/NE	88.87%	99.33%	99.72%	99.04%	99.64%	89.55%	94.17%	96.87%	96.69%	97.96%
0.50	NE/C	94.56%	92.51%	96.02%	95.56%	97.54%	97.97%	98.32%	99.58%	99.47%	99.00%
	NC/C	83.57%	91.91%	95.75%	94.59%	97.12%	88.32%	92.24%	96.22%	96.00%	96.91%
	NC/NE	88.38%	99.35%	99.72%	98.99%	99.57%	90.15%	93.82%	96.63%	96.52%	97.89%
0.75	NE/C	96.38%	95.73%	98.96%	97.94%	99.21%	99.19%	99.66%	99.69%	99.73%	99.59%
	NC/C	91.00%	95.31%	98.83%	97.32%	99.04%	93.78%	97.44%	97.67%	97.87%	98.54%
	NC/NE	94.42%	99.56%	99.88%	99.36%	99.83%	94.55%	97.77%	97.98%	98.13%	98.95%

Table 7.1: Average Payoff Ratios with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	94.67%	97.88%	98.94%	98.50%	99.33%	95.94%	97.88%	98.95%	98.83%	99.27%
	NC/C	94.58%	97.82%	98.87%	98.42%	99.25%	95.82%	97.80%	98.85%	98.68%	99.20%
	NC/NE	99.90%	99.93%	99.92%	99.92%	99.92%	99.88%	99.92%	99.90%	99.85%	99.93%
0.50	NE/C	95.44%	98.36%	99.29%	98.85%	99.61%	96.66%	97.99%	99.08%	98.92%	99.37%
	NC/C	95.27%	98.21%	99.13%	98.69%	99.46%	96.48%	97.86%	98.91%	98.73%	99.25%
	NC/NE	99.82%	99.85%	99.84%	99.84%	99.84%	99.81%	99.87%	99.83%	99.81%	99.88%
0.75	NE/C	97.94%	99.35%	100.15%	99.74%	100.21%	98.55%	99.61%	99.58%	99.53%	99.78%
	NC/C	97.71%	99.16%	99.94%	99.52%	99.98%	98.34%	99.43%	99.39%	99.35%	99.63%
	NC/NE	99.77%	99.81%	99.79%	99.78%	99.78%	99.79%	99.82%	99.81%	99.81%	99.85%

Table 7.2: Average Payoff Ratios with $\mu = 4$

Note, however, that here the same restrictions apply to the comparability of these values as in Section 5.3.5.2 and that hence the cooperative payoffs are only for relative comparison with the other values.

We also compared the central and NE payoffs including code shared products and horizontal competition with the corresponding payoffs without code shared products and only horizon-

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	97.48%	99.20%	99.74%	99.39%	99.88%	98.17%	99.16%	99.67%	99.54%	99.73%
	NC/C	97.40%	99.08%	99.61%	99.28%	99.76%	98.10%	99.10%	99.58%	99.44%	99.66%
	NC/NE	99.91%	99.87%	99.87%	99.89%	99.88%	99.93%	99.93%	99.91%	99.91%	99.92%
0.50	NE/C	98.06%	99.63%	100.11%	99.77%	100.24%	98.68%	99.38%	99.82%	99.70%	99.90%
	NC/C	97.91%	99.38%	99.85%	99.53%	99.99%	98.52%	99.23%	99.66%	99.54%	99.74%
	NC/NE	99.84%	99.75%	99.74%	99.76%	99.75%	99.84%	99.85%	99.85%	99.83%	99.84%
0.75	NE/C	99.35%	100.35%	100.73%	100.40%	100.73%	99.70%	100.20%	100.12%	100.08%	100.18%
	NC/C	99.16%	99.95%	100.33%	100.03%	100.34%	99.47%	99.99%	99.89%	99.86%	99.96%
	NC/NE	99.81%	99.60%	99.61%	99.63%	99.61%	99.78%	99.79%	99.78%	99.77%	99.78%

Table 7.3: Average Payoff Ratios with $\mu = 6$

tal competition. Tables 7.4 – 7.6 show these results. The entries therein were computed as $\frac{\text{Payoff with code sharing}}{\text{Payoff without code sharing}}$. One can see that code sharing tendentially leads to higher payoffs. Including code shared products never lead to lower payoffs than ignoring it, which is obvious since the Constraints (7.4) and (7.5) can only affect the revenue positively.

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	C	122.90%	109.75%	105.04%	106.12%	103.30%	114.52%	108.41%	103.90%	104.00%	103.10%
	NE	112.48%	100.62%	100.24%	100.92%	100.32%	111.60%	106.14%	103.19%	103.38%	102.04%
0.50	C	119.48%	108.71%	104.39%	105.68%	102.95%	113.29%	108.48%	103.93%	104.12%	103.12%
	NE	113.06%	100.57%	100.21%	100.94%	100.35%	110.77%	106.49%	103.40%	103.53%	102.08%
0.75	C	109.72%	104.84%	101.11%	102.72%	100.97%	106.77%	102.78%	102.44%	102.21%	101.45%
	NE	105.82%	100.34%	100.02%	100.54%	100.07%	105.58%	102.15%	101.97%	101.82%	100.98%

Table 7.4: Average Payoff Ratios with and without Code Sharing with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	C	105.75%	102.30%	101.19%	101.62%	100.80%	104.43%	102.23%	101.11%	101.25%	100.74%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.03%	100.01%	100.03%	100.09%	100.01%
0.50	C	105.07%	102.03%	101.04%	101.48%	100.70%	103.91%	102.25%	101.13%	101.26%	100.75%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.01%	100.01%	100.05%	100.08%	100.02%
0.75	C	102.43%	101.11%	100.25%	100.70%	100.23%	102.02%	100.68%	100.67%	100.66%	100.37%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.03%	100.03%	100.00%

Table 7.5: Average Payoff Ratios with and without Code Sharing with $\mu = 4$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	C	102.81%	101.01%	100.49%	100.78%	100.34%	101.99%	100.94%	100.44%	100.55%	100.33%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
0.50	C	102.44%	100.91%	100.43%	100.71%	100.30%	101.75%	100.95%	100.45%	100.54%	100.33%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
0.75	C	101.19%	100.48%	100.08%	100.33%	100.09%	100.88%	100.26%	100.25%	100.28%	100.16%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%

Table 7.6: Average Payoff Ratios with and without Code Sharing with $\mu = 6$

It becomes clear once more, though that a centralized decision leads to better (i.e. more profitable) capacity control decisions than a decentralized competitive one. For, code sharing improves the revenues in the centralized cases more than the summed NE revenues. The tables also reveal that with a higher mean demand and a higher competition intensity the gain from code sharing decreases since there are less routes to share codes on. Instead, the horizontal competition is more dominant here. Also, with a higher mean demand the capacity load factor is already fairly high without code shared products which leaves less space for them.

Chapter 8

Conclusions and Future Research

This thesis treated revenue management under competition taking into account different aspects from a practical side and solved several conceptual issues from the (game-)theoretic side. Due to a twofold complexity of computing optimal seat allocations and exact NE in bimatrix games, we presented presented models to approximate an airline's optimal solution and developed algorithms to only compute best responses. The use of mostly linear models reduced the computational effort by searching for best responses directly and avoiding setting up the complete strategy and payoff matrices. In this chapter we sum up our approaches and present suggestions for further investigations.

8.1 Conclusions

Chapter 5 built the foundation for the competition models and algorithms for the thesis. In Section 5.2 we presented a simple model to optimize a competitor's behavior in a network capacity control problem under competition. There, we distinguished between O&D pairs affected by competition and those not affected and modeled the corresponding demands differently. Assuming that both competitors apply this model, an algorithm was presented in Section 5.3 to compute exact pure Nash equilibria. Pure NE are desirable because they allow a direct interpretation since demand values, strategies (booking limits), and capacity consumptions are integers. The complete algorithm finds a pure NE with certainty if one exists in the game. Finally, using a unique starting strategy and perturbed price vectors we resolved the issue of (dually) degenerated models and degenerated games and thus eliminated the coordination issues connected with multiple optimal solutions and/or NE. Thus, both competitors end up in the same NE when applying our algorithm and perturbing the prices. This makes our algorithm suitable for general games in which the uniqueness of an NE is not guaranteed or cannot be proved.

Our algorithm finds an NE through an iterative search for best responses by the competitors until a best response is chosen a second time. This is called forward search. If the strategy was chosen in the previous iteration for the first time, a pure NE is found. Otherwise the strategy is forbidden through a constraint in the current player's model and an alternative best response is looked for until another strategy is chosen a second time. If an alternative cannot be found, a backtracking procedure is initiated. It searches for alternatives for the best responses chosen before until one can be found and the forward search can continue or until iteration 1 is reached without an alternative best response for the starting player. In this case the algorithm terminates with the message that no pure NE exists in the game.

After illustrating the algorithm through an example, a computational study was conducted. It showed that a pure NE could be found in most instances within three hours and that computation time was acceptable even for big, real-sized networks—up to two hubs and 100 spoke airport connected to each hub—operated by each competitor.

The type of perturbation did not have a noticeable effect on the results. With a higher competition intensity and larger networks, more iterations were needed until the algorithm terminated which added some computation time as well. Comparison of the summed equilibrium payoffs to the summed non-competitive payoffs and the cooperative payoffs revealed that taking competition into account leads to tendentially higher payoffs and that the cooperative payoffs were highest. However, since prices in this case were averages of the competitors' prices and all of a competitor's denied customers are considered in the cooperative case instead of only a proportion as in the competitive case these results are not directly comparable.

For the case that an exact pure NE does not exist in a game or cannot be found, we presented a heuristic in Section 5.4.2 to compute approximate NE. Allowing to sacrifice an optimal solution (an exact NE) for a short computation time, we sped up the procedure and reduced computation times which were already fairly low for the exact algorithm. ed up the procedure and reduced computation times which were already fairly low for the exact algorithm. The essence of the heuristic is a reduction of the original game if an exact NE cannot be found by searching for best responses iteratively. In this case, some of the players' best responses are chosen repeatedly and the heuristic gets stuck in a loop. The game is then reduced to a bimatrix game based on the best response matrices where the strategies consist of only the best responses in the loop. In the reduced game, first all pure extreme NE are calculated and, if at least one exists, one is selected as the final solution to the original game. It was shown with a modified model that a pure NE in the reduced game is also an exact pure NE in the original game. In case no pure NE exists in the reduced game, all of its extreme mixed NE are computed and one of them is picked as the original game's approximate NE. With the reduced game's mixed NE, it was not possible to tell whether it is also an exact mixed NE in the original game because the reduced game only considers an excerpt of the original game.

After illustrating the heuristic through an example, a computational study has been conducted with the same test bed as was used for the exact algorithm. This enabled us to compare the results for the heuristic approximate NE with the results for the exact algorithm. All but three approximate NE were mixed and the player's payoffs in them were almost as high as the payoffs in the exact NE in all cases. However, concerning the computation time, the heuristic lead to a drastic reduction. In those games in which an exact NE could not be found by the forward search, the sizes of the BRMs were encouragingly small and the games based on them mostly had only one mixed NE making the choice easy. In only about 9% each the players had three and four strategies, respectively. The largest game had 28 strategies per player. An analysis of bounds for the players' payoffs in the approximate NE revealed very narrow ranges between the upper and the lower bounds.

All in all, the heuristic produced very good results in very short computation times. However, while it needed less time to terminate on average, it could lead to lower expected revenues than the exact algorithm and could terminate with mixed NE. While these results are acceptable if no pure NE exists or cannot be computed in an acceptable time, the exact algorithm is preferable to the heuristic because of generally long booking periods which compensate much of the longer computation times and the certainty of pure NE. Thus, when time is scarce, an approximate NE is a fairly good solution. However, if there is enough time left before takeoff, the exact algorithm

is the procedure to choose because it yields an exact pure NE with certainty if one exists.

In Chapter 6 simultaneous price and capacity competition with two players was considered. In Section 6.2 a non-linear model considering the new situation ans optimizing a competitor's behavior in a network revenue management game was treated. In the subsequent section we proposed an algorithm to compute approximate Nash equilibria in this setting. To avoid dealing with the model's non-linearity in the solution, the prices and capacity allocations were computed separately. We used the circumstance that under optimal prices, a product's spillover demand (the difference between effective demand and booking limit) would be zero since a higher price can be used to lower demand to the level of the booking limit. In effect, the competition takes place only on the price level, although prices and capacities are determined simultaneously. In a first step, we determined the optimal competitive prices ignoring capacity restrictions. With these prices, we determined the effective demand and computed a pure NE for the quantitative competition with an algorithm introduced in an earlier work. After finding an NE, the prices were updated in a third step with an LP which set the prices so that the products' demands amounted to the booking limits from Step 2. If the demands with the updated prices did not fall below the equilibrium booking limits from Step 2, the algorithm terminated, otherwise Steps 2 and 3 were repeated until the total revenue did not change significantly.

The updated prices led to significant (up to 11.89%) revenue increases compared to the revenues in the quantity NE within our settings. We compared the simultaneous competition situation with the situation when prices were set monopolistic by the competitors. Including price competition led to higher prices and revenues than ignoring it and the revenue increase was quite large with competitive prices versus monopolistic ones. While the results certainly depend on the parameter setting and the revenue gains need not always be so high, including competition leads to higher prices and revenues with our models.

Chapter 7 introduced a model to optimize a competitor's booking limits when he faces competition from an alliance partner under both, horizontal and vertical competition. In horizontal competition the competitors offer parallel and substitutable products and thus compete directly for customers, while in vertical competition the partners operate connecting, adjacent flights and one airline might sell tickets for products occupying the partner's aircraft. Here, a capacity competition arises because customers for the code shared products and local customers must be seated in the same plane. We distinguished the code shared products after their origin and destination into inbound and outbound products. Assuming that the code shared products generated the same revenue like the corresponding local products, we were able to reduce the number of decision variables for the inbound code shared products.

The proposed model was used in the exact algorithm from Section 5.3 to compute pure NE within airline alliances based on mutual best responses. We again compared the NE revenues with the case when a centralized decision maker controlled the capacity. We found that the central revenues were higher than in the NE, but that this difference decreased with the network size, the competition intensity, and mean demand. After all, with a high competition intensity, NE revenues were almost as high as in the centralized case indicating almost (Pareto) optimal NE. We also compared the situations with simultaneous horizontal and vertical competition to the results from Section 5.3.5.2 without code sharing and only horizontal competition. These comparisons revealed that code sharing benefits the partners because new demand is generated and raises the revenues. The revenue of the central decision maker increased most. Yet, again the benefits decreased with larger networks, higher competition intensities and higher mean demands.

We next present suggestions for future research. These include modeling of competitive situations as well as recommendations for solution approaches and might help the theory catch up with practical problems concerning RM under competition.

8.2 Future Research

There are several ways to construct more realistic scenarios and extend the algorithms for determining optimal decisions under competition for the RM case. From a modeling point of view, a natural extension of the problems considered here would be to consider customer choice. On the one hand, a customer could be willing to buy up or buy down from his preferred airline if his first-choice product is not available anymore. On the other hand, itineraries with a high demand can be served using different routes which would make necessary to consider intermediate stops along a trip from an origin to a destination to consider the customer choosing a different route. Mookherjee and Friesz (2008) as well as Lim (2009) have presented first attempts to tackle this issue. However, including buy-ups in single-airline models already leads to a huge increase of the computational burden, as shown in e.g. Talluri and van Ryzin, 2004a. Hence, the development of heuristics might be necessary to solve the resulting problem.

In Chapter 6, we assumed demand as a linear function of prices and the competitor's booking limits. While this is a common approach used for simplicity reasons, linear demand functions are not very realistic. In future work more realistic, i.e. non-linear (e.g. logit) functions should be used to model the demand. This would complicate the pricing, though which might necessitate an algorithm for setting the prices. Also, the assumption about equal sets of fare classes for both players could be relaxed and an investigation of how the results change if there is a "one to many", "many to one", or "many to many" correspondence between the sets of fare classes. Here, the effects on the demand function (6.3) would have to be considered as well. Since the approach proposed in Chapter 6 is a heuristic which does not guarantee to find an exact NE, future work should provide conditions under which our approach will find an exact NE with certainty.

We did not differentiate after which airline received a request for a code-shared product in Chapter 7. In reality this might matter and would intensify competition for capacity on the code-shared flights, especially when the prices an airline receives for a code-shared product depend on who sells the ticket or on the partners' market strength. The operating carrier might accept less code-shared flights sold by the partner than if he received the requests. If the airlines sell code shared products at different prizes (or in case of preferences for certain airlines on the side of the customers) a different type of horizontal competition might arise since the customers could request the same code shared product from the competitor if they are denied once by their preferred airline.

While we assumed the players to have equal market power (or decision power within the alliance), it might be different in certain cases. Here, a Stackelberg game could be used to model the players' different market power or importance within the alliance in future research. Connected to this, an interesting point to be considered is when only the total demand for a product is given, not each competitor's individual demand. Then, the total demand for a product might be split based on the players' booking limits. Li et al. (2007) as well as Li et al. (2008) have proposed such demand splitting rules for single-leg, two-class problems. Their work could serve as a ground to develop such rules for the network case with $\mathcal{F}$ classes. Further, the

alliance problem could be modeled and solved via a multi-objective optimization problem with the different objectives standing for the partners' revenue interests.

Also, considering more than two players is a natural extension, since an airline's network typically interferes with several competitors' networks. Here, the algorithm by Daskalakis and Papadimitriou (2006) to compute pure NE in graphical games might be applicable. In such games there exist several players and the players' payoffs are only affected by the behavior of their "neighbors" and not by all players, which is clearly the case with competitors from different countries. Also, the concept of targeted competition coined just recently by Dubovik and Parakhonyak (2014) should be considered. Here, with at least three players involved, all players must choose the intensity of competing with the other competitors.

The long-term effects of competition should be investigated e.g. in terms of which capacities should be assigned to the individual legs when competition is taken into account leading to tendentially higher total demands. An interesting point to see would be how higher capacities on the aircraft influence the amount of denied demand and the competitors' behavior in the NE. Connected to this is also an investigation of repeated competitive interaction, i.e. repeated games.

The one-shot games considered here can be seen as excerpts from repeated games. In future research, dynamic games and repeated competition should be investigated. As Isler and Imhof (2008) noted, a repeated price competition may lead to a spiral-down effect in which the prices and revenues decrease ever further. A further investigation of their model offers Zimmermann (2014) with a treatment of repeated RM competition in a prisoner's dilemma setting. An interesting question is whether a similar effect can be detected in simultaneous price and quantity competition. This might necessitate solving the more complicated dynamic models and games only heuristically. For dynamic games, re-solving schedules of the DPL for reoptimizing booking limits over time could be used like it has been proposed for individual airlines by e.g. Cooper (2002), Chen and Homem-de-Mello (2010) as well as Jasin and Kumar (2012). Reoptimization usually leads to revenue increases compared to determining the values only once or following a heuristic schedule. In the future one could provide reoptimization schedules in competitive settings and see whether they differ from the non-competitive ones. Furthermore, dynamic games would provide more support for mixed NE since the probabilities could indeed be interpreted as the relative frequency with which the corresponding pure strategies are employed.

From a more technical side, future research could be directed towards providing conditions under which a pure NE exists in games considered in this thesis. Jiang and Pang (2011) provided such conditions for an RM game based on the PNLP but could not do the same for games based on their DLP which was similar to our model in (5.1) – (5.5) which models a generalized game. However, our reformulated model (5.15) – (5.21) might make such proofs easier since it models a "normal" game in which the players' available strategies are always the same. Estimating the number of NE (possibly in relation to the number of strategies available to the players) should provide more insights about the expected running time of the algorithm. Algorithmic game theorists might be interested in the complexity of the algorithms provided here and how they behave compared to the algorithms based on bimatrix games.

As mentioned above, in the exact Algorithm 5.1 the bounds LB^a_{pf} and UB^a_{pf} for the binary variables determine the number of binary variables in the model and affect the solution and computation time. Future research should take into account the stochastic effects of demand and investigate the effects of narrower vs. wider ranges between UB^a_{pf} and LB^a_{pf} in terms of the payoff values and running time. Connected to this is a consideration of incomplete information

where not all information about the competitor is known in the data. Hu et al. (2013), Belobaba and Jain (2013) as well as Wright (2014) have recently proposed approaches in this direction.

A final conclusion that can be drawn from the literature overviews in the previous chapters and the outlook just made is that research is addressing many important fields relevant to the RM practice under competition but still needs to catch up with the practical developments. This thesis presented a further step towards closing the gap between theory and practice by providing several models and algorithms to model and solve several relevant RM problems under competition. Future research will have to find answers to many more questions considering the development in RM under competition, both for individual airlines and for whole alliances. The findings from this thesis can serve as a foundation for extensions aimed at tackling these issues.

Appendix A

Further Computational Results

Here, we present results for the computational studies for the perturbation order "Alphabetic by O&D Name, then Increasing by Fare Class". One can see that the results are similar to those with the prices perturbed in a different order.

A.1 Results for Section 5.3.5.2

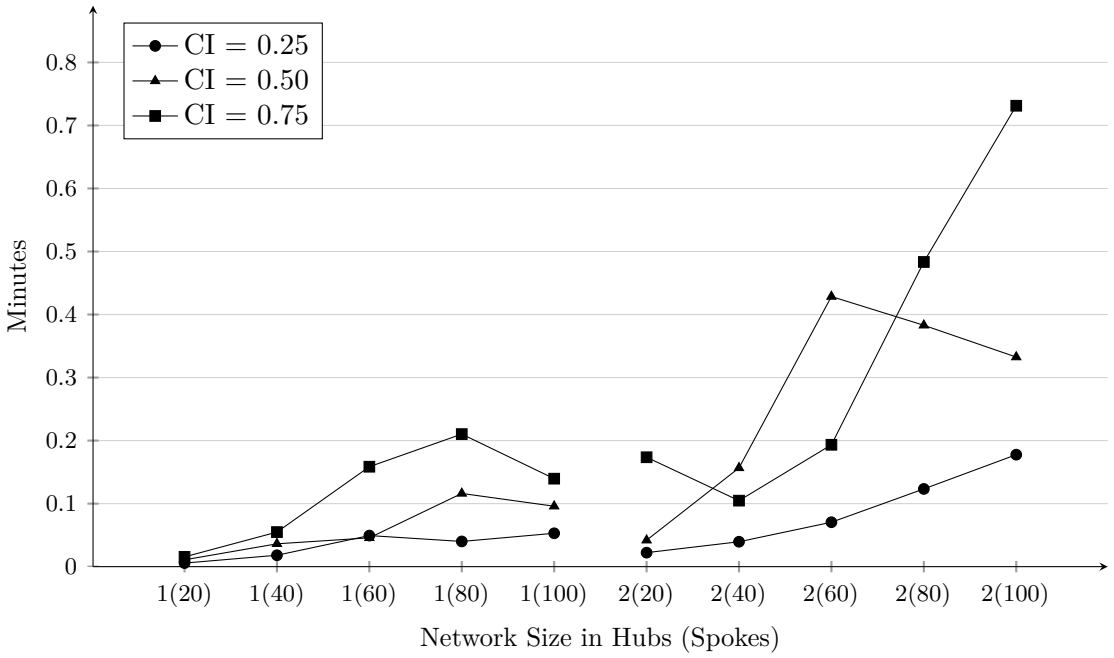


Figure A.1: Average Computation Times in Minutes with $\mu = 2$

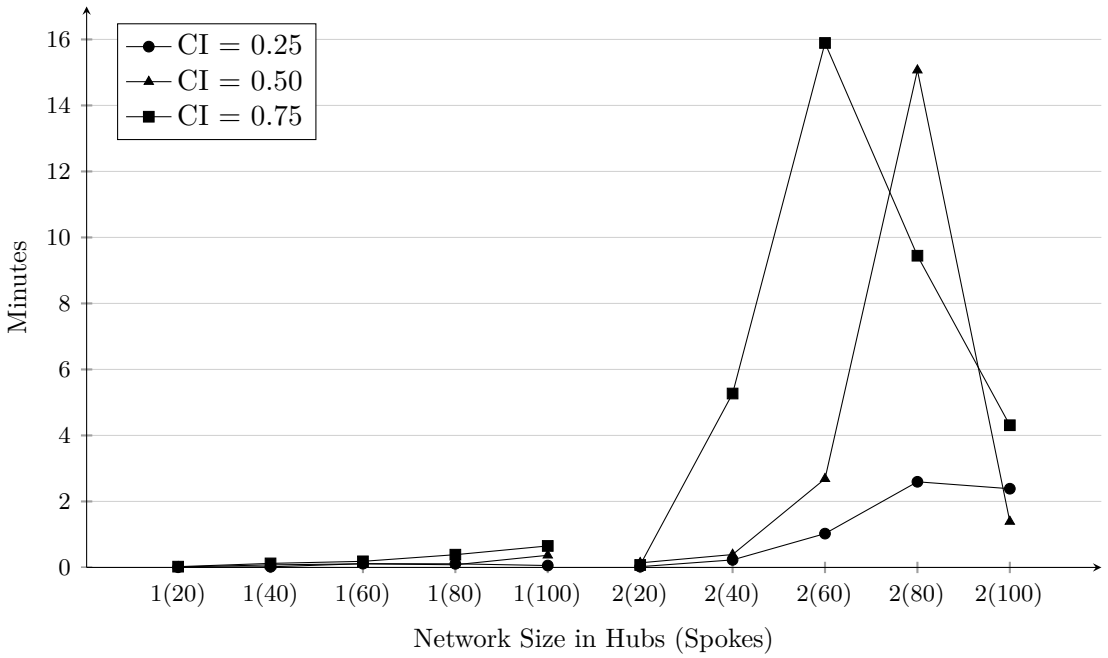


Figure A.2: Average Computation Times in Minutes with $\mu = 4$

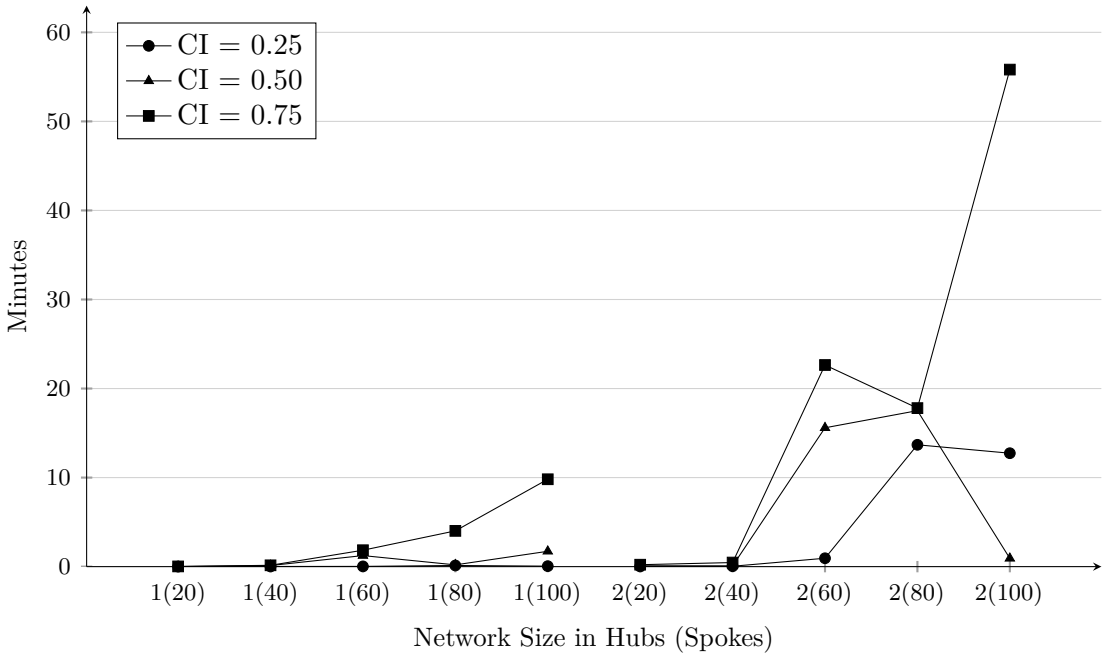


Figure A.3: Average Computation Times in Minutes with $\mu = 6$

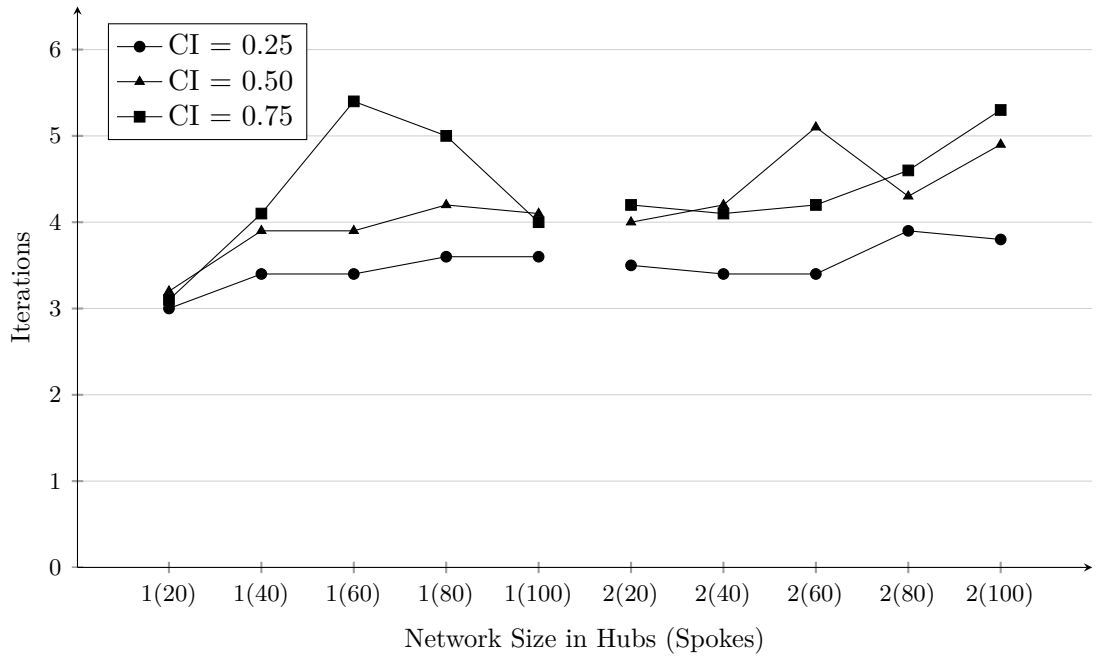


Figure A.4: Average Numbers of Iterations with $\mu = 2$

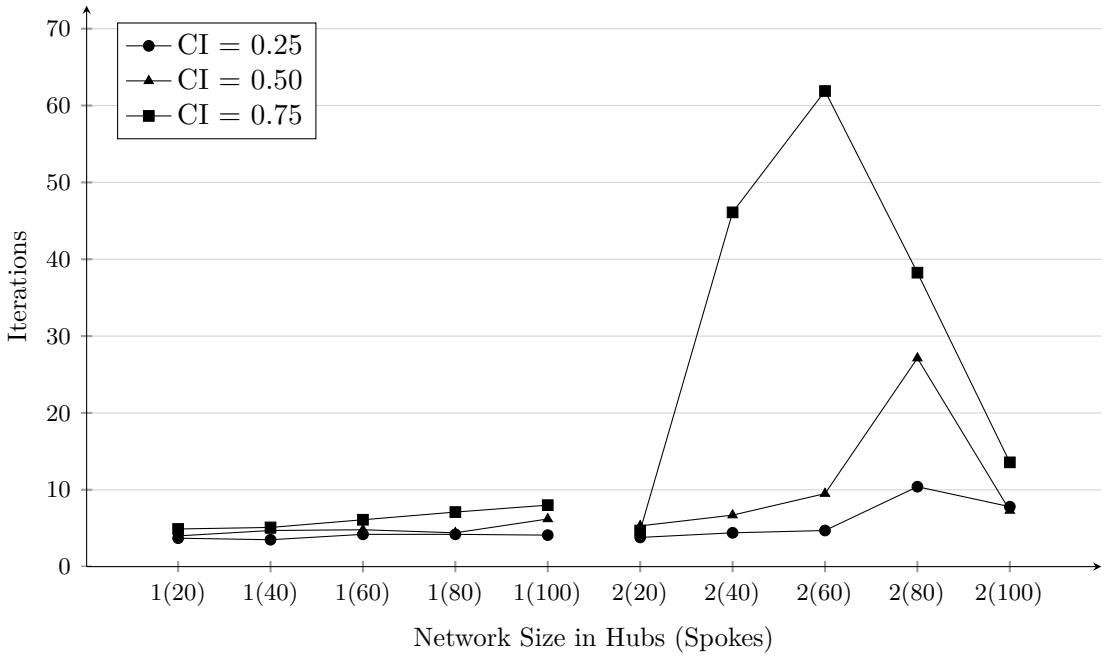


Figure A.5: Average Numbers of Iterations with $\mu = 4$

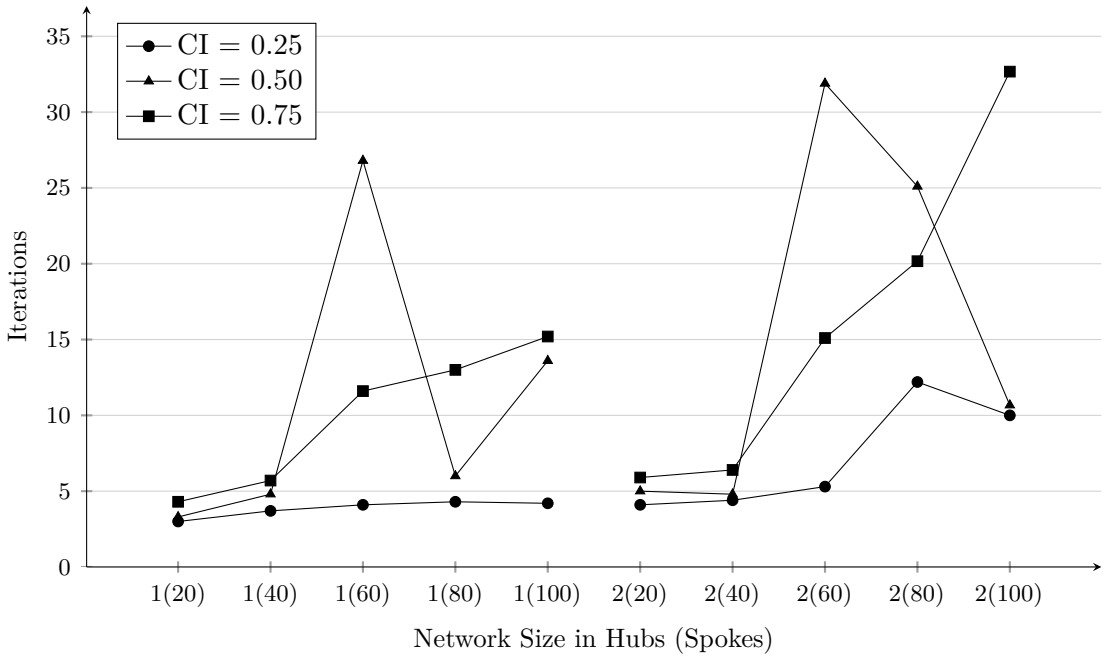


Figure A.6: Average Numbers of Iterations with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	99.91%	99.97%	99.96%	99.97%	99.98%	100.01%	100.04%	100.02%	99.97%	99.95%
	NC/C	99.87%	99.92%	99.92%	99.93%	99.94%	99.94%	99.99%	99.97%	99.93%	99.91%
	NC/NE	99.97%	99.95%	99.96%	99.96%	99.96%	99.93%	99.95%	99.96%	99.96%	99.96%
0.50	NE/C	99.92%	100.00%	100.03%	100.04%	100.06%	100.20%	100.16%	100.09%	100.04%	100.01%
	NC/C	99.85%	99.91%	99.95%	99.97%	99.98%	100.06%	100.06%	100.00%	99.96%	99.93%
	NC/NE	99.93%	99.91%	99.92%	99.92%	99.92%	99.86%	99.90%	99.92%	99.92%	99.92%
0.75	NE/C	99.92%	100.03%	100.03%	100.07%	100.10%	100.31%	100.27%	100.15%	100.12%	100.06%
	NC/C	99.84%	99.93%	99.93%	99.96%	100.00%	100.14%	100.15%	100.05%	100.03%	99.97%
	NC/NE	99.92%	99.90%	99.90%	99.90%	99.90%	99.83%	99.88%	99.90%	99.91%	99.91%

Table A.1: Average Payoff Ratios with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.12%	100.13%	100.12%	100.09%	100.12%	100.14%	100.05%	100.01%	99.97%	99.99%
	NC/C	100.02%	100.06%	100.04%	100.02%	100.04%	100.05%	99.98%	99.95%	99.92%	99.93%
	NC/NE	99.90%	99.93%	99.92%	99.92%	99.92%	99.91%	99.93%	99.94%	99.94%	99.95%
0.50	NE/C	100.28%	100.35%	100.32%	100.31%	100.31%	100.42%	100.18%	100.14%	100.09%	100.10%
	NC/C	100.09%	100.20%	100.16%	100.15%	100.15%	100.25%	100.05%	100.03%	99.98%	99.99%
	NC/NE	99.82%	99.85%	99.84%	99.84%	99.84%	99.83%	99.87%	99.89%	99.89%	99.90%
0.75	NE/C	100.32%	100.45%	100.40%	100.44%	100.44%	100.54%	100.28%	100.21%	100.16%	100.14%
	NC/C	100.09%	100.26%	100.19%	100.21%	100.21%	100.32%	100.09%	100.06%	100.01%	99.99%
	NC/NE	99.77%	99.81%	99.79%	99.77%	99.77%	99.78%	99.82%	99.84%	99.84%	99.85%

Table A.2: Average Payoff Ratios with $\mu = 4$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.22%	100.21%	100.23%	100.17%	100.22%	100.11%	100.10%	100.09%	100.08%	100.06%
	NC/C	100.14%	100.08%	100.10%	100.06%	100.09%	100.02%	100.02%	100.01%	99.99%	99.98%
	NC/NE	99.91%	99.87%	99.87%	99.89%	99.88%	99.91%	99.92%	99.92%	99.91%	99.92%
0.50	NE/C	100.45%	100.53%	100.54%	100.48%	100.54%	100.38%	100.31%	100.25%	100.24%	100.22%
	NC/C	100.30%	100.28%	100.28%	100.24%	100.29%	100.21%	100.16%	100.10%	100.07%	100.07%
	NC/NE	99.84%	99.75%	99.74%	99.76%	99.75%	99.83%	99.85%	99.85%	99.83%	99.84%
0.75	NE/C	100.53%	100.83%	100.81%	100.73%	100.82%	100.54%	100.46%	100.36%	100.36%	100.34%
	NC/C	100.34%	100.43%	100.42%	100.35%	100.43%	100.32%	100.25%	100.15%	100.13%	100.11%
	NC/NE	99.81%	99.60%	99.61%	99.63%	99.61%	99.78%	99.79%	99.78%	99.77%	99.77%

Table A.3: Average Payoff Ratios with $\mu = 6$

A.2 Results for Section 5.4.6.1

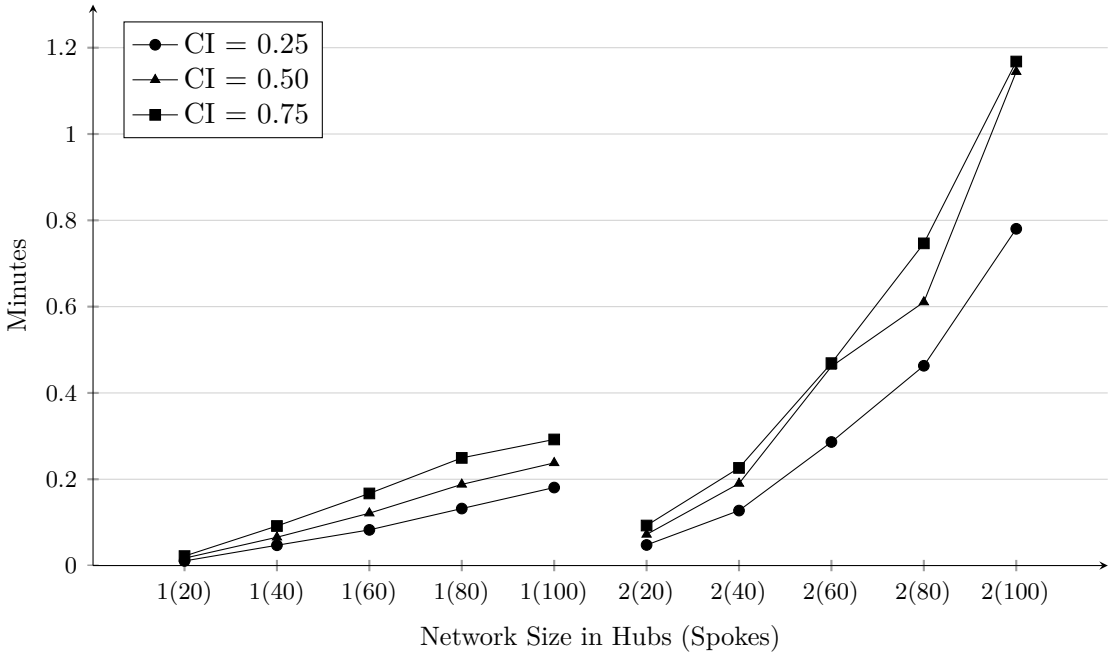


Figure A.7: Average Computation Times in Minutes with $\mu = 2$

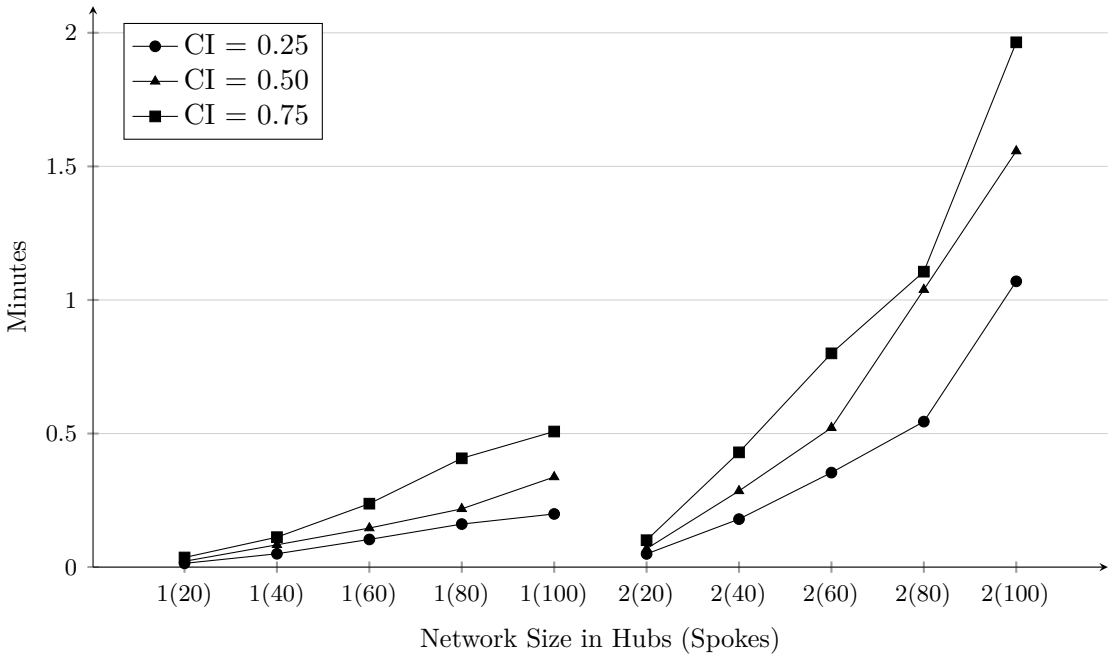


Figure A.8: Average Computation Times in Minutes with $\mu = 4$

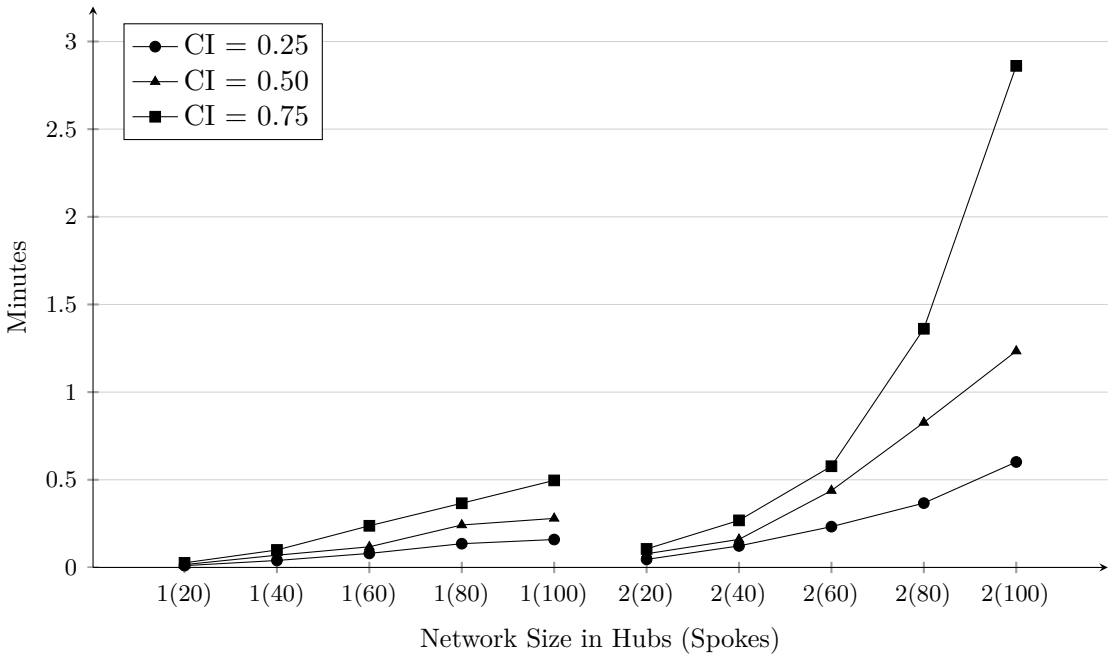


Figure A.9: Average Computation Times in Minutes with $\mu = 6$

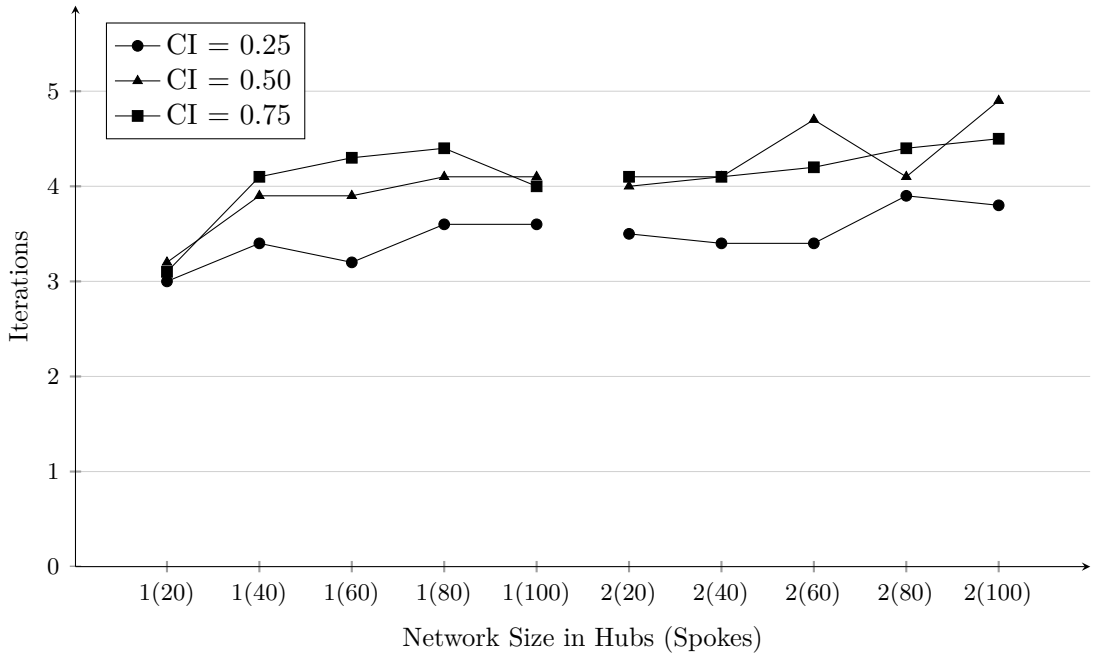


Figure A.10: Average Numbers of Iterations with $\mu = 2$

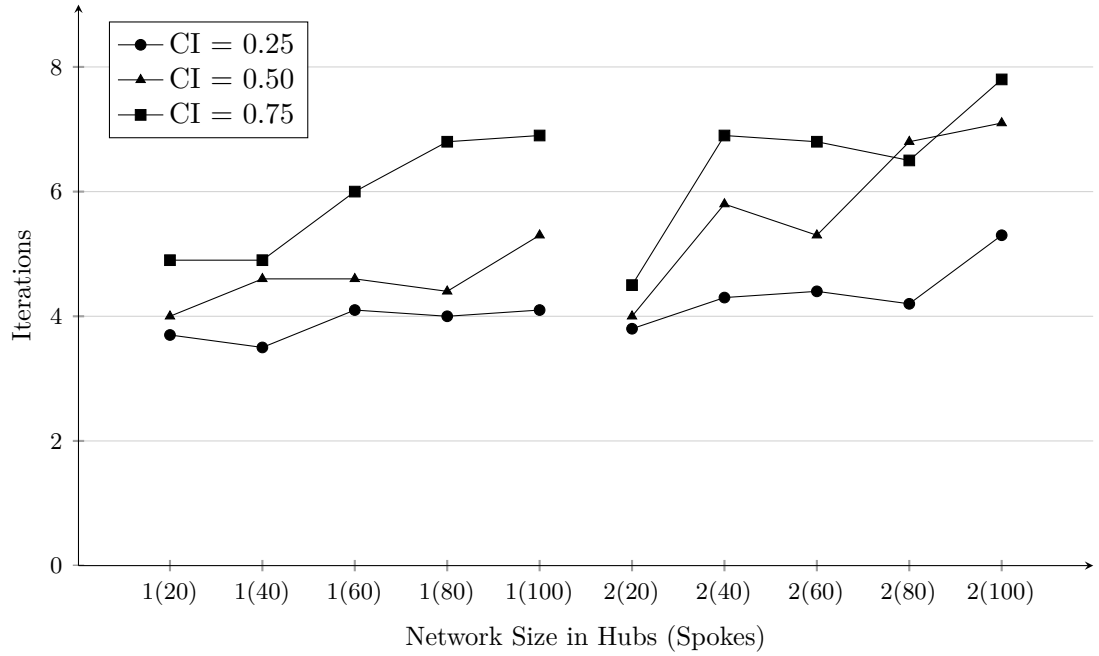


Figure A.11: Average Numbers of Iterations with $\mu = 4$

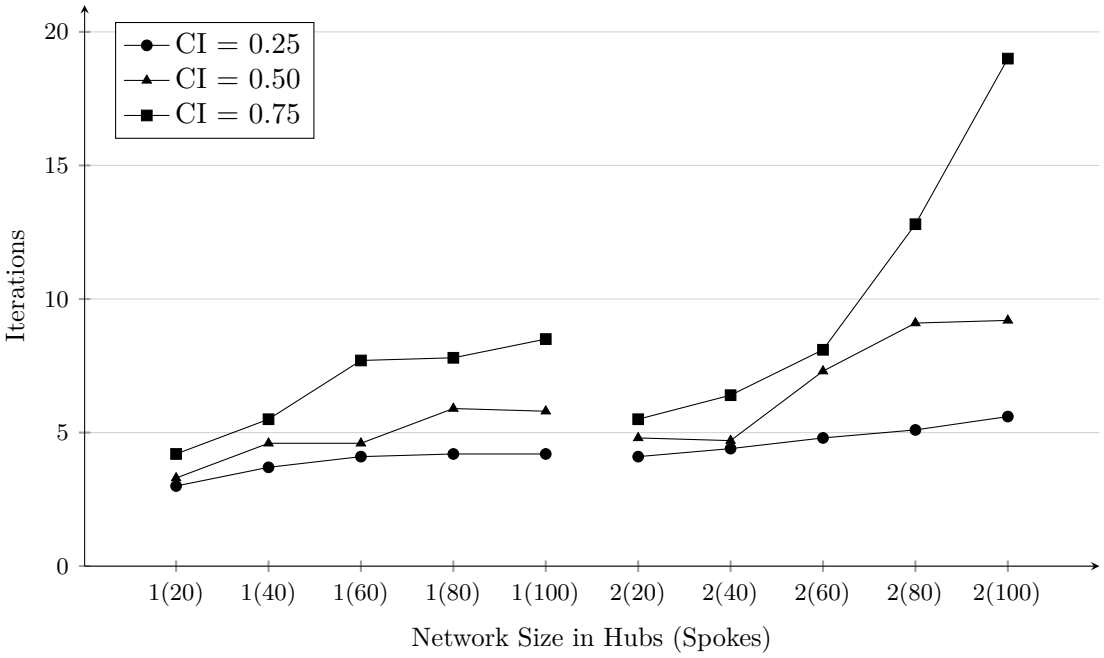


Figure A.12: Average Numbers of Iterations with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	99.91%	99.97%	99.96%	99.97%	99.98%	100.01%	100.04%	100.02%	99.97%	99.95%
	NC/C	99.87%	99.92%	99.92%	99.93%	99.94%	99.94%	99.99%	99.97%	99.93%	99.91%
	NC/NE	99.97%	99.95%	99.96%	99.96%	99.96%	99.93%	99.95%	99.96%	99.96%	99.96%
0.50	NE/C	99.92%	100.00%	100.03%	100.04%	100.06%	100.20%	100.16%	100.09%	100.04%	100.01%
	NC/C	99.85%	99.91%	99.95%	99.97%	99.98%	100.06%	100.06%	100.00%	99.96%	99.93%
	NC/NE	99.93%	99.91%	99.92%	99.92%	99.92%	99.86%	99.90%	99.92%	99.92%	99.92%
0.75	NE/C	99.92%	100.03%	100.03%	100.07%	100.10%	100.31%	100.27%	100.15%	100.12%	100.06%
	NC/C	99.84%	99.93%	99.93%	99.96%	100.00%	100.14%	100.15%	100.05%	100.03%	99.97%
	NC/NE	99.92%	99.90%	99.90%	99.90%	99.90%	99.83%	99.88%	99.90%	99.91%	99.91%

Table A.4: Average Payoff Ratios with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.12%	100.13%	100.12%	100.09%	100.12%	100.14%	100.05%	100.01%	99.97%	99.99%
	NC/C	100.02%	100.06%	100.04%	100.02%	100.04%	100.05%	99.98%	99.95%	99.92%	99.93%
	NC/NE	99.90%	99.93%	99.92%	99.92%	99.92%	99.91%	99.93%	99.94%	99.94%	99.95%
0.50	NE/C	100.28%	100.35%	100.32%	100.31%	100.31%	100.42%	100.18%	100.14%	100.09%	100.10%
	NC/C	100.09%	100.20%	100.16%	100.15%	100.15%	100.24%	100.05%	100.03%	99.98%	99.99%
	NC/NE	99.82%	99.85%	99.84%	99.84%	99.84%	99.83%	99.87%	99.89%	99.89%	99.90%
0.75	NE/C	100.32%	100.45%	100.40%	100.44%	100.44%	100.54%	100.28%	100.21%	100.16%	100.15%
	NC/C	100.09%	100.26%	100.19%	100.21%	100.21%	100.32%	100.09%	100.06%	100.00%	100.00%
	NC/NE	99.77%	99.81%	99.79%	99.78%	99.77%	99.78%	99.82%	99.84%	99.85%	99.85%

Table A.5: Average Payoff Ratios with $\mu = 4$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	100.22%	100.21%	100.23%	100.17%	100.22%	100.11%	100.10%	100.09%	100.08%	100.06%
	NC/C	100.14%	100.08%	100.10%	100.06%	100.09%	100.02%	100.02%	100.01%	99.99%	99.98%
	NC/NE	99.91%	99.87%	99.87%	99.89%	99.88%	99.91%	99.92%	99.92%	99.91%	99.92%
0.50	NE/C	100.45%	100.53%	100.54%	100.48%	100.54%	100.38%	100.31%	100.25%	100.24%	100.22%
	NC/C	100.30%	100.28%	100.28%	100.24%	100.29%	100.21%	100.16%	100.10%	100.07%	100.06%
	NC/NE	99.84%	99.75%	99.74%	99.76%	99.75%	99.83%	99.85%	99.85%	99.83%	99.84%
0.75	NE/C	100.53%	100.83%	100.81%	100.72%	100.82%	100.54%	100.46%	100.36%	100.36%	100.33%
	NC/C	100.34%	100.43%	100.42%	100.35%	100.43%	100.32%	100.25%	100.15%	100.14%	100.11%
	NC/NE	99.81%	99.60%	99.61%	99.63%	99.61%	99.78%	99.79%	99.78%	99.77%	99.78%

Table A.6: Average Payoff Ratios with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Heuristic/Exact	–	–	100.0029%	–	–	–	–	–	–	–
	ϵ	–	–	0.0027	–	–	–	–	–	–	–
0.50	Heuristic/Exact	–	–	–	99.9964%	–	–	99.9978%	100.0035%	100.0029%	–
	ϵ	–	–	–	0.0014	–	–	0.0019	0.0015	0.0015	–
0.75	Heuristic/Exact	–	–	99.9986%	100.0022%	–	99.9985%	–	–	99.9994%	100.0006%
	ϵ	–	–	0.0014	0.0021	–	0.0032	–	–	0.0017	0.0018

Table A.7: Average Payoff Ratios between the Heuristic and the Exact Algorithm with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Heuristic/Exact	–	–	99.9994%	100.0011%	–	–	100.0034%	100.0021%	100.0078%	100.0042%
	ϵ	–	–	0.0016	0.0015	–	–	0.0015	0.0012	0.0013	0.0010
0.50	Heuristic/Exact	–	99.9969%	100.0008%	–	100.0009%	–	100.0012%	100.0021%	100.0022%	99.9983%
	ϵ	–	0.0035	0.0030	–	0.0033	–	0.0026	0.0023	0.0021	0.0020
0.75	Heuristic/Exact	–	99.9974%	99.9995%	99.9978%	99.9986%	–	99.9947%	99.9984%	100.0014%	100.0004%
	ϵ	–	0.0038	0.0040	0.0045	0.0046	–	0.0035	0.0031	0.0031	0.0030

Table A.8: Average Payoff Ratios between the Heuristic and the Exact Algorithm with $\mu = 4$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Heuristic/Exact	–	–	–	99.9953%	–	–	–	100.0004%	100.0007%	100.0009%
	ϵ	–	–	–	0.0020	–	–	–	0.0016	0.0016	0.0015
0.50	Heuristic/Exact	–	99.9991%	100.0038%	100.0005%	99.9996%	99.9994%	–	99.9970%	99.9984%	–
	ϵ	–	0.0049	0.0053	0.0048	0.0049	0.0035	–	0.0029	0.0031	–
0.75	Heuristic/Exact	99.9941%	99.9981%	99.9945%	99.9960%	100.0013%	99.9988%	99.9966%	99.9999%	99.9929%	99.9968%
	ϵ	0.0034	0.0086	0.0078	0.0070	0.0077	0.0044	0.0042	0.0044	0.0044	0.0045

Table A.9: Average Payoff Ratios between the Heuristic and the Exact Algorithm with $\mu = 6$

A.3 Results for Section 6.4.2

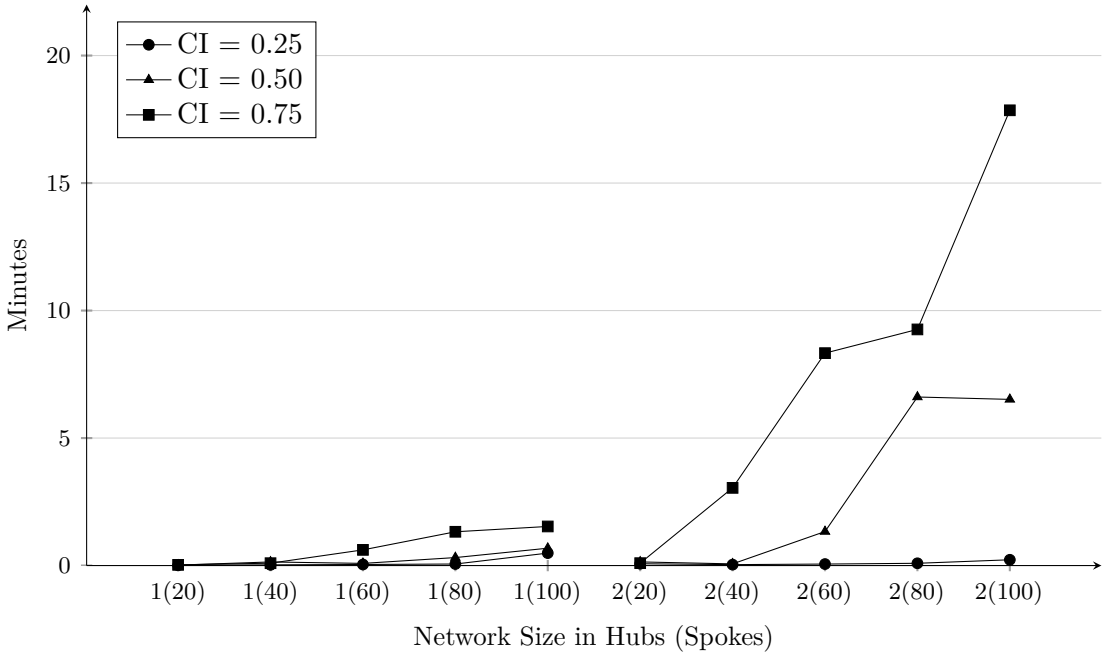


Figure A.13: Average Computation Times in Minutes with Competitive Prices

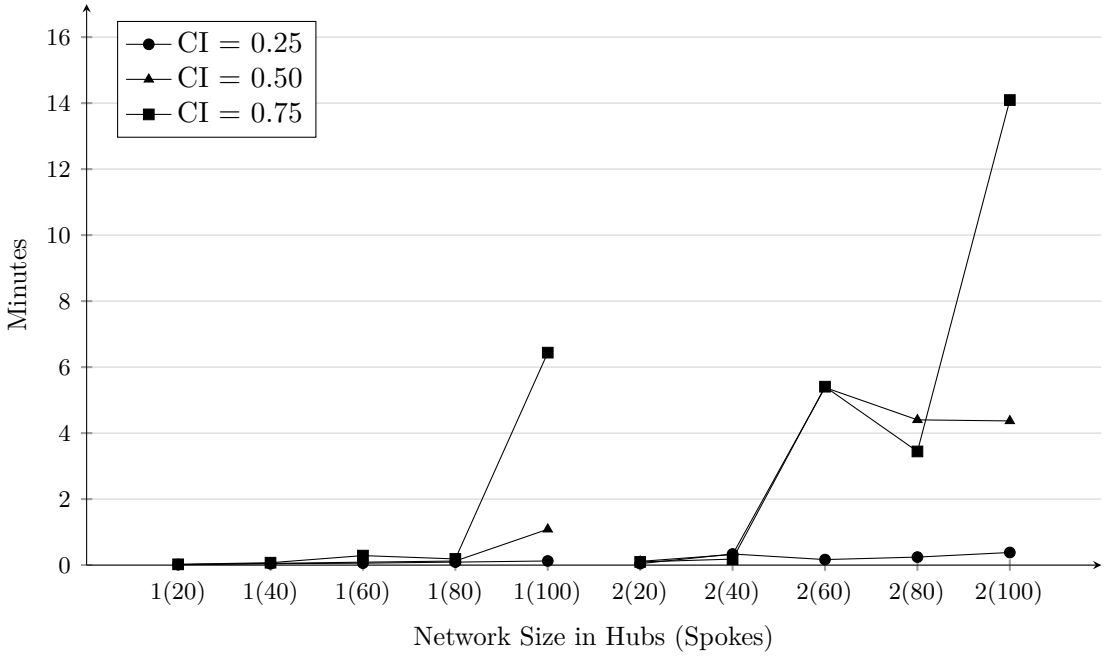


Figure A.14: Average Computation Times in Minutes with Monopolistic Prices

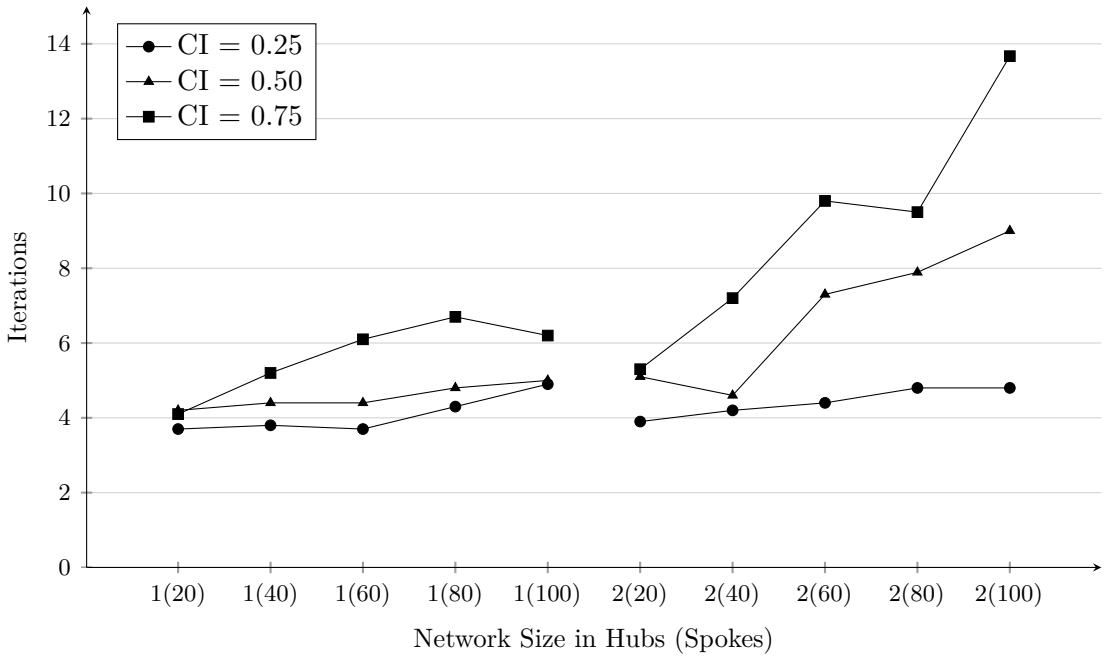


Figure A.15: Average Numbers of Iterations with Competitive Prices

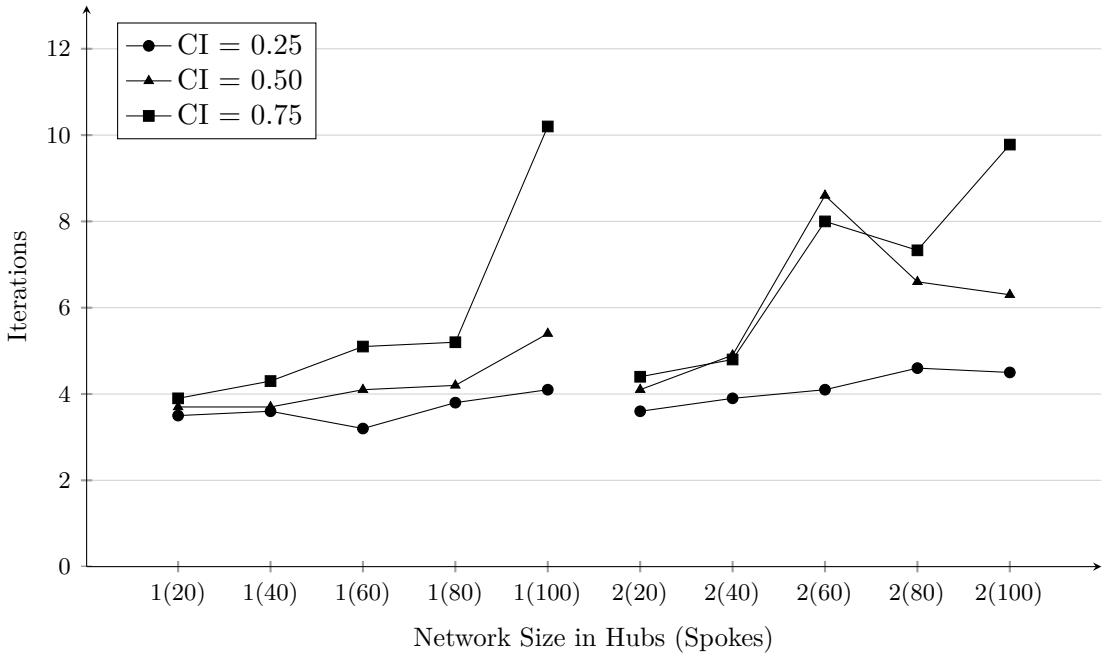


Figure A.16: Average Numbers of Iterations with Monopolistic Prices

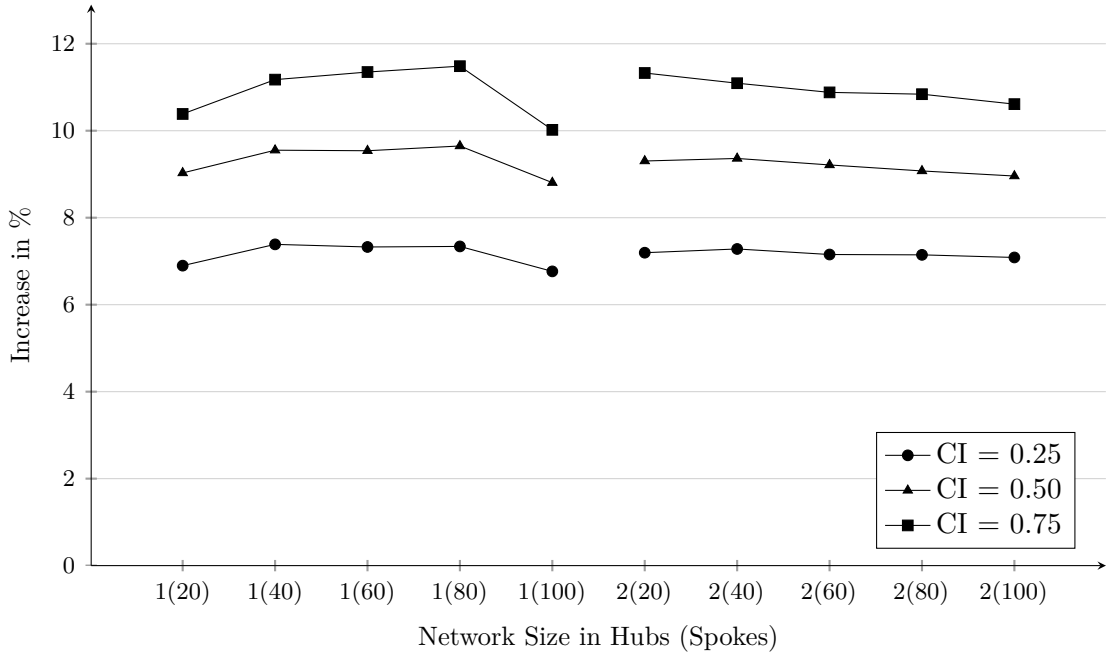


Figure A.17: Average Payoff Increases From Step 2 to Step 3 with Competitive Prices for Player 1

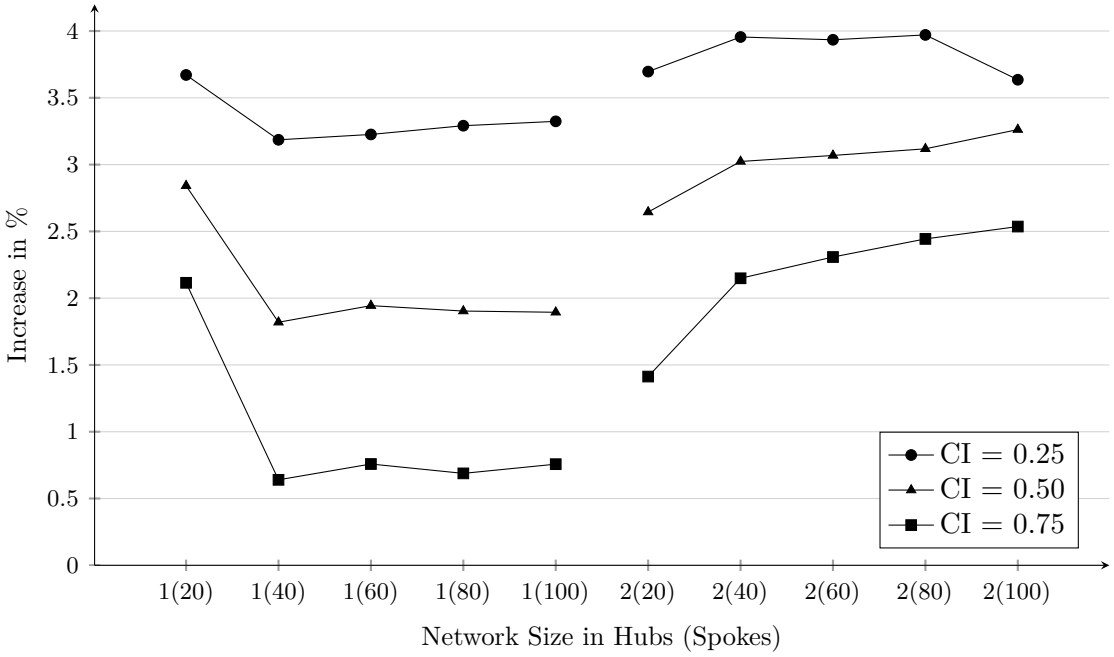


Figure A.18: Average Payoff Increases From Step 2 to Step 3 with Monopolistic Prices for Player 1

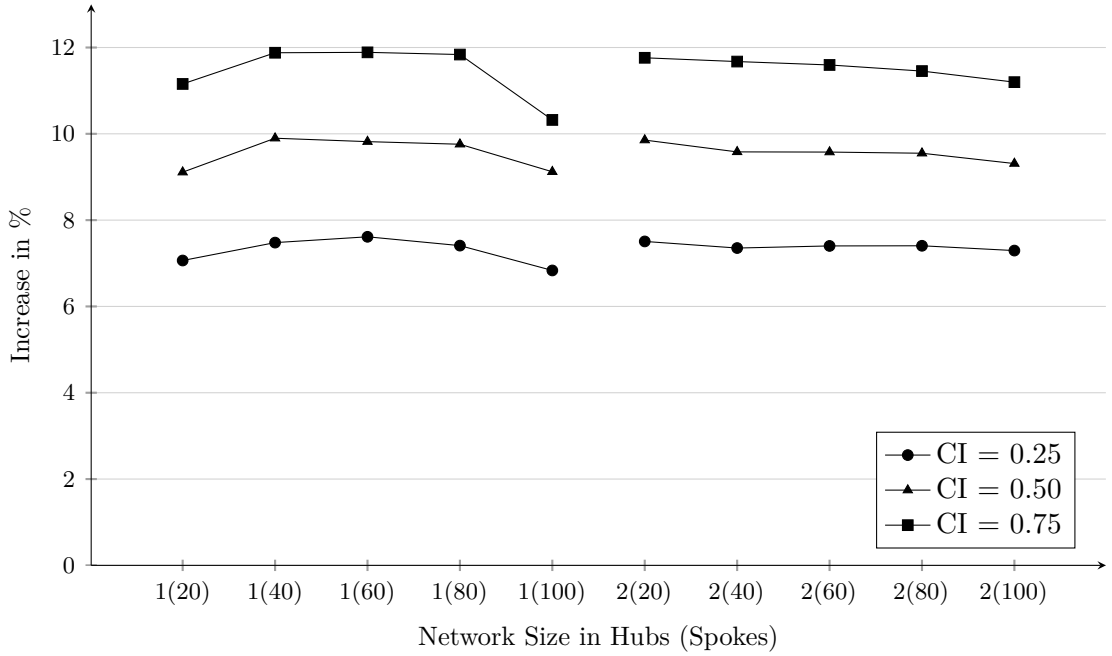


Figure A.19: Average Payoff Increases From Step 2 to Step 3 with Competitive Prices for Player 2

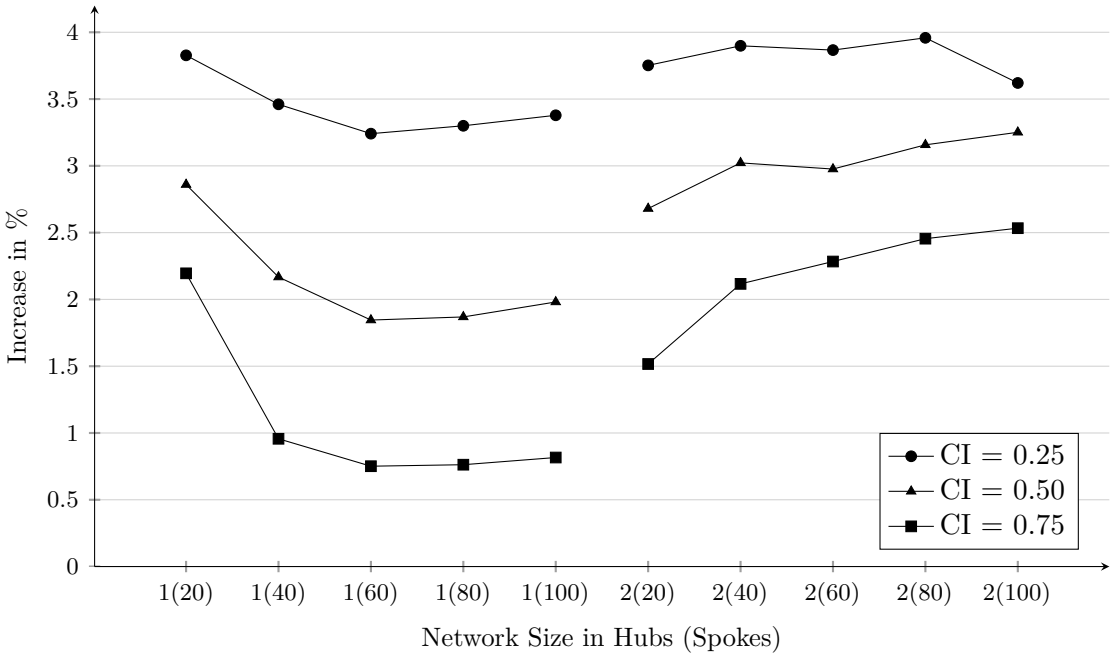


Figure A.20: Average Payoff Increases From Step 2 to Step 3 with Monopolistic Prices for Player 2

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Step 2/Step 3	93.47%	93.08%	93.05%	93.13%	93.21%	93.15%	93.18%	93.22%	93.22%	93.29%
	NC/Step 3	93.17%	92.76%	92.70%	92.81%	92.91%	92.82%	92.89%	92.91%	92.91%	92.97%
	NC/Step 2	99.68%	99.65%	99.62%	99.65%	99.68%	99.65%	99.68%	99.67%	99.67%	99.66%
0.50	Step 2/Step 3	91.69%	91.14%	91.17%	91.15%	91.46%	91.26%	91.35%	91.41%	91.48%	91.63%
	NC/Step 3	91.36%	90.84%	90.86%	90.84%	91.19%	90.97%	91.06%	91.12%	91.18%	91.36%
	NC/Step 2	99.64%	99.67%	99.65%	99.66%	99.70%	99.68%	99.68%	99.68%	99.68%	99.71%
0.75	Step 2/Step 3	90.28%	89.66%	89.59%	89.56%	90.33%	89.65%	89.78%	89.90%	89.97%	90.17%
	NC/Step 3	89.88%	89.28%	89.22%	89.18%	90.02%	89.23%	89.37%	89.48%	89.56%	89.55%
	NC/Step 2	99.56%	99.57%	99.59%	99.57%	99.65%	99.53%	99.54%	99.53%	99.54%	99.32%

Table A.10: Average Payoff Ratios with Competitive Prices

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	Step 2/Step 3	96.39%	96.78%	96.87%	96.81%	96.76%	96.41%	96.22%	96.25%	96.19%	96.50%
	NC/Step 3	96.24%	96.57%	96.64%	96.60%	96.55%	96.26%	96.10%	96.13%	96.05%	96.00%
	NC/Step 2	99.85%	99.78%	99.77%	99.79%	99.79%	99.84%	99.88%	99.88%	99.86%	99.50%
0.50	Step 2/Step 3	97.23%	98.04%	98.14%	98.15%	98.10%	97.41%	97.07%	97.07%	96.96%	96.85%
	NC/Step 3	97.07%	97.84%	97.94%	97.93%	97.88%	97.28%	96.94%	96.94%	96.83%	96.72%
	NC/Step 2	99.84%	99.79%	99.79%	99.78%	99.78%	99.87%	99.87%	99.87%	99.87%	99.87%
0.75	Step 2/Step 3	97.89%	99.21%	99.25%	99.28%	99.22%	98.56%	97.91%	97.76%	97.61%	97.53%
	NC/Step 3	97.77%	99.01%	99.05%	99.07%	99.02%	98.26%	97.78%	97.65%	97.51%	97.42%
	NC/Step 2	99.88%	99.80%	99.79%	99.79%	99.80%	99.70%	99.87%	99.89%	99.90%	99.89%

Table A.11: Average Payoff Ratios with Monopolistic Prices

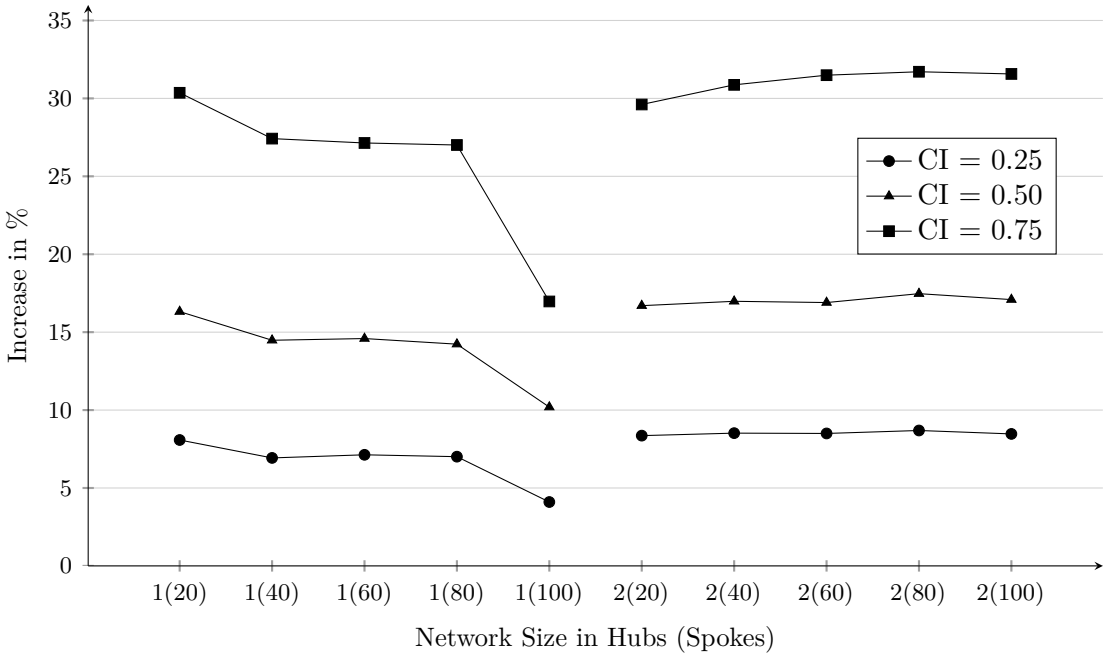


Figure A.21: Average Payoff Increases in Step 2 with Competitive vs. Monopolistic Prices for Player 1

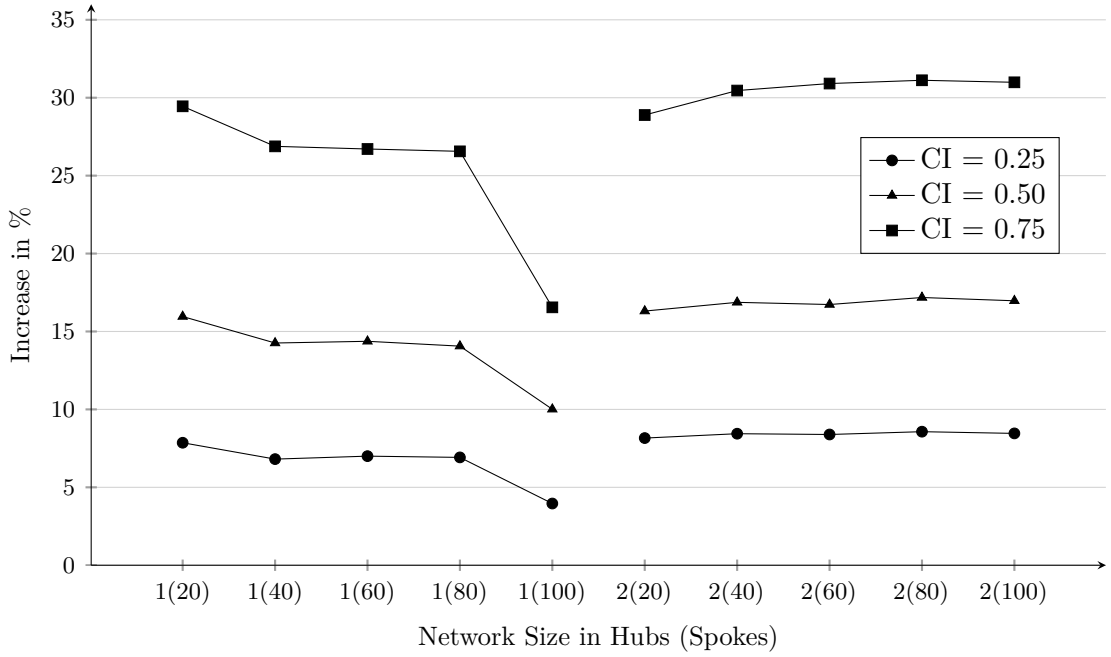


Figure A.22: Average Payoff Increases in Step 2 with Competitive vs. Monopolistic Prices for Player 2

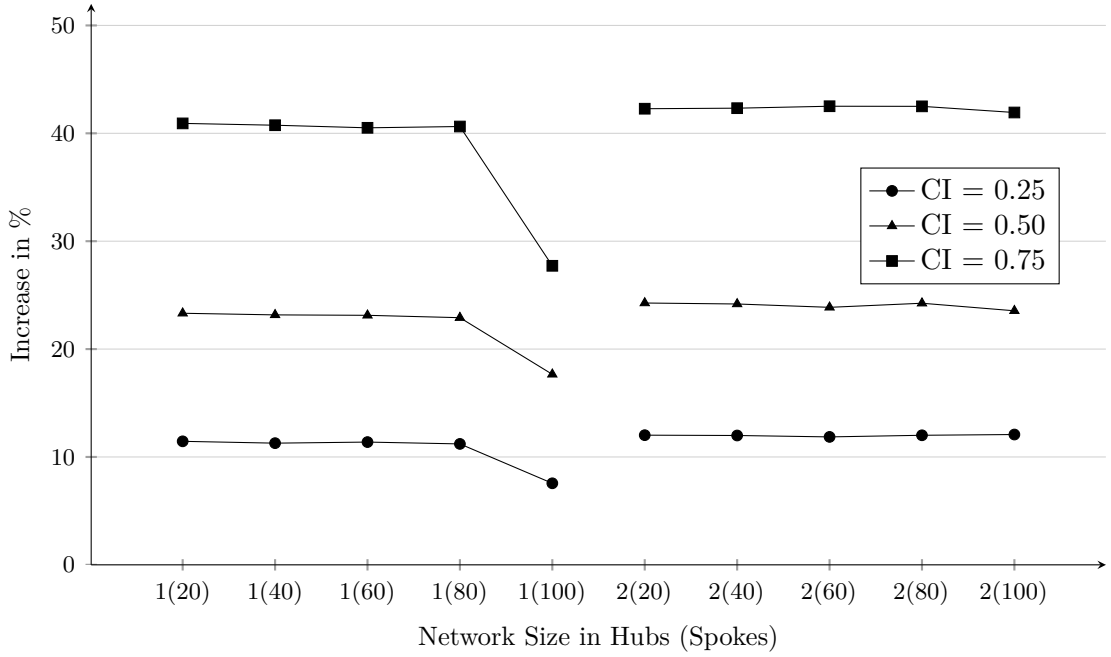


Figure A.23: Average Payoff Increases in Step 3 with Competitive vs. Monopolistic Prices for Player 1

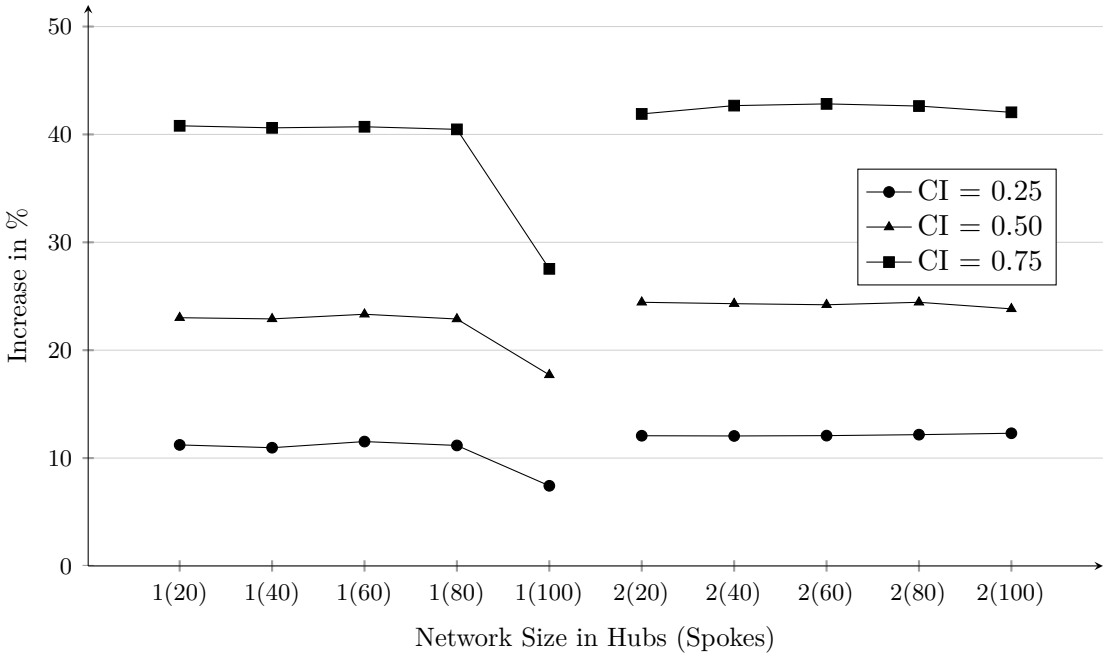


Figure A.24: Average Payoff Increases in Step 3 with Competitive vs. Monopolistic Prices for Player 2

A.4 Results for Section 7.3.2

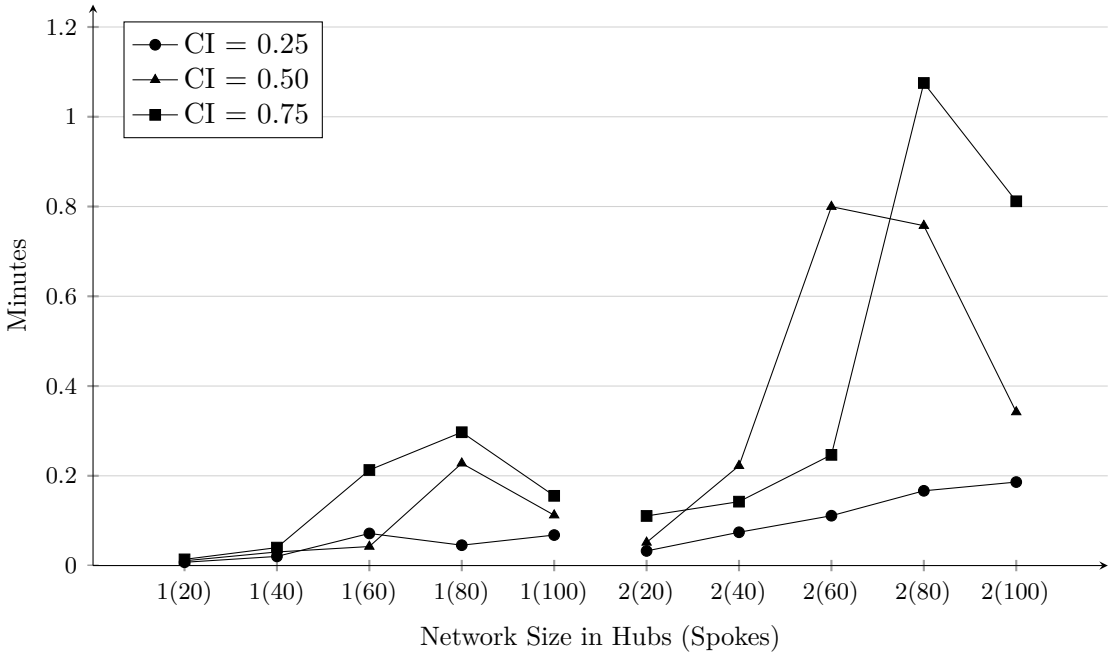


Figure A.25: Average Computation Times in Minutes with $\mu = 2$

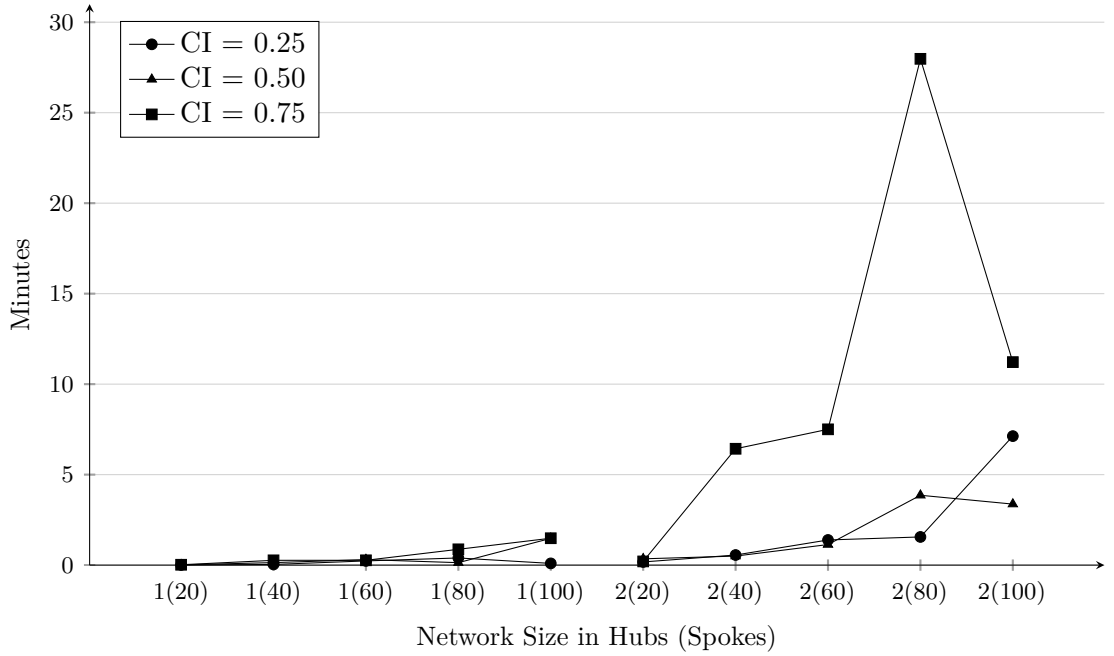


Figure A.26: Average Computation Times in Minutes with $\mu = 4$

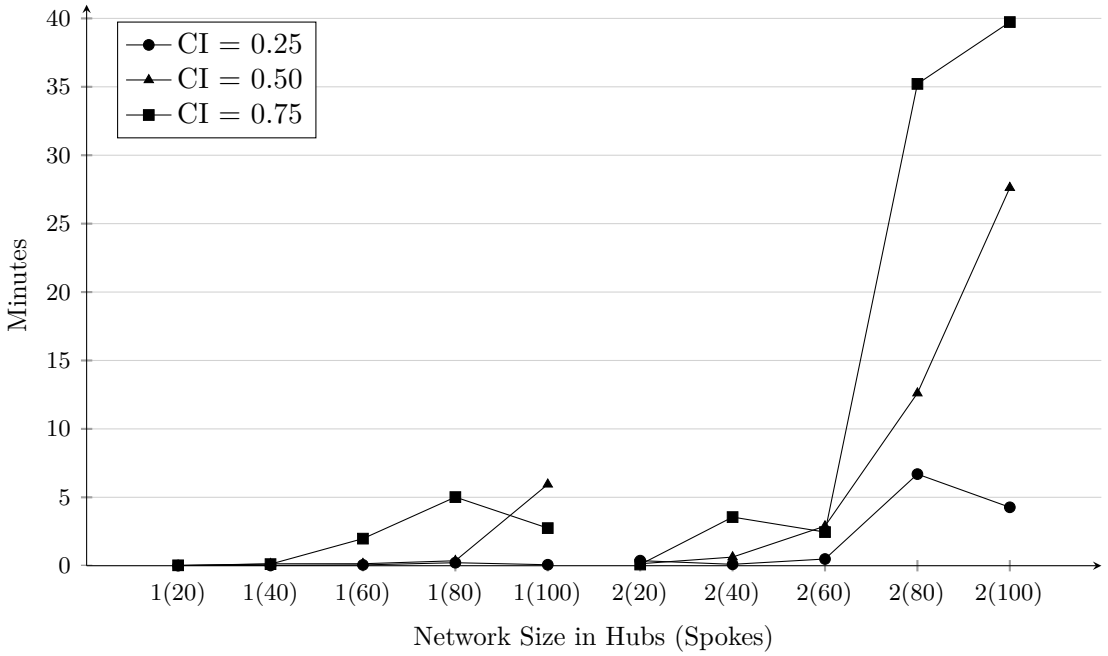


Figure A.27: Average Computation Times in Minutes with $\mu = 6$

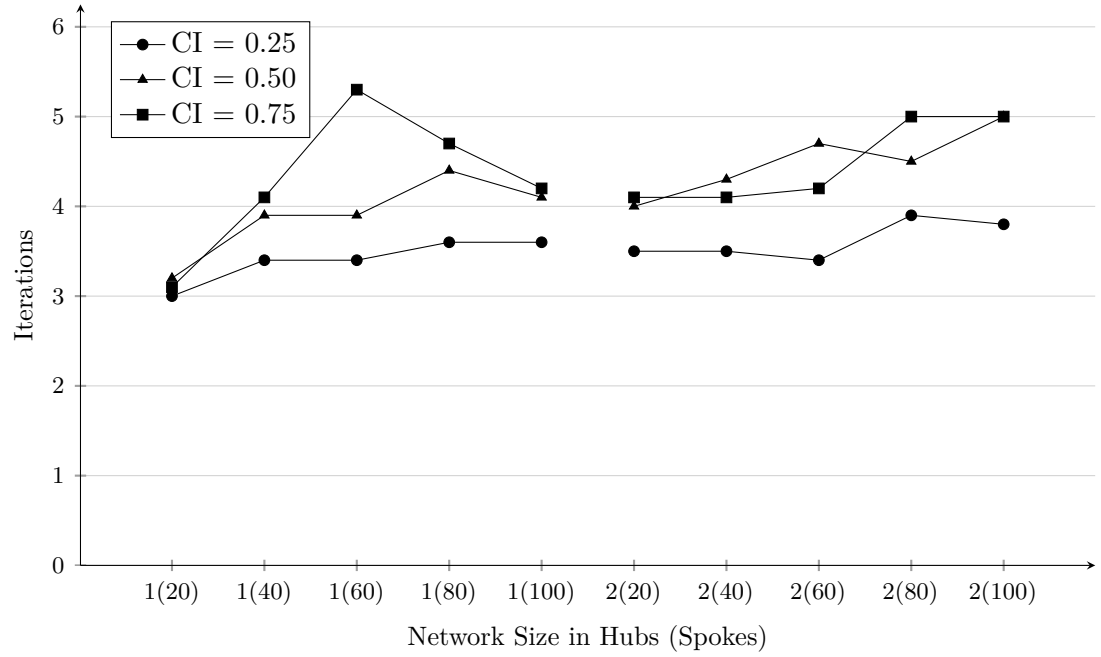


Figure A.28: Average Numbers of Iterations with $\mu = 2$

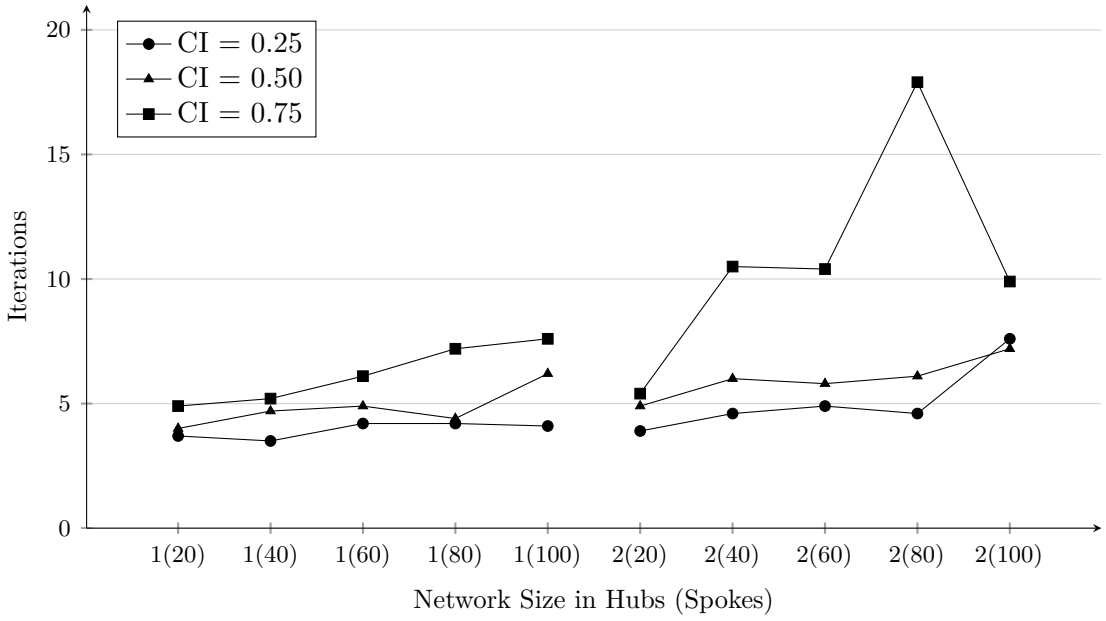


Figure A.29: Average Numbers of Iterations with $\mu = 4$

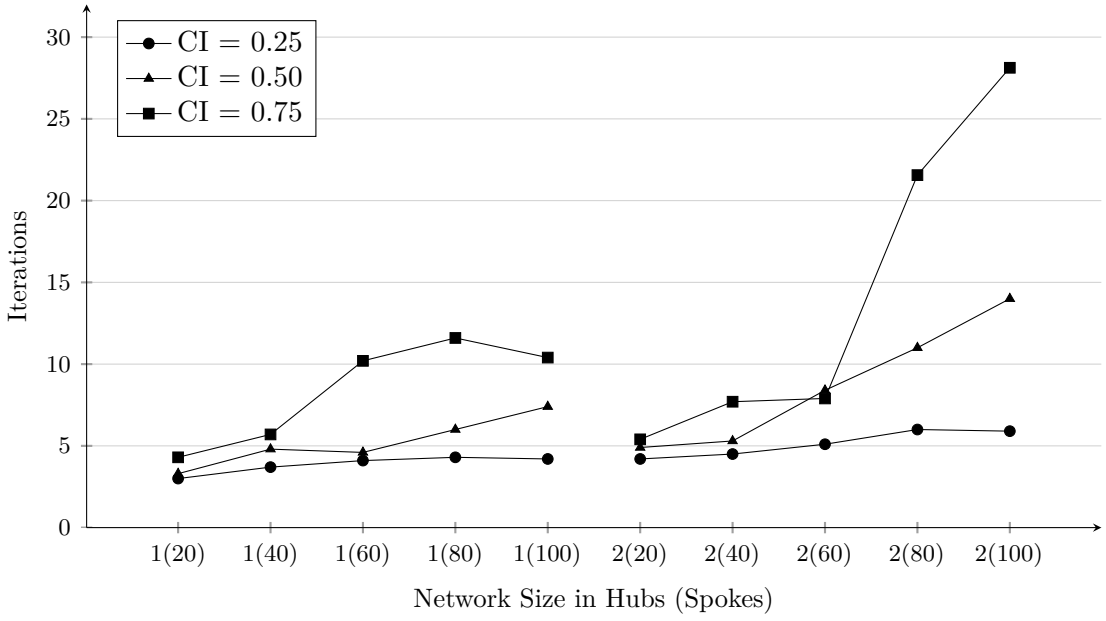


Figure A.30: Average Numbers of Iterations with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	91.44%	91.66%	95.39%	95.07%	97.10%	97.46%	97.95%	99.33%	99.37%	98.92%
	NC/C	81.27%	91.04%	95.13%	94.16%	96.75%	87.27%	92.23%	96.22%	96.08%	96.91%
	NC/NE	88.87%	99.33%	99.72%	99.04%	99.64%	89.55%	94.17%	96.87%	96.69%	97.96%
0.50	NE/C	94.56%	92.51%	96.02%	95.56%	97.54%	97.97%	98.32%	99.58%	99.47%	99.00%
	NC/C	83.57%	91.91%	95.75%	94.59%	97.12%	88.32%	92.24%	96.22%	96.00%	96.91%
	NC/NE	88.38%	99.35%	99.72%	98.99%	99.57%	90.15%	93.82%	96.63%	96.52%	97.89%
0.75	NE/C	96.37%	95.73%	98.95%	97.94%	99.21%	99.19%	99.66%	99.69%	99.73%	99.59%
	NC/C	91.00%	95.31%	98.83%	97.32%	99.04%	93.78%	97.44%	97.67%	97.87%	98.54%
	NC/NE	94.42%	99.56%	99.88%	99.36%	99.83%	94.55%	97.77%	97.98%	98.13%	98.95%

Table A.12: Average Payoff Ratios with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	94.67%	97.88%	98.94%	98.50%	99.33%	95.93%	97.88%	98.95%	98.83%	99.26%
	NC/C	94.58%	97.82%	98.87%	98.42%	99.25%	95.82%	97.80%	98.85%	98.68%	99.20%
	NC/NE	99.90%	99.93%	99.92%	99.92%	99.92%	99.89%	99.92%	99.90%	99.85%	99.94%
0.50	NE/C	95.44%	98.36%	99.29%	98.85%	99.61%	96.67%	97.99%	99.08%	98.92%	99.37%
	NC/C	95.27%	98.21%	99.13%	98.69%	99.46%	96.49%	97.86%	98.91%	98.73%	99.25%
	NC/NE	99.82%	99.85%	99.84%	99.84%	99.84%	99.81%	99.87%	99.83%	99.81%	99.87%
0.75	NE/C	97.93%	99.35%	100.15%	99.74%	100.21%	98.55%	99.60%	99.58%	99.53%	99.79%
	NC/C	97.71%	99.16%	99.94%	99.52%	99.98%	98.35%	99.43%	99.39%	99.35%	99.63%
	NC/NE	99.77%	99.81%	99.79%	99.78%	99.77%	99.80%	99.82%	99.81%	99.82%	99.84%

Table A.13: Average Payoff Ratios with $\mu = 4$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	NE/C	97.48%	99.20%	99.74%	99.39%	99.88%	98.17%	99.17%	99.67%	99.54%	99.74%
	NC/C	97.40%	99.08%	99.61%	99.28%	99.76%	98.10%	99.10%	99.58%	99.44%	99.66%
	NC/NE	99.91%	99.87%	99.87%	99.89%	99.88%	99.93%	99.93%	99.91%	99.90%	99.92%
0.50	NE/C	98.06%	99.63%	100.11%	99.77%	100.24%	98.68%	99.37%	99.82%	99.70%	99.90%
	NC/C	97.91%	99.38%	99.85%	99.53%	99.99%	98.51%	99.23%	99.66%	99.54%	99.74%
	NC/NE	99.84%	99.75%	99.74%	99.76%	99.75%	99.83%	99.86%	99.84%	99.83%	99.84%
0.75	NE/C	99.35%	100.35%	100.73%	100.40%	100.73%	99.69%	100.20%	100.12%	100.09%	100.18%
	NC/C	99.16%	99.95%	100.33%	100.03%	100.34%	99.47%	99.99%	99.90%	99.86%	99.96%
	NC/NE	99.81%	99.60%	99.61%	99.63%	99.61%	99.78%	99.79%	99.78%	99.77%	99.78%

Table A.14: Average Payoff Ratios with $\mu = 6$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	C	122.90%	109.75%	105.04%	106.12%	103.30%	114.52%	108.41%	103.90%	104.00%	103.10%
	NE	112.48%	100.62%	100.24%	100.92%	100.32%	111.60%	106.14%	103.19%	103.38%	102.04%
0.50	C	119.48%	108.71%	104.39%	105.68%	102.95%	113.29%	108.48%	103.93%	104.12%	103.12%
	NE	113.06%	100.57%	100.21%	100.94%	100.35%	110.77%	106.49%	103.40%	103.53%	102.08%
0.75	C	109.72%	104.84%	101.11%	102.72%	100.97%	106.77%	102.78%	102.44%	102.21%	101.45%
	NE	105.82%	100.34%	100.02%	100.54%	100.07%	105.58%	102.15%	101.97%	101.82%	100.98%

Table A.15: Average Payoff Ratios with and without Code Sharing with $\mu = 2$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	C	105.75%	102.30%	101.19%	101.62%	100.80%	104.43%	102.23%	101.11%	101.25%	100.74%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.03%	100.02%	100.03%	100.09%	100.01%
0.50	C	105.07%	102.03%	101.04%	101.48%	100.70%	103.91%	102.25%	101.13%	101.26%	100.75%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.01%	100.01%	100.06%	100.08%	100.02%
0.75	C	102.43%	101.11%	100.25%	100.70%	100.23%	102.02%	100.68%	100.67%	100.66%	100.37%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.03%	100.03%	100.00%

Table A.16: Average Payoff Ratios with and without Code Sharing with $\mu = 4$

CI	Ratio	1(20)	1(40)	1(60)	1(80)	1(100)	2(20)	2(40)	2(60)	2(80)	2(100)
0.25	C	102.81%	101.01%	100.49%	100.78%	100.34%	101.99%	100.94%	100.44%	100.55%	100.33%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
0.50	C	102.44%	100.91%	100.43%	100.71%	100.30%	101.75%	100.95%	100.45%	100.54%	100.33%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
0.75	C	101.19%	100.48%	100.08%	100.33%	100.09%	100.88%	100.26%	100.25%	100.28%	100.16%
	NE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.01%	100.00%

Table A.17: Average Payoff Ratios with and without Code Sharing with $\mu = 6$

Bibliography

Abdelghany, A., Sattayalekha, W., and Abdelghany, K., (2009), On Airlines Code-Share Optimisation: A Modelling Framework and Analysis, *International Journal of Revenue Management*, **3** (3), pp. 307–330. (Cited on page 23.)

Adida, E. and Perakis, G., (2010), Dynamic Pricing and Inventory Control: Uncertainty and Competition, *Operations Research*, **58** (2), pp. 289–302. (Cited on page 44.)

"Airline Alliance Survey", (2013), *Airline Business*, **29** (9), pp. 36–40. (Cited on page 21.)

Ariño, A. and Reuer, J. J., (2006), Alliance Contractual Design, in: Shenkar, O. and Reuer, J. J., eds., *Handbook of Strategic Alliances*, SAGE, Thousand Oaks, pp. 149–167. (Cited on page 20.)

Audet, C., Belhaiza, S., and Hansen, P., (2006), Enumeration of All the Extreme Equilibria in Game Theory: Bimatrix and Polymatrix Games, *Journal of Optimization Theory and Applications*, **129** (3), pp. 349–372. (Cited on pages 34 and 39.)

Audet, C., Hansen, P., Jaumard, B., and Savard, G., (2001), Enumeration of All Extreme Equilibria of Bimatrix Games, *SIAM Journal on Scientific Computing*, **23** (1), pp. 323–338. (Cited on pages 28, 34, 35, 37, 38, 39, 48, and 60.)

Avis, D., Rosenberg, G. D., Savani, R., and von Stengel, B., (2010), Enumeration of Nash Equilibria for Two-Player Games, *Economic Theory*, **42** (1), pp. 9–37. (Cited on pages 35 and 39.)

Backhaus, K. and Piltz, K., (1990), Strategische Allianzen – eine neue Form kooperativen Wettbewerbs?, in: Backhaus, K. and Piltz, K., eds., *Strategische Allianzen*, Zeitschrift für betriebswirtschaftsliche Forschung, Sonderheft 27, pp. 1–10. (Cited on page 20.)

Bazaraa, M. S., Sherali, H. D., and Shetty, C. M., (2006), *Nonlinear Programming: Theory and Algorithms*, 3rd edn., Wiley, Hoboken. (Cited on pages 94 and 98.)

Bellman, R. E., (1957), *Dynamic Programming*, Princeton University Press, Princeton. (Cited on page 15.)

Belobaba, P. P., (1987), Air Travel Demand and Airline Seat Inventory Management, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge. (Cited on page 6.)

Belobaba, P. P., (2012), Airline Network Revenue Management: Recent Developments and State of the Practice, in: Jenkins, D., ed., *The Handbook of Airline Economics*, Edward Elgar, Cheltenham. (Cited on page 15.)

Belobaba, P. P. and Jain, H., (2013), Alliance Revenue Management in Practice: Impacts of Bid Price Sharing and Dynamic Valuation, *Journal of Revenue and Pricing Management*, **12** (6), pp. 475–488. (Cited on pages 24, 109, and 124.)

Belobaba, P. P. and Wilson, J. L., (1997), Impacts of Yield Management in Competitive Airline Markets, *Journal of Air Transport Management*, **3** (1), pp. 3–9. (Cited on page 1.)

Bertsekas, D. P., (2005), *Dynamic Programming and Optimal Control*, vol. 1, 3rd edn., Athena Scientific, Belmont. (Cited on page 15.)

Bertsimas, D. and Popescu, I., (2003), Revenue Management in a Dynamic Network Environment, *Transportation Science*, **37**, pp. 257–277. (Cited on page 16.)

Bertsimas, D. and Shioda, R., (2003), Restaurant Revenue Management, *Operations Research*, **51** (3), pp. 472–486. (Cited on page 7.)

Binmore, K. G., (2003), *Fun and Games*, 2nd edn., Houghton Mifflin, Boston. (Cited on page 28.)

Birbil, Ş. İ., Frenk, J. B. G., Gromicho, J. A. S., and Zhang, S., (2014), A Network Airline Revenue Management Framework Based on Decomposition by Origins and Destinations, *Transportation Science*, **48** (3), pp. 313–333. (Cited on page 15.)

Birge, J. R. and Louveaux, F., (2011), *Introduction to Stochastic Programming*, 2nd edn., Springer, New York. (Cited on page 16.)

Bitran, G. R. and Caldentey, R., (2003), An Overview of Pricing Models for Revenue Management, *Manufacturing and Service Operations Management*, **5** (3), pp. 203–229. (Cited on pages 13 and 92.)

Bitran, G. R. and Gilbert, S. M., (1996), Managing Hotel Reservations with Uncertain Arrivals, *Operations Research*, **44** (1), pp. 35–49. (Cited on page 6.)

Bobb, L. M. and Veral, E., (2008), Open Issues and Future Directions in Revenue Management, *Journal of Revenue and Pricing Management*, **7** (3), pp. 291–301. (Cited on page 1.)

de Boer, S. V., (2003), Advances in Airline Revenue Management and Pricing, Ph.D. Thesis, Massachusetts Institute of Technology, Boston. (Cited on page 92.)

de Boer, S. V., Freling, R., and Piersma, N., (2002), Mathematical Programming for Network Revenue Management Revisited, *European Journal of Operational Research*, **137** (1), pp. 72–92. (Cited on pages 16, 45, and 95.)

Borm, P. and Peters, H., eds., (2002), *Chapters in Game Theory: In Honor of Stef Tijs*, Kluwer Academic Publishers, New York. (Cited on page 34.)

Bosse, H., Byrka, J., and Markakis, E., (2010), New Algorithms for Approximate Nash Equilibria in Bimatrix Games, *Theoretical Computer Science*, **411** (1), pp. 164–173. (Cited on page 71.)

Boyd, E. A., (1998), Airline Alliances, *OR/MS Today*, **25** (5), pp. 28–31. (Cited on pages 20, 21, 23, 24, and 112.)

Boyd, E. A. and Bilegan, I. C., (2003), Revenue Management and E-Commerce, *Management Science*, **49** (10), pp. 1363–1386. (Cited on page 13.)

Brueckner, J. K. and Pels, E., (2005), European Airline Mergers, Alliance Consolidation, and Consumer Welfare, *Journal of Air Transport Management*, **11** (1), pp. 27–41. (Cited on page 20.)

Caldentey, R. and Vulcano, G., (2007), Online Auction and List Price Revenue Management, *Management Science*, **53** (5), pp. 795–813. (Cited on page 9.)

Carroll, W. J. and Grimes, R. C., (1995), Evolutionary Change in Product Management: Experiences in the Car Rental Industry, *Interfaces*, **25** (5), pp. 84–104. (Cited on page 6.)

Casson, M. and Mol, M. J., (2006), Strategic Alliances: A Survey of Issues From an Entrepreneurial Perspective, in: Shenkar, O. and Reuer, J. J., eds., *Handbook of Strategic Alliances*, SAGE, Thousand Oaks, pp. 17–37. (Cited on page 19.)

Çetiner, D., (2013), *Fair Revenue Sharing Mechanisms for Strategic Passenger Airline Alliances*, Springer, Berlin. (Cited on pages 24 and 108.)

Çetiner, D. and Kimms, A., (2013), Assessing Fairness of Selfish Revenue Sharing Mechanisms for Airline Alliances, *Omega*, **41** (4), pp. 641–652. (Cited on pages 24, 91, and 108.)

Charnes, A., (1952), Optimality and Degeneracy in Linear Programming, *Econometrica*, **20** (2), pp. 160–170. (Cited on page 59.)

Chen, L. and Homem-de-Mello, T., (2010), Re-Solving Stochastic Programming Models for Airline Revenue Management, *Annals of Operations Research*, **177** (1), pp. 91–114. (Cited on page 123.)

Chen, V. C. P., Gunther, D., and Johnson, E. L., (1998), A Markov Decision Problem-Based Approach to the Airline Yield Management Problem, Working Paper, Georgia Institute of Technology, Atlanta. (Cited on page 14.)

Chen, X., Deng, X., and Teng, S.-H., (2009), Settling the Complexity of Computing Two-Player Nash Equilibria, *Journal of the Association for Computing Machinery*, **56** (3), pp. 1–57. (Cited on pages 49 and 71.)

Chen, X. and Hao, G., (2013), Co-Opetition Alliance Models of Parallel Flights for Determining Optimal Overbooking Policies, *Mathematical and Computer Modelling*, **57** (5-6), pp. 1101–1111. (Cited on page 25.)

Chen, Z. and Ross, T. W., (2000), Strategic Alliances, Shared Facilities, and Entry Deterrence, *The RAND Journal of Economics*, **31** (2), pp. 326–344. (Cited on page 20.)

Chew, E. P., Lee, C., and Liu, R., (2009), Joint Inventory Allocation and Pricing Decisions for Perishable Products, *International Journal of Production Economics*, **120** (1), pp. 139–150. (Cited on page 92.)

Chiang, W. C., Chen, J. C., and Xu, X., (2007), An Overview of Research on Revenue Management: Current Issues and Future Research, *International Journal of Revenue Management*, **1** (1), pp. 97–128. (Cited on pages 1, 5, 7, and 8.)

Cizaire, C. and Belobaba, P. P., (2013), Joint Optimization of Airline Pricing and Fare Class Seat Allocation, *Journal of Revenue and Pricing Management*, **12** (1), pp. 83–93. (Cited on page 92.)

Cleophas, C., Yeoman, I., McMahon-Beattie, U., and Veral, E., (2011), The Applications of Revenue Management and Pricing, in: Yeoman, I. and McMahon-Beattie, U., eds., *Revenue Management*, Palgrave Macmillan, Basingstoke, pp. 9–16. (Cited on page 7.)

Conitzer, V. and Sandholm, T., (2006), A Technique for Reducing Normal-Form Games to Compute a Nash Equilibrium, Working Paper, Carnegie Mellon University, Pittsburgh. (Cited on page 73.)

Conitzer, V. and Sandholm, T., (2008), New Complexity Results About Nash Equilibria, *Games and Economic Behavior*, **63** (2), pp. 621–641. (Cited on page 49.)

Conrady, R., Fichert, F., and Sterzenbach, R., (2013), *Luftverkehr*, 5th edn., Oldenbourg, Munich. (Cited on pages 20, 21, 23, and 26.)

Cooper, W. L., (2002), Asymptotic Behavior of an Allocation Policy for Revenue Management, *Operations Research*, **50** (4), pp. 720–727. (Cited on page 123.)

Cooper, W. L. and Homem-de-Mello, T., (2007), Some Decomposition Methods for Revenue Management, *Transportation Science*, **41** (3), pp. 332–353. (Cited on page 15.)

Cooper, W. L., Homem-de-Mello, T., and Kleywegt, A. J., (2006), Models of the Spiral-Down Effect in Revenue Management, *Operations Research*, **54** (5), pp. 968–987. (Cited on page 1.)

Côté, J.-P., Marcotte, P., and Savard, G., (2003), A Bilevel Modelling Approach to Pricing and Fare Optimisation in the Airline Industry, *Journal of Revenue and Pricing Management*, **2** (1), pp. 23–36. (Cited on pages 92 and 93.)

Cournot, A.-A., (1838), *Recherches sur les principes mathématiques de la théorie des richesses*, Hachette, Paris. (Cited on page 33.)

Cross, R. G., (1998), *Revenue Management: Hard Core Tactics for Market Domination*, Broadway Books, New York. (Cited on page 6.)

Cross, R. G., Higbie, J. A., and Cross, Z. N., (2010), Milestones in the Application of Analytical Pricing and Revenue Management, *Journal of Revenue and Pricing Management*, **10** (1), pp. 8–18. (Cited on page 7.)

Curry, R. E., (1990), Optimal Airline Seat Allocation with Fare Classes Nested by Origins and Destinations, *Transportation Science*, **24** (3), pp. 193–204. (Cited on pages 15 and 17.)

Dantzig, G. B., (1951), A Proof of the Equivalence of the Programming Problem and the Game Problem, in: Koopmans, T. C., ed., *Activity Analysis of Production and Allocation*, John Wiley & Sons, New York, pp. 330–335. (Cited on page 43.)

Das, T. K. and Teng, B.-S., (2000), Instabilities of Strategic Alliances: An Internal Tensions Perspective, *Organization Science*, **11** (1), pp. 77–101. (Cited on page 19.)

Daskalakis, C., Goldberg, P. W., and Papadimitriou, C. H., (2006), The Complexity of Computing a Nash Equilibrium, in: Kleinberg, J., ed., *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, ACM Press, New York, pp. 71–78. (Cited on page 49.)

Daskalakis, C., Goldberg, P. W., and Papadimitriou, C. H., (2009a), The Complexity of Computing a Nash Equilibrium, *SIAM Journal on Computing*, **39** (1), pp. 195–259. (Cited on page 71.)

Daskalakis, C., Mehta, A., and Papadimitriou, C. H., (2007), Progress in Approximate Nash Equilibria, in: *Proceedings of the EC 07, ACM Conference on Electronic Commerce*, pp. 355–358. (Cited on page 71.)

Daskalakis, C., Mehta, A., and Papadimitriou, C. H., (2009b), A Note on Approximate Nash Equilibria, *Theoretical Computer Science*, **410** (17), pp. 1581–1588. (Cited on page 71.)

Daskalakis, C. and Papadimitriou, C. H., (2006), Computing Pure Nash Equilibria in Graphical Games via Markov Random Fields, in: Feigenbaum, J., ed., *Proceedings of the 7th ACM Conference on Electronic Commerce*, ACM Press, New York, pp. 91–99. (Cited on page 123.)

d'Huart, O. and Belobaba, P. P., (2012), A Model of Competitive Airline Revenue Management Interactions, *Journal of Revenue and Pricing Management*, **11** (1), pp. 109–124. (Cited on page 1.)

Dror, M., Trudeau, P., and Ladany, S. P., (1988), Network Models for Seat Allocation on Flights, *Transportation Research Part B: Methodological*, **22** (4), pp. 239–250. (Cited on page 14.)

Dubovik, A. and Parakhonyak, A., (2014), Drugs, Guns, and Targeted Competition, *Games and Economic Behavior*, **87**, pp. 497–507. (Cited on page 123.)

Dunleavy, H. N., (1995), Airline Pasenger Overbooking, in: Jenkins, D., ed., *Handbook of Airline Economics*, Edward Elgar, Cheltenham, pp. 469–476. (Cited on page 13.)

Dunleavy, H. N. and Phillips, G., (2009), The Future of Airline Revenue Management, *Journal of Revenue and Pricing Management*, **8** (4), pp. 388–395. (Cited on page 1.)

Dussauge, P. and Garrette, B., (1999), *Cooperative Strategy*, Wiley, Chichester. (Cited on page 19.)

Echenique, F., (2007), Finding All Equilibria in Games of Strategic Complements, *Journal of Economic Theory*, **135** (1), pp. 514–532. (Cited on page 48.)

van den Elzen, A. H. and Talman, A. J. J., (1991), A Procedure for Finding Nash Equilibria in Bi-Matrix Games, *Zeitschrift für Operations Research*, **35** (1), pp. 27–43. (Cited on page 47.)

van den Elzen, A. H. and Talman, A. J. J., (1994), Finding a Nash Equilibrium in Noncooperative N-Person Games by Solving a Sequence of Linear Stationary Point Problems, *Zeitschrift für Operations Research*, **39** (3), pp. 365–375. (Cited on page 47.)

Etihad Airways, (2014), Etihad Airways Partners, URL: http://www.etihad.com/about-us/etihad-airways-partners/. (Cited on page 21.)

European Union, (2004), Regulation No 261/2004 of the European Parliament and of the Council, *Official Journal of the European Union*, **46** (1), pp. 1–7. (Cited on page 13.)

Feng, Y. and Xiao, B., (2006), Integration of Pricing and Capacity Allocation for Perishable Products, *European Journal of Operational Research*, **168** (1), pp. 17–34. (Cited on page 92.)

Fitzsimmons, J. A. and Fitzsimmons, M. J., (2008), *Service Management*, 6th edn., McGraw-Hill, Boston. (Cited on page 8.)

Friesz, T. L., Mookherjee, R., and Rigdon, M. A., (2007), Formulating and Solving Service Network Pricing and Resource Allocation Games as Differential Variational Inequalities, in: Jørgensen, S., Quincampoix, M., and Vincent, T. L., eds., *Advances in Dynamic Game Theory*, vol. 9, Birkhäuser, Boston, pp. 587–614. (Cited on page 44.)

Fudenberg, D. and Tirole, J., (1991), *Game Theory*, MIT Press, Cambridge. (Cited on pages 28, 29, 30, 33, and 58.)

Gale, D., Kuhn, H. W., and Tucker, A. W., (1950), Reductions of Game Matrices, in: Kuhn, H. W. and Tucker, A. W., eds., *Contributions to the Theory of Games*, vol. 1, Princeton University Press, Princeton, pp. 89–96. (Cited on page 73.)

Gallego, G. and Hu, M., (2014), Dynamic Pricing of Perishable Assets Under Competition, *Management Science*, **60** (5), pp. 1241–1259. (Cited on page 44.)

Gallego, G. and van Ryzin, G. J., (1997), A Multiproduct Dynamic Pricing Problem and Its Applications to Network Yield Management, *Operations Research*, **45** (1), pp. 24–41. (Cited on page 13.)

Gao, H., Ball, M. O., and Karaesmen, I. Z., (2010), Competitive Seat Inventory Control Decisions Under the Regret Criterion, *Journal of Revenue and Pricing Management*, **9** (1-2), pp. 49–65. (Cited on pages 31, 44, and 47.)

Geraghty, M. K. and Johnson, E., (1997), Revenue Management Saves National Car Rental, *Interfaces*, **27** (1), pp. 107–127. (Cited on page 6.)

Gilboa, I., Kalai, E., and Zemel, E., (1990), On the Order of Eliminating Dominated Strategies, *Operations Research Letters*, **9** (2), pp. 85–89. (Cited on page 33.)

Gilboa, I. and Zemel, E., (1989), Nash and Correlated Equilibria: Some Complexity Considerations, *Games and Economic Behavior*, **1** (1), pp. 80–93. (Cited on page 49.)

Glover, F., Glover, R., Lorenzo, J., and McMillan, C., (1982), The Passenger-Mix Problem in the Scheduled Airlines, *Interfaces*, **12** (3), pp. 73–80. (Cited on page 16.)

Goldman, P., Freling, R., Pak, K., and Piersma, N., (2002), Models and Techniques for Hotel Revenue Management Using a Rolling Horizon, *Journal of Revenue and Pricing Management*, **1** (3), pp. 207–219. (Cited on page 6.)

Goodwin, J. and Cook, G., (2008), Airline Networks: A Comparison of Hub-and-Spoke and Point-to-Point Systems, *The Journal of Aviation/Aerospace Education & Research*, **17** (2), pp. 51–60. (Cited on page 4.)

Graf, M., (2011), *Revenue Management for Strategic Alliances with Applications to the Airline Industry*, Dr. Hut, Munich. (Cited on page 23.)

Graf, M. and Kimms, A., (2011), An Option-Based Revenue Management Procedure for Strategic Airline Alliances, *European Journal of Operational Research*, **215** (2), pp. 459–469. (Cited on pages 24 and 108.)

Graf, M. and Kimms, A., (2013), Transfer Price Optimization for Option-Based Airline Alliance Revenue Management, *International Journal of Production Economics*, **145** (1), pp. 281–293. (Cited on pages 25 and 108.)

Grauberger, W. and Kimms, A., (2014a), Airline Revenue Management Games under Simultaneous Price and Quantity Competition, Working Paper, University of Duisburg-Essen, Duisburg. (Cited on pages 91, 92, and 93.)

Grauberger, W. and Kimms, A., (2014b), Computing Approximate Nash Equilibria in General Network Revenue Management Games, *European Journal of Operational Research*, **237** (3), pp. 1008–1020. (Cited on pages 43, 45, 68, and 77.)

Grauberger, W. and Kimms, A., (2014c), Computing Pure Nash Equilibria in Network Revenue Management Games, Working Paper, University of Duisburg-Essen, Duisburg. (Cited on pages 43, 45, and 49.)

Grauberger, W. and Kimms, A., (2014d), Revenue Management under Competition within Airline Alliances, Working Paper, University of Duisburg-Essen, Duisburg. (Cited on pages 108 and 109.)

Gupta, D. and Wang, L., (2008), Revenue Management for a Primary-Care Clinic in the Presence of Patient Choice, *Operations Research*, **56** (3), pp. 576–592. (Cited on page 7.)

Harks, T., Hoefer, M., Klimm, M., and Skopalik, A., (2010), Computing Pure Nash and Strong Equilibria in Bottleneck Congestion Games, in: Berg, M. and Meyer, U., eds., *Algorithms ESA 2010*, *Lecture Notes in Computer Science*, vol. 6347, Springer, Berlin, pp. 29–38. (Cited on page 48.)

Harsanyi, J. C. and Selten, R., (1988), *A General Theory of Equilibrium Selection in Games*, MIT Press, Cambridge. (Cited on pages 58, 74, and 75.)

Hazan, E. and Krauthgamer, R., (2011), How Hard is it to Approximate the Best Nash Equilibrium?, *SIAM Journal on Computing*, **40** (1), pp. 79–91. (Cited on page 49.)

Hémon, S., de Rougemont, M., and Santha, M., (2008), Approximate Nash Equilibria for Multi-Player Games, in: Monien, B. and Schroeder, U.-P., eds., *Algorithmic Game Theory*, *Lecture Notes in Computer Science*, vol. 4997, Springer, Berlin, pp. 267–278. (Cited on page 71.)

Hergert, M. and Morris, D., (1988), Trends in International Collaborative Agreements, in: Contractor, F. J. and Lorange, P., eds., *Cooperative Strategies in International Business*, Lexington Books, Lexington, pp. 99–110. (Cited on page 21.)

Hodges, J. L. and Lehmann, E. L., (2005), *Basic Concepts of Probability and Statistics*, SIAM, Philadelphia. (Cited on page 17.)

Holler, M. J. and Illing, G., (2009), *Einführung in die Spieltheorie*, 7th edn., Springer, Berlin. (Cited on pages 28, 30, 33, and 60.)

Hoseason, J., (2002), Capacity Management in the Cruise Industry, in: Ingold, A., McMahon-Beattie, U., and Yeoman, I., eds., *Yield Management*, Continuum, London, pp. 289–302. (Cited on page 7.)

Howson, J. T. J., (1972), Equilibria of Polymatrix Games, *Management Science*, **18** (5), pp. 312–318. (Cited on page 48.)

Hu, X., Caldentey, R., and Vulcano, G., (2013), Revenue Sharing in Airline Alliances, *Management Science*, **59** (5), pp. 1177–1195. (Cited on pages 25, 109, and 124.)

Isler, K. and Imhof, H., (2008), A Game Theoretic Model for Airline Revenue Management and Competitive Pricing, *Journal of Revenue and Pricing Management*, **7** (4), pp. 384–396. (Cited on page 123.)

Jacobs, T. L., Ratliff, R. M., and Smith, B. C., (2000), Soaring with Synchronized Systems, *OR/MS Today*, **27** (4), pp. 36–44. (Cited on page 91.)

Jasin, S. and Kumar, S., (2012), A Re-Solving Heuristic with Bounded Revenue Loss for Network Revenue Management with Customer Choice, *Mathematics of Operations Research*, **37** (2), pp. 313–345. (Cited on pages 16 and 123.)

Jasin, S. and Kumar, S., (2013), Analysis of Deterministic LP-Based Booking Limit and Bid Price Controls for Revenue Management, *Operations Research*, **61** (6), pp. 1312–1320. (Cited on page 16.)

Jiang, A. X. and Leyton-Brown, K., (2007), Computing Pure Nash Equilibria in Symmetric Action Graph Games, in: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, AAAI Press, Menlo Park, pp. 79–85. (Cited on page 48.)

Jiang, H. and Pang, Z., (2011), Network Capacity Management Under Competition, *Computational Optimization and Applications*, **50** (2), pp. 287–326. (Cited on pages 41, 44, 45, 47, 73, and 123.)

Jones, P., (2002), Defining Yield Management and Measuring its Impact on Hotel Performance, in: Ingold, A., McMahon-Beattie, U., and Yeoman, I., eds., *Yield Management*, Continuum, London, pp. 85–97. (Cited on page 6.)

Kimes, S. E., (1989), Yield Management: A Tool for Capacity-Constrained Service Firms, *Journal of Operations Management*, **8** (4), pp. 348–363. (Cited on pages 7, 8, 9, and 10.)

Kimes, S. E., (2002), A Strategic Approach to Yield Management, in: Ingold, A., McMahon-Beattie, U., and Yeoman, I., eds., *Yield Management*, Continuum, London, pp. 3–14. (Cited on page 6.)

Kimms, A. and Çetiner, D., (2012), Approximate Nucleolus-Based Revenue Sharing in Airline Alliances, *European Journal of Operational Research*, **220** (2), pp. 510–521. (Cited on pages 24, 91, and 108.)

Kimms, A. and Klein, R., (2005), Revenue Management im Branchenvergleich, *Zeitschrift für Betriebswirtschaft, Ergänzungsheft 1 "Revenue Management"*, pp. 1–30. (Cited on pages 6, 8, 9, 10, 11, and 12.)

Kimms, A. and Müller-Bungart, M., (2003), Revenue Management beim Verkauf auftragsorientierter Sachleistungen, Working Paper, Technical University Bergakademie Freiberg. (Cited on page 8.)

Kimms, A. and Müller-Bungart, M., (2006), Simulation of Stochastic Demand Data Streams for Network Revenue Management Problems, *OR Spectrum*, **29** (1), pp. 5–20. (Cited on page 18.)

Kimms, A. and Müller-Bungart, M., (2007), Revenue Management for Broadcasting Commercials: The Channel's Problem of Selecting and Scheduling Ads to be Aired, *International Journal of Revenue Management*, **1**, pp. 28–44. (Cited on page 7.)

Klein, R., (2001), Revenue Management: Quantitative Methoden zur Erlösmaximierung in der Dienstleistungsproduktion, *Betriebswirtschaftliche Forschung und Praxis*, **53**, pp. 245–259. (Cited on pages 6, 8, and 11.)

Klein, R. and Steinhardt, C., (2008), *Revenue Management: Grundlagen und mathematische Methoden*, Springer, Berlin. (Cited on pages 7, 8, 9, 10, 11, 13, 14, and 15.)

Kleymann, B. and Seristö, H., (2004), *Managing Strategic Airline Alliances*, Ashgate, Aldershot. (Cited on pages 20, 21, 22, and 26.)

Knuth, D. E., Papadimitriou, C. H., and Tsitsiklis, J. N., (1988), A Note on Strategy Elimination in Bimatrix Games, *Operations Research Letters*, **7** (3), pp. 103–107. (Cited on page 33.)

Kocabiyikoğlu, A., Popescu, I., and Stefanescu, C., (2014), Pricing and Revenue Management: The Value of Coordination, *Management Science*, **60** (3), pp. 730–752. (Cited on page 91.)

Kontogiannis, S. C., Panagopoulou, P. N., and Spirakis, P. G., (2009), Polynomial Algorithms for Approximating Nash Equilibria of Bimatrix Games, *Theoretical Computer Science*, **410** (17), pp. 1599–1606. (Cited on pages 35 and 71.)

Kontogiannis, S. C. and Spirakis, P. G., (2007a), Efficient Algorithms for Constant Well Supported Approximate Equilibria in Bimatrix Games, in: Arge, L., Cachin, C., Jurdzínski, T., and Tarlecki, A., eds., *Automata, Languages and Programming*, Springer, Berlin, pp. 595–606. (Cited on page 71.)

Kontogiannis, S. C. and Spirakis, P. G., (2007b), Well Supported Approximate Equilibria in Bimatrix Games: A Graph Theoretic Approach, in: Kučera, L. and Kučera, A., eds., *Mathematical Foundations of Computer Science 2007*, Springer, Berlin, pp. 596–608. (Cited on page 71.)

Kuhn, H. W., (1961), An Algorithm for Equilibrium Points in Bimatrix Games, *Proceedings of the National Academy of Sciences of the United States of America*, **47** (10), pp. 1657–1662. (Cited on page 33.)

Kunnumkal, S. and Topaloglu, H., (2008), A Refined Deterministic Linear Program for the Network Revenue Management Problem with Customer Choice Behavior, *Naval Research Logistics*, **55** (6), pp. 563–580. (Cited on page 18.)

LaRoche, M.-A., Gamache, M., and Olivier-Ouellet, J., (2012), Optimization of Codeshare Flight Selection for an Airline Company, *Information Systems and Operational Research*, **50** (1), pp. 31–39. (Cited on page 23.)

Lee, T. C. and Hersh, M., (1993), A Model for Dynamic Airline Seat Inventory Control with Multiple Seat Bookings, *Transportation Science*, **27** (3), pp. 252–265. (Cited on page 16.)

Lemke, C. E. and Howson, J. T. J., (1964), Equilibrium Points of Bimatrix Games, *Journal of the Society for Industrial and Applied Mathematics*, **12** (2), pp. 413–423. (Cited on pages 47, 58, 59, and 76.)

Levin, Y., McGill, J., and Nediak, M., (2009), Dynamic Pricing in the Presence of Strategic Consumers and Oligopolistic Competition, *Management Science*, **55** (1), pp. 32–46. (Cited on page 44.)

Li, M. Z. F. and Oum, T. H., (1998), Seat Allocation Game on Flights with Two Fares, Working Paper, Nanyang Technological University, Nanyang. (Cited on pages 44 and 47.)

Li, M. Z. F., Oum, T. H., and Anderson, C. K., (2007), An Airline Seat Allocation Game, *Journal of Revenue and Pricing Management*, **6** (4), pp. 321–330. (Cited on pages 44, 47, and 122.)

Li, M. Z. F., Zhang, A., and Zhang, Y., (2008), Airline Seat Allocation Competition, *International Transactions in Operational Research*, **15** (4), pp. 439–459. (Cited on pages 31, 41, 44, 47, 112, and 122.)

Lim, W. S., (2009), Overselling in a Competitive Environment: Boon or Bane?, *Marketing Science*, **28** (6), pp. 1129–1143. (Cited on pages 44 and 122.)

Lin, K. Y. and Sibdari, S. Y., (2009), Dynamic Price Competition with Discrete Customer Choices, *European Journal of Operational Research*, **197** (3), pp. 969–980. (Cited on page 44.)

Littlewood, K., (1972), Forecasting and Control of Passenger Bookings, in: *Proceedings of the Twelfth Annual AGIFORS Symposium*, Nathanya, Isreal, pp. 95–117. (Cited on page 5.)

Liu, Q. and van Ryzin, G. J., (2008), On the Choice-Based Linear Programming Model for Network Revenue Management, *Manufacturing and Service Operations Management*, **10** (2), pp. 288–310. (Cited on page 18.)

Liu, Q. and Zhang, D., (2013), Dynamic Pricing Competition with Strategic Customers Under Vertical Product Differentiation, *Management Science*, **59** (1), pp. 84–101. (Cited on page 44.)

Lorange, P. and Roos, J., (1993), *Strategic Alliances: Formation, Implementation, and Evolution*, Blackwell, Cambridge. (Cited on pages 19, 21, and 26.)

Lufthansa, (2014), Annual Report. (Cited on page 88.)

Maglaras, C. and Meissner, J., (2006), Dynamic Pricing Strategies for Multiproduct Revenue Management Problems, *Manufacturing & Service Operations Management*, **8** (2), pp. 136–148. (Cited on page 13.)

Maleri, R. and Frietzsche, U., (2008), *Grundlagen der Dienstleistungsproduktion*, 5th edn., Springer, Berlin. (Cited on page 8.)

Mangasarian, O. L., (1964), Equilibrium Points of Bimatrix Games, *Journal of the Society for Industrial and Applied Mathematics*, **12** (4), pp. 778–780. (Cited on page 34.)

Mangasarian, O. L. and Stone, H., (1964), Two-Person Nonzero-Sum Games and Quadratic Programming, *Journal of Mathematical Analysis and Applications*, **9** (3), pp. 317–504. (Cited on page 38.)

Martínez-de-Albéniz, V. and Talluri, K. T., (2011), Dynamic Price Competition with Fixed Capacities, *Management Science*, **57** (6), pp. 1078–1093. (Cited on pages 1 and 44.)

Marx, L. M. and Swinkels, J. M., (1997), Order Independence for Iterated Weak Dominance, *Games and Economic Behavior*, **18** (2), pp. 219–245. (Cited on page 33.)

Maurer, P., (2006), *Luftverkehrsmanagement*, 4th edn., Oldenbourg, Munich. (Cited on pages 20, 21, and 23.)

Mavronicolas, M., Papadopoulou, V., and Spirakis, P. G., (2008), Algorithmic Game Theory and Applications, in: Nayak, A. and Stojmenovic, I., eds., *Handbook of Applied Algorithms*, Wiley-Interscience, Hoboken, pp. 285–313. (Cited on page 33.)

Mayer, G., (2001), *Strategische Logistikplanung von Hub&Spoke-Systemen*, Gabler, Wiesbaden. (Cited on page 10.)

McGill, J. I. and van Ryzin, G. J., (1999), Revenue Management: Research Overview and Prospects, *Transportation Science*, **33** (2), pp. 233–256. (Cited on pages 4 and 5.)

McKelvey, R. D. and McLennan, A., (1996), Computation of Equilibria in Finite Games, in: Amman, H. M., Kendrick, D. A., and Rust, J., eds., *Handbook of Computational Economics*, vol. 1, Elsevier, Amsterdam, pp. 87–142. (Cited on pages 48, 58, and 76.)

McKelvey, R. D., McLennan, A. M., and Turocy, T. L., (2014), Gambit: Software Tools for Game Theory, Version 14.1.0, http://www.gambit-project.org. (Cited on pages 41 and 77.)

Metters, R. and Vargas, V., (1999), Yield Management for the Nonprofit Sector, *Journal of Service Research*, **1** (3), pp. 215–226. (Cited on page 7.)

Mohr, J. and Spekman, R., (1994), Characteristics of Partnership Success: Partnership Attributes, Communication Behavior, and Conflict Resolution Techniques, *Strategic Management Journal*, **15** (2), pp. 135–152. (Cited on page 19.)

Mookherjee, R. and Friesz, T. L., (2008), Pricing, Allocation, and Overbooking in Dynamic Service Network Competition when Demand is Uncertain, *Production and Operations Management*, **17** (4), pp. 455–474. (Cited on pages 44 and 122.)

Moore, D. S., McCabe, G. P., and Craig, B. A., (2009), *Introduction to the Practice of Statistics*, 6th edn., Freeman, New York. (Cited on page 17.)

Müller-Bungart, M., (2007), *Revenue Management with Flexible Products*, Springer, Berlin. (Cited on pages 7, 8, 9, 10, 13, 17, and 18.)

Murty, K. G., (1983), *Linear Programming*, Wiley, New York. (Cited on pages 43, 47, and 59.)

Myerson, R. B., (2001), *Game Theory: Analysis of Conflict*, 4th edn., Harvard University Press, Cambridge. (Cited on pages 28, 30, 33, and 35.)

Nash, J. F., (1950), Equilibrium Points in N-Person Games, *Proceedings of the National Academy of Sciences of the United States of America*, **36** (1), pp. 48–49. (Cited on page 30.)

Nash, J. F., (1951), Non-Cooperative Games, *The Annals of Mathematics*, **54** (2), pp. 286–295. (Cited on pages 30 and 50.)

Nason, S. D., (2009), The Six C's of Modern Airline Competition, *Journal of Revenue and Pricing Management*, **8** (4), pp. 291–294. (Cited on page 1.)

Netessine, S. and Shumsky, R. A., (2005), Revenue Management Games: Horizontal and Vertical Competition, *Management Science*, **51** (5), pp. 813–831. (Cited on pages 2, 25, 31, 41, 44, 47, 51, 68, 93, 109, 112, and 113.)

von Neumann, J., (1928), Zur Theorie der Gesellschaftsspiele, *Mathematische Annalen*, **100**, pp. 295–300. (Cited on page 43.)

von Neumann, J. and Morgenstern, O., (1953), *Theory of Games and Economic Behavior*, 3rd edn., Princeton University Press, Princeton. (Cited on page 30.)

Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. V., eds., (2007), *Algorithmic Game Theory*, Cambridge University Press, Cambridge. (Cited on page 33.)

O'Neal, J. W., Jacob, M. S., Farmer, A. K., and Martin, K. G., (2007), Development of a Codeshare Flight-Profitability System at Delta Air Lines, *Interfaces*, **37** (5), pp. 436–444. (Cited on pages 23 and 108.)

OneWorld, (2014a), Member Airlines, URL: http://www.oneworld.com/member-airlines/overview. (Cited on page 60.)

OneWorld, (2014b), US Airways to Join OneWorld on 31 March 2014, http://www.oneworld.com/news-information/oneworldnews. (Cited on page 26.)

Osborne, M. J. and Rubinstein, A., (1995), *A Course in Game Theory*, MIT Press, Cambridge. (Cited on pages 28, 29, 30, 32, 33, and 35.)

Oum, T. H. and Park, J.-H., (1997), Airline Alliances: Current Status, Policy Issues, and Future Directions, *Journal of Air Transport Management*, **3** (3), pp. 133–144. (Cited on page 108.)

Oum, T. H., Park, J.-H., and Zhang, A., (2000), *Globalization and Strategic Alliances*, Pergamon, Amsterdam. (Cited on pages 20 and 21.)

Oum, T. H., Taylor, A. J., and Zhang, A., (1993), Strategic Airline Policy in the Globalizing Airline Networks, *Transportation Journal*, **32** (3), pp. 14–30. (Cited on page 22.)

Oum, T. H., Yu, C., and Zhang, A., (2001), Global Airline Alliances: International Regulatory Issues, *Journal of Air Transport Management*, **7** (1), pp. 57–62. (Cited on pages 20, 21, and 108.)

Pak, K. and Piersma, N., (2002), Overview of OR Techniques for Airline Revenue Management, *Statistica Neerlandica*, **56** (4), pp. 480–496. (Cited on page 12.)

Panagopoulou, P. N. and Spirakis, P. G., (2006), Algorithms for Pure Nash Equilibria in Weighted Congestion Games, *Journal of Experimental Algorithmics*, **11**, pp. 1–19. (Cited on page 48.)

Panagopoulou, P. N. and Spirakis, P. G., (2007), Approximate and Well-Supported Approximate Nash Equilibria of Random Bimatrix Games, in: *Proceedings of the 11th Panhellenic Conference on Informatics (PIC 2007)*, pp. 569–578. (Cited on page 71.)

Papadimitriou, C. H., (1994), On the Complexity of the Parity Argument and Other Inefficient Proofs of Existence, *Journal of Computer and System Sciences*, **48** (3), pp. 498–532. (Cited on page 49.)

Papadimitriou, C. H., (2007), The Complexity of Finding Nash Equilibria, in: Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. V., eds., *Algorithmic Game Theory*, Cambridge University Press, Cambridge, pp. 29–51. (Cited on page 49.)

Park, J.-H., (1997), The Effects of Airline Alliances on Markets and Economic Welfare, *Transportation Research Part E: Logistics and Transportation Review*, **33** (3), pp. 181–195. (Cited on page 108.)

Pels, E., (2001), A Note on Airline Alliances, *Journal of Air Transport Management*, **7** (1), pp. 3–7. (Cited on page 21.)

Petersen, T., (1994), On the Promise of Game Theory in Sociology, *Contemporary Sociology*, **23** (4), pp. 498–502. (Cited on page 1.)

Phillips, R. L., (2007), *Pricing and Revenue Optimization*, Stanford University Press, Stanford. (Cited on pages 1, 6, 11, 13, and 93.)

Porter, M. E. and Fuller, M. B., (1986), Coalitions and Global Strategy, in: Porter, M. E., ed., *Competition in Global Industries*, Harvard Business School Press, Boston, pp. 315–343. (Cited on page 21.)

Porter, R. W., Nudelman, E., and Shoham, Y., (2008), Simple Search Methods for Finding a Nash Equilibrium, *Games and Economic Behavior*, **63** (2), pp. 642–662. (Cited on page 48.)

Ratliff, R. and Vinod, B., (2005), Future of Revenue Management: Airline Pricing and Revenue Management, *Journal of Revenue and Pricing Management*, **4** (3), pp. 302–307. (Cited on page 1.)

Raza, A. S. and Akgunduz, A., (2008), An Airline Revenue Management Pricing Game with Seat Allocation, *International Journal of Revenue Management*, **2** (1), pp. 42–62. (Cited on pages 92 and 93.)

Rosenmüller, J., (1971), On a Generalization of the Lemke-Howson Algorithm to Noncooperative N-Person Games, *SIAM Journal on Applied Mathematics*, **21** (1), pp. 73–79. (Cited on page 48.)

Roughgarden, T., (2008), Algorithmic Game Theory: Some Greatest Hits and Future Directions, in: Ausiello, G., Karhumäki, J., Mauri, G., and Ong, L., eds., *Proceedings of the Fifth IFIP International Conference on Theoretical Computer Science*, vol. 273, Springer, New York, pp. 21–42. (Cited on page 33.)

Roughgarden, T., (2010), Computing Equilibria: A Computational Complexity Perspective, *Economic Theory*, **42** (1), pp. 193–236. (Cited on page 48.)

Ryan, C. T., Jiang, A. X., and Leyton-Brown, K., (2010), Computing Pure Strategy Nash Equilibria in Compact Symmetric Games, in: *Proceedings of the 2010 ACM Conference on Electronic Commerce*, ACM, New York, pp. 63–72. (Cited on page 48.)

Sandholm, T., Gilpin, A., and Conitzer, V., (2005), Mixed-Integer Programming Methods for Finding Nash Equilibria, in: Veloso, M. and Kambhampati, S., eds., *Proceedings of the Twentieth National Conference on Artificial Intelligence*, AAAI Press, Menlo Park, pp. 495–501. (Cited on pages 34 and 48.)

Savin, S. V., Cohen, M. A., Gans, N., and Katalan, Z., (2005), Capacity Management in Rental Businesses with Two Customer Bases, *Operations Research*, **53** (4), pp. 617–631. (Cited on page 6.)

Sfodera, F., ed., (2006), *The Spread of Yield Management Practices*, Physica-Verlag, Heidelberg. (Cited on page 7.)

Shapley, L. S., (1974), A Note on the Lemke-Howson Algorithm, *Mathematical Programming Studies*, **1**, pp. 175–189. (Cited on pages 36 and 48.)

Shenkar, O. and Reuer, J. J., eds., (2006), *Handbook of Strategic Alliances*, SAGE, Thousand Oaks. (Cited on page 19.)

Shugan, S. M., (2002), Marketing Science, Models, Monopoly Models, and why we Need Them, *Marketing Science*, **21** (3), pp. 223–228. (Cited on page 1.)

Shumsky, R., (2006), The Southwest Effect, Airline Alliances and Revenue Management, *Journal of Revenue and Pricing Management*, **5** (1), pp. 83–89. (Cited on pages 24 and 112.)

SkyTeam, (2014), SkyTeam Members, URL: http://www.skyteam.com/en/About-us/Our-members/. (Cited on page 60.)

Spengler, T., Rehkopf, S., and Volling, T., (2006), Revenue Management in Make-to-Order Manufacturing—An Application to the Iron and Steel Industry, *OR Spectrum*, **29** (1), pp. 157–171. (Cited on page 7.)

Spirakis, P. G., (2008), Approximate Equilibria for Strategic Two Person Games, in: Monien, B. and Schroeder, U.-P., eds., *Algorithmic Game Theory*, *Lecture Notes in Computer Science*, vol. 4997, Springer, Berlin, pp. 5–21. (Cited on page 71.)

StarAlliance, (2014), Member Airlines, URL: http://www.staralliance.com/en/about/member_airlines/. (Cited on page 60.)

Steinhardt, C. and Gönsch, J., (2012), Integrated Revenue Management Approaches for Capacity Control with Planned Upgrades, *European Journal of Operational Research*, **223** (2), pp. 380–391. (Cited on pages 6 and 18.)

von Stengel, B., (2002), Computing Equilibria for Two-Person Games, in: Aumann, R. J. and Hart, S., eds., *Handbook of Game Theory with Economic Applications*, North Holland, Amsterdam, pp. 1723–1759. (Cited on pages 36 and 48.)

von Stengel, B., (2007), Equilibrium Computation for Two-Player Games in Strategic and Extensive Form, in: Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. V., eds., *Algorithmic Game Theory*, Cambridge University Press, Cambridge, pp. 53–78. (Cited on pages 31, 36, and 48.)

von Stengel, B., ed., (2010), *Economic Theory*, Volume 42, Issue 1. (Cited on page 48.)

Subramanian, J., Stidham, S., and Lautenbacher, C. J., (1999), Airline Yield Management with Overbooking, Cancellations, and No-Shows, *Transportation Science*, **33** (2), pp. 147–167. (Cited on pages 13 and 16.)

Talluri, K. T. and van Ryzin, G., (1999), A Randomized Linear Programming Method for Computing Network Bid Prices, *Transportation Science*, **33** (2), pp. 207–216. (Cited on pages 16 and 17.)

Talluri, K. T. and van Ryzin, G. J., (2004a), Revenue Management Under a General Discrete Choice Model of Consumer Behavior, *Management Science*, **50** (1), pp. 15–33. (Cited on pages 16, 95, and 122.)

Talluri, K. T. and van Ryzin, G. J., (2004b), *The Theory and Practice of Revenue Management*, Kluwer Academic Publishers, Boston. (Cited on pages 6, 7, 10, 11, 12, 13, 14, 15, 17, 18, 22, 93, and 95.)

Topaloglu, H., (2012), A Duality Based Approach for Network Revenue Management in Airline Alliances, *Journal of Revenue and Pricing Management*, **11** (5), pp. 500–517. (Cited on pages 25 and 108.)

Topkis, D. M., (1998), *Supermodularity and Complementarity*, Princeton University Press, Princeton. (Cited on page 52.)

Transchel, S. and Shumsky, R. A., (2012), Frenemies: Price Competition between Codesharing Airlines, *International Annual Conference of the German Operations Research Society (GOR)*, Hannover (Germany). (Cited on page 110.)

Tsaknakis, H. and Spirakis, P. G., (2007), An Optimization Approach for Approximate Nash Equilibria, in: Deng, X. and Graham, F. C., eds., *Internet and Network Economics / Lecture Notes in Computer Science*, vol. 4858, Springer, Berlin, pp. 42–56. (Cited on page 71.)

de Véricourt, F. and Lobo, M. S., (2009), Resource and Revenue Management in Nonprofit Operations, *Operations Research*, **57** (5), pp. 1114–1128. (Cited on page 7.)

Vinod, B., (2004), Unlocking the Value of Revenue Management in the Hotel Industry, *Journal of Revenue and Pricing Management*, **3** (2), pp. 178–190. (Cited on page 6.)

Vinod, B., (2005), Alliance Revenue Management, *Journal of Revenue and Pricing Management*, **4** (1), pp. 66–82. (Cited on pages 24 and 112.)

Vöcking, B., ed., (2013), *Algorithmic Game Theory: Proceedings of the 6th International Symposium, SAGT*, vol. 8146, Springer, Berlin. (Cited on page 33.)

Vorob'ev, N. N., (1958), Equilibrium Points in Bimatrix Games, *Theory of Probability and its Applications*, **3** (3), pp. 297–309. (Cited on page 33.)

Vulcano, G., van Ryzin, G. J., and Maglaras, C., (2002), Optimal Dynamic Auctions for Revenue Management, *Management Science*, **48** (11), pp. 1388–1407. (Cited on page 9.)

Weatherford, L. R., (1997), Using Prices More Realistically as Decision Variables in Perishable-Asset Revenue Management Problems, *Journal of Combinatorial Optimization*, **1** (3), pp. 277–304. (Cited on page 92.)

Weatherford, L. R. and Bodily, S. E., (1992), A Taxonomy and Research Overview of Perishable-Asset Revenue Management: Yield Management, Overbooking, and Pricing, *Operations Research*, **40** (5), pp. 831–844. (Cited on pages 6, 7, and 8.)

Williamson, E. L., (1992), Airline Network Seat Inventory Control: Methodologies and Revenue Impacts, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge. (Cited on page 15.)

Wilson, J. L., (1995), The Value of Revenue Management Innovation in a Competitive Environment, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge. (Cited on page 43.)

Wilson, R., (1971), Computing Equilibria of N-Person Games, *SIAM Journal on Applied Mathematics*, **21** (1), pp. 80–87. (Cited on page 48.)

Wong, J.-T., Koppelman, F. S., and Daskin, M. S., (1993), Flexible Assignment Approach to Itinerary Seat Allocation, *Transportation Research Part B: Methodological*, **27** (1), pp. 33–48. (Cited on page 14.)

Wright, C. P., (2014), Decomposing Airline Alliances: A Bid-Price Approach to Revenue Management with Incomplete Information Sharing, *Journal of Revenue and Pricing Management*, **13** (3), pp. 164–182. (Cited on pages 25, 109, and 124.)

Wright, C. P., Groenevelt, H., and Shumsky, R. A., (2010), Dynamic Revenue Management in Airline Alliances, *Transportation Science*, **44** (1), pp. 15–37. (Cited on pages 2, 25, and 109.)

Yeoman, I. and McMahon-Beattie, U., eds., (2004), *Revenue Management and Pricing: Case Studies and Applications*, Thomson, London. (Cited on page 7.)

Yeoman, I. and McMahon-Beattie, U., eds., (2011), *Revenue Management: A Practical Pricing Perspective*, Palgrave Macmillan, Basingstoke. (Cited on page 7.)

Zhang, A. and Zhang, Y., (2006), Rivalry between Strategic Alliances, *International Journal of Industrial Organization*, **24** (2), pp. 287–301. (Cited on page 21.)

Zhao, W. and Zheng, Y.-S., (2001), A Dynamic Model for Airline Seat Allocation with Passenger Diversion and No-Shows, *Transportation Science*, **35** (1), pp. 80–98. (Cited on pages 13 and 16.)

Zhao, X. and Atkins, D. R., (2008), Newsvendors Under Simultaneous Price and Inventory Competition, *Manufacturing and Service Operations Management*, **10** (3), pp. 539–546. (Cited on pages 93 and 102.)

Zimmermann, B., (2014), Revenue Management with Repeated Competitive Interactions, Ph.D. Thesis, Freie Universität Berlin, Berlin. (Cited on page 123.)