Diss. ETH No. 17387

The P-Matrix Linear Complementarity Problem — Generalizations and Specializations

A dissertation submitted to the
SWISS FEDERAL INSTITUTE OF TECHNOLOGY
ZURICH

for the degree of
Doctor of Sciences

presented by
LEONARD YVES RÜST
M.Sc. ETH in Computer Science
born March 23, 1980
citizen of Thal (SG), Switzerland

accepted on the recommendation of
Prof. Dr. Emo Welzl, ETH Zurich, examiner
Dr. Bernd Gärtner, ETH Zurich, co-examiner
Prof. Dr. Hans-Jakob Lüthi, ETH Zurich, co-examiner
Prof. Dr. Walter D. Morris, Jr., George Mason University, Fairfax, co-examiner

2007

Abstract

The goal of this thesis is to give a better understanding of the *linear complementarity problem* with a P-*matrix* (PLCP). Finding a polynomial time algorithm for the PLCP is a longstanding open problem. Such an algorithm would settle the complexity status of many problems reducing to the PLCP. Most of the papers dealing with the PLCP look at it from an algebraic point of view. We analyze the combinatorial structure of the PLCP.

Wherever possible, we state our results for the *generalized* PLCP (PGLCP), a natural generalization of the PLCP. In the first part of the thesis, we present further generalizations of the PGLCP. We show that the PGLCP fits into the framework of *unique sink orientations* (USO) of grids. Finding a solution to the PGLCP can therefore be done by finding the sink in a grid USO. Several algorithms are known for this latter task, and we analyze the behavior of some of them on small grids. We thereby make use of the result that PGLCP-induced USOs fulfill a combinatorial property known as the *Holt-Klee condition.*

Grid USOs are then shown to fit into the framework of *violator spaces.* Violator spaces have been introduced as a generalization of LP-*type problems* capturing the combinatorics behind many problems like linear programming or finding the smallest enclosing ball of a point set. We prove that Clarkson's algorithms, originally developed for low-dimensional linear programs, work for violator spaces that in contrast to LP-type problems have a cyclic structure in general. This yields an optimal linear time algorithm for solving PGLCP with a fixed number of blocks.

The second part of the thesis deals with specializations of the PGLCP. We first focus on a subclass of P-matrices, known as *hidden* K-*matrices*. The hidden K-matrix PGLCP is known to be solvable in time polynomial in the input size. We give an alternative proof for this fact and strengthen it by the following statement: the USO arising from a hidden K-matrix PGLCP is LP-induced and therefore always acyclic. Furthermore, a nontrivial and large subclass of non hidden K-matrices is given, and we prove that the PLCP with a 3-dimensional square P-matrix in this class reduces to a cyclic USO in general.

Our last result is that *simple stochastic games* (SSG) can be reformulated as PGLCP. People have unavailingly been trying to show that games like SSG are polynomial-time solvable for over 15 years. The connection to PGLCP gives us powerful tools to further attack this. Unfortunately, SSG do in general not reduce to PGLCP with a matrix in a known polynomially solvable class.

Zusammenfassung

Das Ziel dieser Dissertation ist es, das *lineare Komplementaritätsproblem mit einer* P-*Matrix* (PLCP) besser zu verstehen. Einen Algorithmus zu finden der das PLCP in polynomieller Zeit löst ist ein seit langem offenes Problem. So ein Algorithmus würde den Komplexitätsstatus vieler Probleme festlegen, die auf das PLCP reduzierbar sind. In den meisten Arbeiten über das PLCP wird es von einer algebraischen Sichtweise analysiert. Im Gegensatz dazu schauen wir uns die kombinatorische Struktur des PLCP genauer an.

Wann immer möglich formulieren wir unsere Resultate für das *verallgemeinerte* PLCP (PGLCP), eine natürliche Verallgemeinerung des PLCP. Im ersten Teil dieser Arbeit präsentieren wir eine weitere Verallgemeinerung des PGLCP. Wir zeigen dass das PGLCP formuliert werden kann als das Problem, die eindeutige Senke einer *eindeutige-Senke-Orientierung* (USO) eines Gitters zu finden. Wir schauen uns das Verhalten einiger Algorithmen für das USO-Problem auf kleinen Gittern an. Dabei benutzen wir das Resultat, dass die USO, die von einem PGLCP stammen, die *Holt-Klee Eigenschaft* haben.

Danach zeigen wir, dass USO auf Gittern wiederum verallgemeinert werden können, und zwar auf *Verletzer-Räume*. Diese wurden als Verallgemeinerung von Problemen, die ähnlich wie *lineares Programmieren* (LP) sind, eingeführt und haben eine zyklische zugrunde liegende Struktur. Wir beweisen, dass die bekannten LP-Algorithmen von Clarkson auf Verletzer-Räume angewandt werden können und erhalten so einen optimalen Algorithmus, der das PGLCP mit einer fixen Anzahl Blöcken in linearer Zeit löst.

Im zweiten Teil dieser Dissertation befassen wir uns mit Spezialisierungen des PGLCP. Zuerst schauen wir uns eine Unterklasse von P-Matrizen an, bekannt als *versteckte* K-*Matrizen.* Das PGLCP mit einer versteckten K-Matrix kann mittels LP gelöst werden. Wir geben einen alternativen Beweis für diese Aussage und stärken sie zudem wie folgt: die USO die aus dem PGLCP mit einer versteckten K-Matrix resultiert stammt von einem LP und ist deshalb azyklisch. Zudem definieren wir eine neue Unterklasse von Matrizen die nicht versteckt K sind und zeigen, dass das PLCP mit 3-dimensionalen quadratischen P-Matrizen in dieser Klasse im Allgemeinen in einer zyklischen USO resultiert.

Unser letztes Resultat ist, dass *einfache stochastische Spiele* (SSG) als PGLCP formuliert werden können. Seit über 15 Jahren versucht man zu zeigen, dass SSG in polynomieller Zeit gelöst werden können. Die von uns aufgezeigte Verbindung zum PGLCP gibt neue und mächtige Werkzeuge um dieses Ziel weiter zu verfolgen. Leider reduziert sich das SSG im Allgemeinen auf PGLCPs mit Matrizen die zu keiner bekannten polynomiell lösbaren Klasse gehören.

Acknowledgments

My deepest gratitude goes to my advisor Bernd Gärtner who made my PhD possible. He was there whenever I needed advice and helped me out of many dead ends. All my papers are co-authored by him and without his broad knowledge, they would not have reached their quality. Thanks for all your support!

A big thank you goes to Emo Welzl for letting me be part of his research group and for his uncomplicated and upright manner, making the time at ETH unforgettable.

I thank my co-referees Hans-Jakob Lüthi for reviewing this thesis and sharing the passion for the LCP and Walter D. Morris for reviewing, supporting me over the years via e-mail and the joint paper.

The following people made it a pleasure to work at ETH:

Robert Berke, Yves Brise, Tobias Christ, Kaspar Fischer, Heidi Gebauer, Jochen Giesen, Franziska Hefti, Michael Hoffmann, Martin Jaggi, Shankar Lakshminarayanan, Andreas Meyer, Dieter Mitsche, Yoshio Okamoto, Andreas Razen, Dominik Scheder, Eva Schuberth, Ingo Schurr, Miloš Stojaković, Marek Sulovský, Tibor Szabó, Patrick Traxler, Floris Tschurr, Elias Vicari, Uli Wagner, Frans Wessendorp, Philipp Zumstein. Moreover all the members from the other research groups at our institute and, of course, the 22 red and blue guys on the H-floor.

Further a big thank you to the following people:

My co-authors Jirka Matoušek and Petr Škovroň for interesting meetings and the joint paper.

Markus Brill, Matúš Mihal'ák, Rahul Savani, Alex Souza-Oftermatt and the A-Team for an incredible time in Denmark.

Aniekan Ebiefung, Nir Halman, Klaus Simon, Roman Sznajder, Ola Svensson, Takeaki Uno, Bernhard von Stengel for inspiring discussions.

My family and all of my friends outside ETH for always being there.

My wife for loving me and my boys for tearing me away from the computer and trying to convince me that playing tag is more important than writing a thesis.

Contents

Chapter 1

Introduction

Against widespread belief, the abbreviation LCP does not stand for *Leo's Core Problem* but for the *Linear Complementarity Problem.* However, the former interpretation of the abbreviation is true for sure. This thesis collects results achieved during the author's PhD studies, all of which nicely connect to the LCP. It is based on three papers [33, 31, 34], their journal versions [32, 30] and unpublished material (for example Section 3.3 or Chapter 5).

The next section gives a detailed overview about our work and about what has been done and known before. It describes which of our results have been published where, with whom (all results are published together with Bernd Gärtner, so only different co-authors are mentioned below) and with what differences to the thesis at hand. The hurried reader looking for specific results might prefer the condensed overview given in Section 1.2.

1.1 Overview

Although special instances of the LCP first appear in a paper by Du Val already in 1940 [23], intensive analysis started in the mid 1960's where also the name *linear complementarity problem* originated. One of the earliest applications is that the first-order optimality conditions of a quadratic program can be written as an LCP [43, 1]. Besides countless other applications – a small but significant selection is for example given in [80] – there are also connections to game theory: bimatrix games allow an LCP formulation [15, 95]. We show in this thesis that other games like *simple stochastic games* can be reduced to the LCP as well.

The most comprehensive sources for the LCP are the books [17] by Cottle, Pang, and Stone and [67] by Murty. The LCP is given by linear equations, described by a square matrix $M \in \mathbb{R}^{n\times n}$ and a right-hand side vector $q \in \mathbb{R}^n$, and nonnegativity conditions as well as complementarity conditions. More concretely, given M and q, it is to find vectors $w \in \mathbb{R}^n$ and $z \in \mathbb{R}^n$ such that

$$\begin{aligned} w - Mz &= q \\ w, z &\geq 0 \\ w^T z &= 0, \end{aligned}$$

or to show that no such vectors exist.

Many relevant applications reduce to an LCP where M has special properties. A lot of research has been done on various matrix classes, a classical paper is [26] and a rich collection of results about matrix classes and their connections to the LCP can be found in [17]. We focus on the class of P-*matrices*. A matrix is a P-matrix if the determinants of all principal submatrices are positive. This class is interesting because it is known that the LCP has a unique solution for every q if and only if the matrix is a P-matrix [79]. Applications for the P-matrix LCP (PLCP) can be found for example in [7, 80, 77, 20].

It has been shown by Megiddo that hardness of the PLCP would imply that NP = co-NP [58]. Despite this fact, no polynomial algorithm is known up to date. Finding such an algorithm is one of the major goals of the LCP community since it would settle the complexity status of many problems reducing to the PLCP. One of the main LCP algorithms is the one from Cottle and Dantzig [15]. It is a *principal pivot algorithm*, pivoting from one basis of the LCP to the next according to a specified pivot-rule. The behavior of principal pivot algorithms has been investigated for many different pivot-rules, see [17] in general or [66, 27] for a nice appetizer.

Another important algorithm for the LCP is the one given by Lemke in [50]. Remarkably, the Lemke-Howson algorithm, which is a variant of Lemke's algorithm tailored for LCPs arising from bimatrix games, is an efficient constructive procedure for obtaining mixed equilibrium strategies for bimatrix games [51]. This constituted the simplest and most elegant constructive proof for the existence of Nash equilibria in bimatrix games (Nash's previous proof was based on the nonconstructive Brouwer fixed point theorem [68, 69]).

Besides the classical algorithms by Lemke and Cottle and Dantzig, there are still new ones being developed, see for example [38] for a more recent one. Interior point algorithms are also known, described for instance in [49]. Each of the above mentioned algorithms works for a larger class of matrices than P-matrices. See the references for discussions about termination depending on the matrix class.

We tackle the PLCP from two sides in this thesis. On the one hand, we show in the first part of this work that it fits into more general, easily structured frameworks. Although the generalizations are proper, and we therefore lose information, the simplicity of the frameworks allows for an improvement of algorithm running time. In fact, the fastest known algorithms solving PLCP were developed for these frameworks. On the other hand, we describe two specializations in the second part of this thesis. We first look at a subclass of P-matrices and examine the combinatorial structure behind the LCP associated with these matrices. The second specialization is actually a new application for the PLCP. We reduce a game theory problem to the PLCP. The relations between the frameworks and problem classes discussed in this thesis are depicted at the end of this section in Figure 1.2 on page 11.

Generalized linear complementarity problems. The first generalization we look at is the *generalized linear complementarity problem* with a P-matrix (PGLCP) in Chapter 2. The PGLCP was introduced by Cottle and Dantzig in [16]. It is more general than the PLCP since the matrix M is not restricted to be square. The matrix in the GLCP is a *vertical block matrix* to which the notion of P-matrix can be extended. The important properties of the PLCP generalize to the PGLCP: the GLCP has a unique solution for every right-hand side vector q if and only if its matrix is a P-matrix (see [16, 90, 39] or also [37]), and hardness of the PGLCP would imply NP = co-NP (this is easy to prove along the lines in [58]). In our paper [33, 32], we state the PGLCP and correlated results in a form dual to the one of Cottle and Dantzig. In this thesis, we mostly stick to the original setting of Cottle and Dantzig, but we explain the dual setting in Section 2.2, revealing connections to *linear programming* (LP). These connections are useful to derive results about the PGLCP with the special matrix class described in Chapter 5.

Unique sink orientations of grids. In Chapter 3 we then present our result from [33, 32], that the PGLCP fits into the framework of *unique sink orientations of grids.* A grid is a graph whose vertex set is the Cartesian product of finite sets, with edges joining all pairs of vertices that differ in exactly one component. Alternatively, we can view a grid as the skeleton (vertex-edge graph) of a specific polytope, namely a product of simplices. If all sets have size two, which is the case if we reduce the PLCP, we get the graph of a cube. A *face* or *subgrid* is any induced subgraph spanned by the Cartesian product of subsets of the original sets. An orientation ψ of the grid is called a unique sink orientation (USO) if every face has a unique sink with respect to ψ. USOs can in general be cyclic.

A polynomial-time algorithm for finding the sink of a grid USO (using an oracle that returns the orientation of a given edge) would solve the PGLCP in *strongly* polynomial time. Candidates for strongly polynomial algorithms must be *combinatorial* in the sense that the number of arithmetic operations they perform depends only on the combinatorial structure of the PGLCP but not on the actual numbers that encode it.

Most of the papers dealing with PLCP or PGLCP analyze the problem from an algebraic point of view. With the USO approach we shed more light on the combinatorial structure of the PGLCP. First efforts in this direction have been made by Stickney and Watson [87], who considered orientations of cubes as digraph models for PLCP. Although the name USO was formed only later in [89], their orientations are in fact USOs. Stickney and Watson give an example of a PLCP resulting in a cyclic 3-dimensional cube USO. Morris constructed highly cyclic PLCP-induced USOs in higher dimensions for which the algorithm following a random outgoing edge at every cube-vertex performs very badly [64]. Most remarkably, Szabó and Welzl gave algorithms for finding the sink of any n-cube USO by looking at only $O(c^n)$ vertices and edges, for some c strictly smaller than 2 [89]. This in particular yields the first combinatorial algorithms for PLCP with nontrivial runtime bounds. For acyclic cube USO, Gärtner gives a subexponential algorithm in [28]. Cube USO are useful as combinatorial models for many other problems, see [89, 33, 32] and references therein for more detailed discussions.

The generalization from PGLCP to grid USO reveals some (algorithmically useful) hidden structure, leading to new results for PGLCP. Since most of these results were already derived in the author's Master thesis [78], we present only the actual reduction from PGLCP to grid USO in Chapter 3. In [33, 32] we presented the result that PGLCP-induced grid USOs satisfy the *Holt-Klee* condition. The Holt-Klee condition is a combinatorial property shared by a diminishing fraction of all USO [22]. The fraction of PGLCP-induced grid USOs is even smaller, since there are Holt-Klee grid USOs that do not come from PGLCP [63].

The problem of finding an optimal sink-finding algorithm can be stated as a zero-sum game which in turn can be solved by linear programming [29]. Unfortunately, the LP gets too big already for small grid USOs. In the last section of Chapter 3, Section 3.3, we analyze a subclass of 2-dimensional grids called *ladders*. Ladders are the simplest grid USO for which we don't know optimal algorithms. For small enough ladders, by making use of isomorphisms as described in [91], the LP whose solution encodes an optimal algorithm is solvable in reasonable time. We describe optimal algorithms derived this way for small ladders. These give us an intuition how optimal algorithms might behave on general ladders. At the end of Chapter 3, we give an almost optimal algorithm for ladders satisfying the Holt-Klee condition. Our findings about ladders have not been published.

Violator spaces. A further result, motivated by our ambition to analyze the combinatorics behind the PGLCP, is the generalization from grid USOs to *violator spaces* in Chapter 4. Violator spaces are proper generalizations of LP-*type problems*. The framework of LP-type problems, invented by Sharir and Welzl in 1992 [84], was used to show that LP is solvable in subexponential time in the RAM model (independent of the precision of the input numbers) [57].

Violator spaces were introduced by Jirka Matoušek and Petr Škovroň in [85]. In this thesis, we present results derived together with them in our joint paper [31, 30]. The reason we look at violator spaces is that LP-type problems have an acyclic structure, which prevents general grid USOs (in particular PGLCP-induced cyclic ones) to fit into this framework. To our knowledge, violator spaces are, besides oriented matroid programs (see for example [6]), the only abstract optimization framework allowing cycles. We prove that grid USOs, and therefore PGLCPs, are subsumed by violator spaces.

Section 4.3 shows that Clarkson's randomized algorithms [11], developed for low-dimensional LP (they are also applicable to LP-type problems, see [35, 9]) work in the context of violator spaces. These algorithms give us an optimal linear time algorithm for the PGLCP with a constant number of blocks. More results about violator spaces can be found in [85, 86, 31, 30], for example that LP-type problems and *acyclic* violator spaces are equivalent.

Hidden K-matrices. Chapter 5 is dedicated to the PGLCP with *hidden* K-*matrices*, a proper subclass of P-matrices that also appears under the name *hidden Minkowski matrices* or *mime matrices* (see [93]). Hidden K-matrices show up in real world problems, for example the problem of pricing American put options can be solved with the help of a hidden K-matrix LCP [7] (more precisely, the matrix belongs to the even smaller subclass of K-*matrices*). The theory behind the hidden K-matrix GLCP was founded by Mangasarian in his papers [52, 53, 54, 55]. Mangasarian analyzed GLCPs that can be restated as linear programs. Cottle and Pang extended research in this direction [19, 18, 72, 71, 70], finally ending up with the main result that the hidden K-matrix LCP can be solved by linear programming and therefore in time polynomial in the input size [47] (see also the strongly polynomial time algorithms for matrices whose transpose is hidden K [73, 62]). This was generalized to the GLCP by Mohan and Neogy in [61].

We strengthen the hidden K theory by showing that the grid USO we get from the hidden K-matrix GLCP is an orientation we get from an LP. The USO is therefore acyclic for any right-hand side vector q. In an attempt to characterize those non hidden K but P-matrices that yield a cyclic orientation for some q (we call such a q a *cyclic* q), we first derive a characterization for matrices that are not hidden K. Our characterization generalizes the characterization of [65] to the GLCP case. We then introduce a new nontrivial and large subclass of non hidden K-matrices and prove that 3-dimensional P-matrices in this class have a cyclic q. However, we fail to prove this for higher dimensions. It thus remains an open problem to characterize those P-matrices having a cyclic q. None of our results about the hidden K-matrix GLCP has been published yet.

Simple stochastic games. The last chapter of the thesis is about *simple stochastic games*. Simple stochastic games (SSG) form a subclass of general stochastic games, introduced by Shapley in 1953 [83]. Condon was first to study the complexity-theoretic aspects of SSG [12]. She showed that the decision version of the problem is in NP $\cap$ co-NP. This is considered as evidence that the problem is not NP-complete, because the existence of an NP-complete problem in NP $\cap$ co-NP would imply NP = co-NP. Despite this evidence and a lot of research, the question whether a polynomial time algorithm exists remains open. This reminds us of the PGLCP, whose hardness would also imply NP = co-NP and for which no polynomial time algorithm is known. Indeed, we present in Chapter 6 our result from [34] that SSG can be reduced to PGLCP.

A SSG is played by moving a token on a directed graph whose vertex set consists of two sinks, the **0**-sink and the **1**-sink, and the rest partitioned into three parts, vertices belonging to the max player, min player, and *average* vertices, respectively. At a vertex, the player it belongs to chooses along which outgoing edge to move the token. The goal of the max player is to (maximize the probability to) reach the **1**-sink, the goal of the min player to reach the **0**-sink. We consider SSG with vertices of arbitrary outdegree and with average vertices determining the next vertex according to an arbitrary probability distribution. This is a natural generalization of *binary* SSG introduced by Condon [12]. As a specialization of the results in Chapter 6 we get that a binary SSG reduces to the PLCP.

SSG are significant because they allow polynomial-time reductions from other interesting classes of games. Zwick and Paterson proved a reduction from *mean payoff games* [98] which in turn admit a reduction from *parity games*, a result of Puri [75]. See the references for applications of these games.

A *strategy* of a player determines for every vertex belonging to the player along which outgoing edge to move the token. Given a strategy of one player, the optimal counterstrategy of the other player can be computed by a linear program [21]. Using this, Halman could show that the problem of finding an optimal strategy for one of the players can be stated as an LP-type problem [41] (see also [42, 40]) and can therefore be solved in strongly subexponential time.

Independently, Björklund et al. arrived at subexponential methods by showing that SSG can be mapped to the combinatorial problem of optimizing a *completely local-global* function over the Cartesian product of sets [3]. This setup is the same as the setup of *acyclic* grid USOs, which can in turn be formulated as LP-type problems. In fact, they find the sink of the acyclic USO by applying LP-type algorithms, so their approach is actually equivalent to Halman's.

The methods of Halman and Björklund et al. focus on the strategy of one player while recomputing the optimal counterstrategy of the other player in every step. By solving the SSG via PGLCP we do not distinguish between the two players. This is best explained in the USO world. An algorithm following an outgoing edge in a grid USO induced by an SSG via the PGLCP formulation improves the strategy of the player the “edge belongs to”. The dimension of the grid is the number of player vertices in the SSG and an edge along dimension i in the grid is associated with player vertex i. Following one edge in the USO setting means to *switch* the strategy of one player at one vertex in the game. Condon [13] (see also the example in Subsection 6.3.1) shows that switching algorithms can cycle, so the underlying USO can be cyclic. This is not the case in the setting of Björklund et al. Their USO is with respect to one player, the max player, say, so the dimension of the grid is the number of vertices of the max player. Every vertex in the grid corresponds to a strategy of the max player, and a grid-vertex evaluation, returning the orientations of incident edges, is achieved by computing the optimal counterstrategy of the min player. An edge in the grid is outgoing, if the strategy at the neighboring grid-vertex is better for the max player. This implies that the grid USO is acyclic.

In [78, 33] we introduced the concept of *projected* (or *inherited*) grid USOs. Roughly, a projected grid USO is derived by merging sets of vertices along a subset of the grid's dimensions into single vertices and taking the outgoing edges of the sink in the set of original vertices merged into one to be the outgoing edges of the resulting vertex. One can show that this is again a (smaller dimensional) grid USO. See Fig-

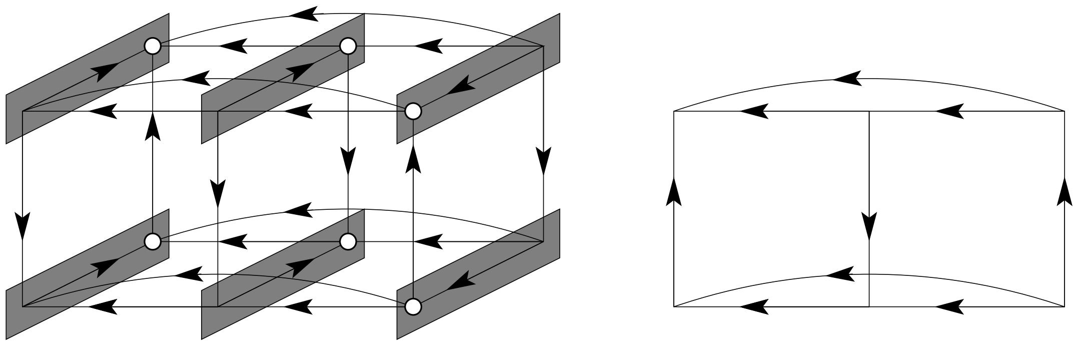


Figure 1.1: A 3-dimensional grid USO and a projected grid USO of it derived by merging vertices along one dimension.

ure 1.1 for an example. If, in the grid USO derived from an SSG via PGLCP, all dimensions corresponding to one player are merged, then the resulting grid USO is exactly the one from Björklund et al. and therefore acyclic. This is a very special property, which we don't expect from general PGLCP-induced USO. This is the reason why we think that PGLCP is more general than SSG, meaning that it is not possible to get every P-matrix in the reduction from SSG to PGLCP.

Nevertheless, by looking at a superclass of SSG, derived from SSG by adding payoff-values to the edges, our reduction may yield *any* right-hand side vector q in the PGLCP (changing the payoffs changes the q-vector but does not affect the matrix). Shapley's stochastic games still contain this superclass of SSG and Shapley's theorem [83] proving uniqueness of *game values* then implies that the GLCP has a unique solution for every q and its matrix must therefore be a P-matrix. Our result that the matrix in the reduction is a P-matrix thus provides an alternative proof of Shapley's theorem, specialized to SSG, and it makes the connection to matrix theory explicit.

There are interesting connections between algorithms in the USO setting and algorithms for the SSG. The algorithm *bottom-antipodal* in the USO world, for example, being at vertex v jumps to the vertex antipodal in the face spanned by all the outgoing edges of v. There are exponential lower bounds for the performance of this algorithm [82]. We can interpret the behavior of bottom-antipodal in SSG. In the setting of Björklund et al. it means that given a strategy for the max player, simultaneously switch the strategy at every max vertex that has a switch improving the strategy. Then compute the optimal counterstrategy of the min player and proceed as before. This algorithm is a variant of the *Hoffman-Karp algorithm* for SSG [44, 13].

The fact that there is a connection between games and LCP is not entirely surprising, since, as noted in the beginning of this section, for example bimatrix games can be formulated as LCP [15, 95]. Also Cottle, Pang and Stone [17, Section 1.2] list a simple game on Markov chains as an application for LCP, and certain (very easy) SSG are actually of the type considered. Björklund et al. describe a reduction from games to what they call *controlled linear programming* [4]; controlled linear programs are easily mapped to (non-standard) LCP. Independently from our work, Björklund et al. have made this mapping explicit by deriving LCP-formulations for mean payoff games [5]. Their reduction is very similar to ours, but the authors do not prove that the resulting matrices are P-matrices, or belong to some other known class. In fact, Björklund et al. point out that the matrices they get are in general not P-matrices, and this stops them from further investigating the issue. We have a similar phenomenon here: applying our reduction to *non-stopping* SSG, we may also obtain matrices that are not P-matrices. The fact that comes to our rescue is that the stopping assumption incurs no loss of generality and, in contrast to our paper [34], we make heavy use of it in this thesis, simplifying the reduction a lot. For mean payoff games, a similar result holds. They can without loss of generality be transformed into *discounted* mean payoff games [98]. Jurdziński and Savani [45] could prove that reducing discounted mean payoff games results in LCP with a P-matrix.

Our result that SSG-induced GLCPs come with P-matrices puts SSG into the realm of 'well-behaved' GLCP, but it does not give improved runtime bounds. Unfortunately, the matrices we get from SSG do in general not belong to a class known to be polynomial-time solvable, see Section 6.3. Still, properties of the subclass of matrices we ask for might allow their PGLCP to be solved in polynomial time. By, without loss of generality, simplifying the graph underlying the SSG first, Svensson and Vorobyov managed to reduce SSG to PGLCP with a very simply structured matrix [88]. However, no progress has been made so far in algorithmically exploiting this structure.

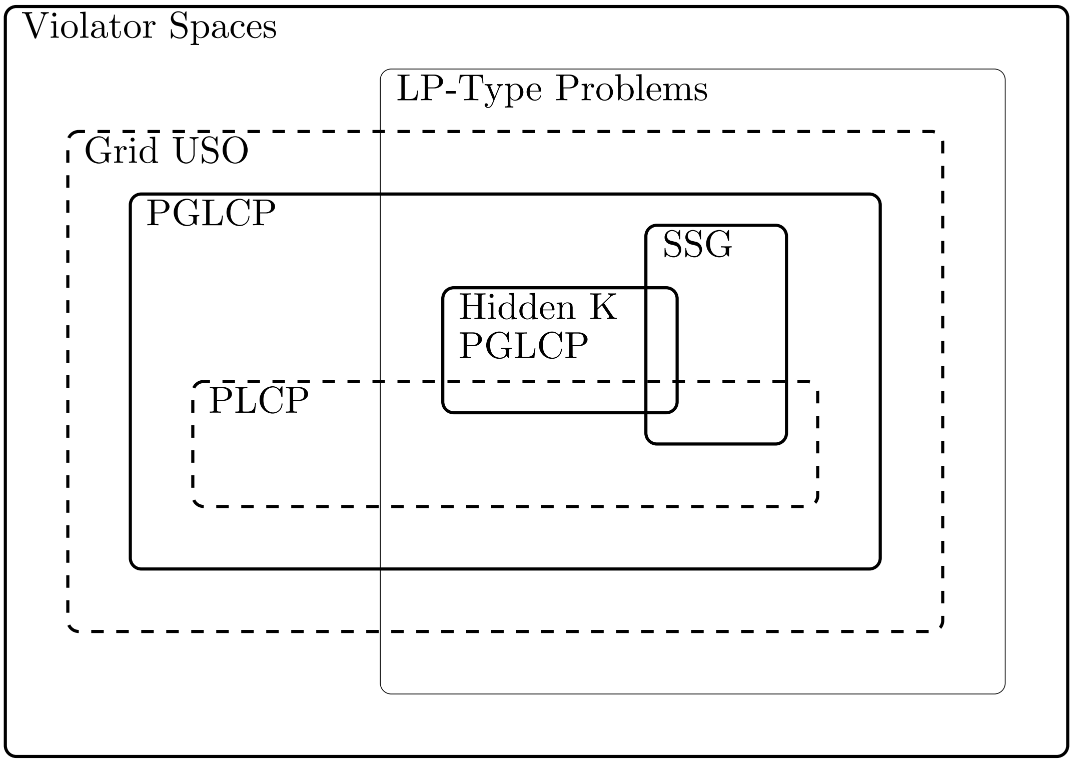


Figure 1.2: An overview of the classes used in this thesis.

1.2 Short Outline of the Thesis

In the next section, we define the linear complementarity problem and introduce the necessary notations. After that, the thesis is split into two parts. In the first part we present three generalizations of the PLCP, the first being the P-matrix *generalized linear complementarity problem* (PGLCP) in Chapter 2. Most of the chapter is consumed by definitions and setting notation, but we also provide a dual view of the PGLCP which we introduced in our paper together with Walter D. Morris [33, 32], making visible connections to linear programming.

The second generalization described in Chapter 3 is that of *unique sink orientations* (USO). We show, in a different setting than the one we used in [33, 32], how the PGLCP reduces to finding the sink of a USO on a grid graph. Moreover, we review the *Holt-Klee condition*, a combinatorial property shown to hold for PGLCP-induced grid USO in [33, 32]. The chapter ends with a detailed presentation of unpublished results about special instances of grid USOs, which we call *ladders*.

These are the simplest grid USOs for which we don't know the optimal algorithms. We give an almost optimal algorithm to find the sink in a Holt-Klee ladder.

The third generalization is that of *violator spaces*. Violator spaces form a framework subsuming LP-*type problems*, a framework invented by Sharir and Welzl [84]. We show in Chapter 4 that grid USOs (and therefore the PGLCP) are models for violator spaces but not for LP-type problems in general. Moreover, Clarkson's algorithms, that have been designed for low dimensional linear programs and later adapted for LP-type problems, are shown to work for violator spaces. This yields an optimal linear time algorithm for solving PGLCP with a fixed number of blocks. These results, together with some structural results about violator spaces, are published in [31, 30]. This is joint work with Jirka Matoušek and Petr Škovroň.

In the second part we look at specializations of the PLCP. In Chapter 5 we consider LCPs associated with a subclass of P-matrices: *hidden* K-*matrices*. Altough being in the specialization part here, we look at the hidden K-matrix *generalized* LCP whenever possible. The main achievement is a proof that the USO arising from a hidden K-matrix GLCP is LP-induced and therefore always acyclic. Additionally, we attempt to characterize those matrices from which a cyclic USO arises and succeed to define a nontrivial and large subclass of 3-dimensional non hidden K-matrices that give rise to a cycle. The material in this chapter has not been published yet.

The final Chapter 6 extends our results from [34]. Using a simpler setting than in the paper, we show how *simple stochastic games* can be reduced to the PGLCP. This makes the whole PGLCP machinery available for games. We further give some negative results, for example that the resulting matrix in the PGLCP is not hidden K in general.

1.3 The P-Matrix Linear Complementarity Problem

Given a matrix $M \in \mathbb{R}^{n\times n}$ and a vector $q \in \mathbb{R}^n$, the *linear complementarity problem* (LCP) is to find a vector $z \in \mathbb{R}^n$ such that

$$\begin{array}{rcl} z & \geq & 0 \\ q + Mz & \geq & 0 \\ z^T(q+Mz) & = & 0 \end{array} \tag{1.1}$$

or to show that no such vector exists. In this thesis, we use an often encountered equivalent formulation of the LCP, that will simplify our analysis in later chapters: given a matrix $M \in \mathbb{R}^{n\times n}$ and a vector $q \in \mathbb{R}^n$, the LCP is to find vectors $w \in \mathbb{R}^n$ and $z \in \mathbb{R}^n$ such that

$$\begin{array}{rcl} w - Mz & = & q \\ w, z & \geq & 0 \\ w^T z & = & 0 \end{array} \tag{1.2}$$

or to show that no such vectors exist. We refer to (1.2) as LCP(M, q). Note that the nonnegativity conditions $w, z \geq 0$ together with the complementarity condition $w^T z = 0$ force at least one of the variables w_i and z_i to be zero for all $i \in \{1, \ldots, n\}$.

We are interested in the LCP(M, q) where M is a P-*matrix.*

Definition 1.3 *A matrix $M \in \mathbb{R}^{n\times n}$ is a* P-matrix *if the determinants of all principal submatrices are positive.*

We refer to determinants of principal submatrices as *principal minors.* The relevance of the LCP with a P-matrix, abbreviated as PLCP, stems from the following theorem, first proved in [79].

Theorem 1.4 *The* LCP*(M, q) has a unique solution for all vectors $q \in \mathbb{R}^n$ if and only if $M \in \mathbb{R}^{n\times n}$ is a* P*-matrix.*

Beauty and simplicity of the PLCP are best explained in the geometric view. In order to be able to do this, we first need some notation. We denote by $[n]$ the set of integers from 1 to n. Given matrix $A \in \mathbb{R}^{n\times n}$

and a set $\alpha \subseteq [n]$, the matrix $A_\alpha \in \mathbb{R}^{n\times|\alpha|}$ is derived by deleting those columns from A whose indices are not in α. For readability, we let $A_{\bar{\alpha}} := A_{[n]\setminus\alpha}$. The same notation is applied to n-vectors, with the obvious meaning. Moreover, for $B \in \mathbb{R}^{n\times n}$ we define $(A_\alpha \| B_{\bar{\alpha}}) \in \mathbb{R}^{n\times n}$ to be the matrix whose ith column is $A_{\{i\}}$ if $i \in \alpha$ and $B_{\{i\}}$ otherwise. The same definition analogously holds for vectors, so given $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^n$, the ith component of $\binom{a_\alpha}{b_{\bar{\alpha}}}$ is a_i if $i \in \alpha$ and b_i otherwise. To make the reader more familiar with this notation, we give an example for $n = 3$. Let

$$A := \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \qquad B := \begin{pmatrix} 4 & 5 & 6 \\ 4 & 5 & 6 \\ 4 & 5 & 6 \end{pmatrix},$$

and

$$a := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \qquad b := \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix}.$$

Then

$$(A_{\{1,3\}} \| B_{\{2\}}) = \begin{pmatrix} 1 & 5 & 3 \\ 1 & 5 & 3 \\ 1 & 5 & 3 \end{pmatrix} \text{ and } \begin{pmatrix} a_{\{1,3\}} \\ b_{\{2\}} \end{pmatrix} = \begin{pmatrix} 1 \\ 5 \\ 3 \end{pmatrix}.$$

There are 2^n possibilities to satisfy the complementarity condition in the PLCP, achieved by setting either w_i or z_i to zero in each coordinate i. As soon as we fix the complementary set of variables that should be zero, say the variables $\binom{w_{\bar{\alpha}}}{z_\alpha}$ for some α, then the values of the remaining variables $\binom{w_\alpha}{z_{\bar{\alpha}}}$ are uniquely determined: pre-multiplying the PLCP with $(I_\alpha \| - M_{\bar{\alpha}})^{-1}$ (note that existence of the inverse easily follows from the fact that M is a P-matrix) results in

$$\begin{pmatrix} w_\alpha \\ z_{\bar{\alpha}} \end{pmatrix} - M' \begin{pmatrix} w_{\bar{\alpha}} \\ z_\alpha \end{pmatrix} = (I_\alpha \| - M_{\bar{\alpha}})^{-1} q$$

for $M' := -(I_\alpha \| {-M_{\bar{\alpha}}})^{-1}(I_{\bar{\alpha}} \| {-M_\alpha})$. Since $\binom{w_{\bar{\alpha}}}{z_\alpha}$ are the zero variables, we get $\binom{w_\alpha}{z_{\bar{\alpha}}} = (I_\alpha \| - M_{\bar{\alpha}})^{-1} q$.

Definition 1.5 *A subset $\alpha \subseteq [n]$ is called a* basis *of the* PLCP *(1.2), and*

$$B(\alpha) := (I_\alpha \| - M_{\bar{\alpha}})$$

is the corresponding basis matrix.

A basis α determines that the variables $w_{\bar{\alpha}}$ and z_α are set to zero. Taking into account the nonnegativity constraints for w and z, solving the PLCP is equivalent to finding a basis $\alpha \subseteq [n]$ for which $\binom{w_\alpha}{z_{\bar{\alpha}}} = B(\alpha)^{-1}q \geq 0$. We often say that such an α is a solution to the PLCP, since w and z are derived immediately from it.

The procedure of computing $M' = -B(\alpha)^{-1}B(\bar{\alpha})$ from M as above is known as a *principal pivot transform*, abbreviated as PPT. Tucker first proved that a matrix derived from a P-matrix via a PPT is again a P-matrix [94]. A PPT corresponds to rewriting the PLCP with variables w_i and z_i interchanged for some indices $i \in [n]$.

This thesis deals with *nondegenerate* PLCPs only. A PLCP is *degenerate* if $B(\alpha)^{-1}q$ has a zero entry for some α and it is nondegenerate otherwise. Nondegeneracy can easily be achieved by slightly perturbing the vector q, such that for all α, $B(\alpha)^{-1}q$ has no zero entries. For a solution α, this implies that all nonzero variables are positive, $B(\alpha)^{-1}q > 0$. Moreover, there is a *unique* set α for which $B(\alpha)^{-1}q > 0$, since two different sets fulfilling the condition would imply two different solutions (because different variables are nonzero in the two solutions) to the PLCP, contradicting Theorem 1.4. We come back to the nondegeneracy issue in the geometric view of the PLCP which we describe now.

The 2^n basis matrices of the PLCP can be interpreted as n-dimensional cones in $\mathbb{R}^n$, called *complementary cones*. For any α, the complementary cone associated with $B(\alpha)$ is spanned by the n vectors corresponding to the columns of $B(\alpha)$ (since $B(\alpha)$ is non-singular, those columns are linearly independent). Finding the unique α such that $B(\alpha)^{-1}q > 0$ is then to find the unique complementary cone which contains the vector q. Since the PLCP has a unique solution for all q (Theorem 1.4), the complementary cones cover the whole space. Moreover, they intersect only in lower dimensional cones, since otherwise, a q lying in the interior of a full dimensional intersection would be contained in two distinct complementary cones (contradicting the uniqueness of

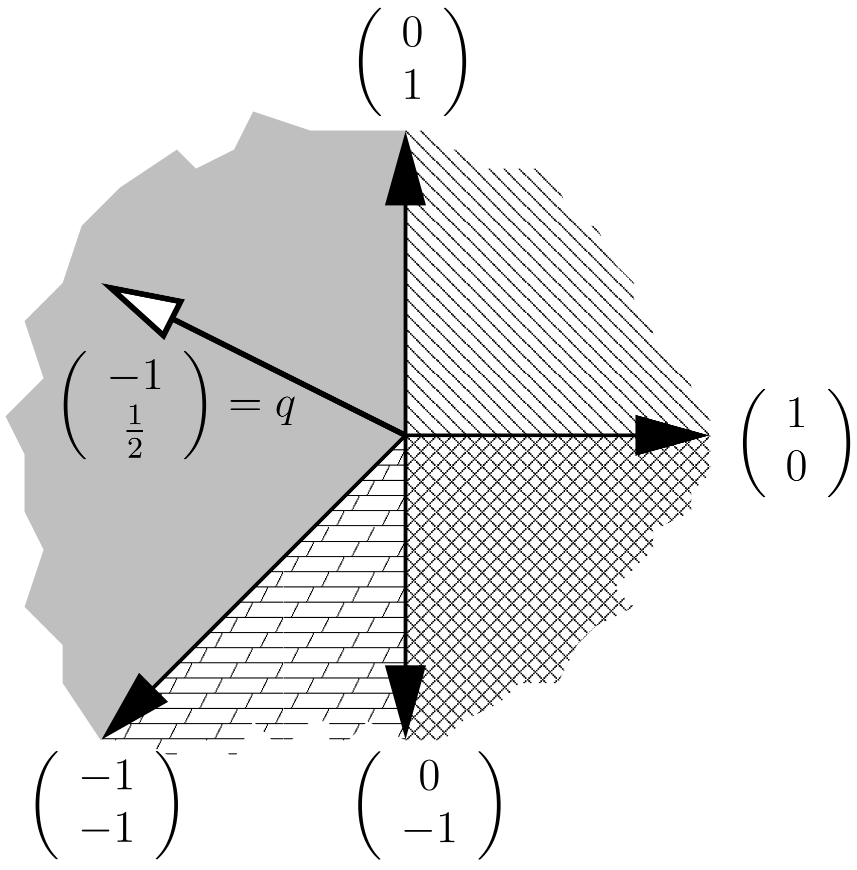

$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\begin{pmatrix} -1 \\ \frac{1}{2} \end{pmatrix} = q$
$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\begin{pmatrix} -1 \\ -1 \end{pmatrix}$
$\begin{pmatrix} 0 \\ -1 \end{pmatrix}$

Figure 1.3: Geometric view of the PLCP (1.6).

the basis α with $B(\alpha)^{-1}q > 0$). As an example for the geometric view, look at the PLCP

$$\begin{aligned} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} &= \begin{pmatrix} -1 \\ \frac{1}{2} \end{pmatrix} \\ w_1, w_2, z_1, z_2 &\geq 0 \\ w_1 \cdot z_1 + w_2 \cdot z_2 &= 0 \end{aligned} \tag{1.6}$$

whose geometric view is given in Figure 1.3. From the picture we see that q lies in the complementary cone spanned by the columns of the basis matrix $B(\{2\})$. The variables w_1 and z_2 are therefore zero and the positive values of the variables z_1 and w_2 can be computed as

$$\begin{pmatrix} w_{\{2\}} \\ z_{\{1\}} \end{pmatrix} = \begin{pmatrix} z_1 \\ w_2 \end{pmatrix} = B(\{2\})^{-1}q = \begin{pmatrix} -1 & 0 \\ -1 & 1 \end{pmatrix}^{-1}\begin{pmatrix} -1 \\ \frac{1}{2} \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{3}{2} \end{pmatrix}.$$

If q is contained in a hyperplane that contains the $(n-1)$-dimensional intersection of two complementary cones, then the PLCP is degenerate. Let $B(\alpha)$ and $B(\alpha')$ correspond to two complementary cones whose $(n-1)$-dimensional intersection is contained in a hyperplane containing q as well. Since the intersection of the two cones is $(n-1)$-dimensional, α and α' differ only in one element, i.e., their symmetric difference is 1: $|\alpha \oplus \alpha'| = |\{i\}| = 1$. Although the solution to the PLCP is still unique, the complementary cone containing q might not be unique since $B(\alpha)^{-1}q$ and $B(\alpha')^{-1}q$ are zero at coordinate i and possibly positive in all other coordinates. In order to get rid of such cases, we perturb q a little bit, i.e., we move it slightly such that it is no more contained in any hyperplane containing the intersection of two complementary cones. This affects the solution to the PLCP in a controllable way (variables that are zero in the original setting can have arbitrarily small solution values in the perturbed setting) and for the rest of this thesis we therefore stick to the assumption of nondegeneracy, i.e., $(B(\alpha)^{-1}q)_i \neq 0$ for all α and i. In particular, $B(\alpha)^{-1}q > 0$ for the solution α.

Part I: Generalizations

In this first part of the thesis we present three generalizations of the PLCP, the first being the *generalized linear complementarity problem* (PGLCP) in Chapter 2. We define the notation needed for the PGLCP and also state the problem in the dual setting we used in [33, 32]. All further results, if possible, are then stated for the PGLCP.

Chapter 3 shows how PGLCP can be reduced to *unique sink orientations* (USO) of grids. In [33, 32] we showed that these orientations fulfill a well-known combinatorial property, the *Holt-Klee condition.* At the end of this chapter, the Holt-Klee condition is useful in the analysis of algorithms for the smallest class of grid USOs for which no optimal algorithms are known.

The last chapter in this part introduces *violator spaces* as a further generalization of grid USOs (and therefore PGLCP). Violator spaces form a simple combinatorial framework, and our result that Clarkson's algorithms work for them yields an optimal linear time algorithm for solving PGLCPs of fixed dimension.

Chapter 2

The P-Matrix Generalized LCP

The *generalized linear complementarity problem*, abbreviated as GLCP, was introduced by Cottle and Dantzig in 1970 [16]. It generalizes the LCP(M, q) by dropping the requirement of M being a square matrix.

2.1 The Setup

In order to state the GLCP, we first define the type of a matrix.

Definition 2.1 *A matrix G is a* vertical block matrix of type $(g_1, \ldots, g_n)$ *if it is of the form*

$$G = \begin{bmatrix} G^1 \\ \vdots \\ G^n \end{bmatrix}$$

where the ith block G^i, $i \in [n]$, has order $g_i \times n$.

For reasons that will become clear in the next chapter, we define N to be

$$N := \sum_{i=1}^{n} g_i + n,$$

i.e., G is an $(N-n)\times n$ matrix. The definition of a vertical block matrix applies in a straightforward way also to vectors.

Given a vertical block matrix $G \in \mathbb{R}^{(N-n)\times n}$ and a vertical block vector $q \in \mathbb{R}^{N-n}$, both of type $(g_1,\ldots,g_n)$, the GLCP is to find a vertical block vector $w \in \mathbb{R}^{N-n}$ of type $(g_1,\ldots,g_n)$ and a vector $z \in \mathbb{R}^n$ such that

$$\begin{array}{rcll} w - Gz & = & q & \\ w, z & \geq & 0 & \\ z_i \prod_{j=1}^{g_i} w_j^i & = & 0, & \text{for all } i \in [n]. \end{array} \tag{2.2}$$

In analogy to G^i, w^i is the ith block of size g_i of the vector w. In the GLCP, complementarity holds block-wise, i.e., either z_i is zero or at least one variable in the block w^i. We refer to (2.2) as GLCP(G,q).

Definition 2.3 *A* representative submatrix $\bar{G} \in \mathbb{R}^{n\times n}$ *of G is derived by letting the ith row of $\bar{G}$ be one row out of the block G^i for all $i \in [n]$.*

There are $\prod_{i=1}^{n} g_i$ different representative submatrices of G. With their help, the P-matrix notion can be generalized to vertical block matrices.

Definition 2.4 *A vertical block matrix $G \in \mathbb{R}^{(N-n)\times n}$ is a* vertical block P-matrix *if all principal minors of all representative submatrices are positive.*

So, every (square) representative submatrix of a vertical block P-matrix is itself a P-matrix (as defined in Definition 1.3). We sometimes omit the words "vertical block" when it is clear from the context what kind of matrix we mean. Cottle and Dantzig show that a solution to the P-matrix GLCP always exists [16], and Szanc shows in his dissertation that the solution is unique for all q [90, 39], so Theorem 1.4 generalizes to the GLCP.

Theorem 2.5 *The* GLCP(G, q) *has a unique solution for all vectors* $q \in \mathbb{R}^{N-n}$ *if and only if* $G \in \mathbb{R}^{(N-n)\times n}$ *is a* P-*matrix.*

We abbreviate the P-matrix GLCP by PGLCP. For our analysis, it will be easier to restate the PGLCP in the following form. Let I be the identity matrix in $\mathbb{R}^{(N-n)\times(N-n)}$ and divide it into n blocks of columns with the ith block having size $(N-n)\times g_i$. Now expand the ith block by appending $-G_{\{i\}}$, the negated ith column of G. The resulting matrix H[1] is of order $(N-n)\times N$ and consists of n blocks of columns with the ith block having size $(N-n)\times(g_i+1)$. The first g_i columns of block i are the ones from I and the last column is $-G_{\{i\}}$. The same is done with the vectors w and z. A new vector $x \in \mathbb{R}^N$ is formed from w by appending z_i to the bottom of w^i. The first g_i components of block x^i are therefore the components of w^i and the last component of x^i is z_i. Finally, in order to simplify notation, we set $h_i := g_i + 1$ for all i and rewrite the PGLCP (2.2) as

$$\begin{array}{rcll} Hx & = & q & \\ x & \geq & 0 & \\ \prod_{j=1}^{h_i} x_j^i & = & 0, & \text{for all } i \in [n]. \end{array} \tag{2.6}$$

We refer to this setting as PGLCP(H, q). In this form, it is easier to define bases for the PGLCP. The partition of H into n blocks of columns corresponds to a partition Π of $[N]$ into n subsets Π_i of size h_i each,

$$\Pi = (\Pi_1, \ldots, \Pi_n).$$

Let $\beta \subseteq [N]$ be an n-element set consisting of one element out of each Π_i, $i = 1, \ldots, n$. Such a set is called *representative* and the element in β belonging to Π_i is denoted by β_i. There are $\prod_{i=1}^n h_i$ many representative sets β. To shortcut notation, we set $\bar{\beta} := [N] \setminus \beta$. Then, $H_{\bar{\beta}} \in \mathbb{R}^{(N-n)\times(N-n)}$ is the matrix H restricted to columns whose indices are not in β.

Definition 2.7 *A representative subset* $\beta \in [N]$ *is called a* basis *of the* PGLCP(H, q), *and*

$$B(\beta) := H_{\bar{\beta}}$$

[1]Since the matrix is a mixture of $-G$ and I, it seems appropriate to choose the letter which lies between G and I in the alphabet.

is the corresponding basis matrix *in* $\mathbb{R}^{(N-n)\times(N-n)}$.

We also define $x_{\bar{\beta}}$ to be the vector x restricted to elements with indices not in β. More precisely, the ith block of $x_{\bar{\beta}}$ is the ith block of x reduced by the element with index β_i in x.

The elements of β correspond to the variables in x that are set to zero such that complementarity is fulfilled. Once β is fixed, the remaining variables are determined, since the PGLCP(H, q) can be pre-multiplied by $B(\beta)^{-1}$ (nonsingularity of $B(\beta)$ comes out of G's P-matrix property):

$$B(\beta)^{-1}Hx = H'x = B(\beta)^{-1}q$$

with $H'_{\bar{\beta}}$ being the identity matrix in $\mathbb{R}^{(N-n)\times(N-n)}$ and H'_{β} being a vertical block matrix $-G'$. Setting x_β to zero then yields $x_{\bar{\beta}} = B(\beta)^{-1}q$.

The matrix G' is derived from the original matrix G via a principal pivot transform. Habetler and Szanc generalized Tucker's result that square P-matrices are closed under PPT to vertical block P-matrices [39]. The matrix G' is thus a vertical block P-matrix.

As in the PLCP case, solving the PGLCP amounts to finding the β for which $B(\beta)^{-1}q > 0$, where we again assume that the PGLCP is nondegenerate. Note that the geometric point of view can also be taken in the PGLCP. Any basis matrix can be interpreted as an $(N-n)$-dimensional complementary cone in $\mathbb{R}^{(N-n)}$ and Theorem 2.5 makes sure that these cones cover the whole space and intersect only in lower dimensional cones. The same considerations about nondegeneracy as for the PLCP apply, and we therefore assume for the rest of the thesis nondegeneracy of the PGLCP.

The cone point of view will be important in Section 3.2, where we state the result derived in [33, 32] that the orientation implicitly underlying a PGLCP (described in the next section) satisfies a special property. But first we devote a section to the description of the PGLCP in the dual setting we used in [33, 32], revealing connections to the *linear programming problem.*

2.2 Π-Compatible Linear Programming

Consider a linear program (LP) in the variables $x = (x_1, \ldots, x_N)^T$, of the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \geq 0, \end{array} \tag{2.8}$$

where $A \in \mathbb{R}^{n \times N}, b \in \mathbb{R}^n$ and $c \in \mathbb{R}^N$. The index set $[N]$ of A's columns is partitioned into n blocks by $\Pi = (\Pi_1, \ldots, \Pi_n)$. As in the previous section, let β be a representative set consisting of exactly one element per block Π_i. Assume that for all β, A_β is a nondegenerate *basis matrix* in (2.8), meaning that A_β is invertible and $A_\beta^{-1} b > 0$. We say that the LP is Π-*compatible*, and we call A_β a *representative submatrix*. We will assume that the ordering of the columns in A_β is compatible with Π, meaning that the i-th column of A_β comes from $A_{\Pi_i}, i \in [n]$.

Using Cramer's rule, the following is not hard to establish.

Observation 2.9 *Consider a* Π-*compatible* LP *of the form (2.8). Then*

(i) all determinants $\det(A_\beta)$ *of representative submatrices have the same (nonzero) sign, and*

(ii) the A_β *are the only basis matrices of the* LP *(2.8).*

Proof. Let the $n \times n$ basis matrices A_β and $A_{\beta'}$ differ in one column, column i. Then, by Cramer's rule, the ith component of the solution to $A_\beta x_\beta = b$ and $A_{\beta'} x_{\beta'} = b$ respectively, is given by

$$(x_\beta)_i = \frac{\det(A^i_\beta)}{\det(A_\beta)}, \qquad (x_{\beta'})_i = \frac{\det(A^i_{\beta'})}{\det(A_{\beta'})},$$

where the matrix A^i_β is A_β with the ith column replaced by the vector b which is the same as $A_{\beta'}$ with the ith column replaced by b. Since $x_\beta > 0$ for all β, we get that $sign(\det(A_\beta)) = sign(\det(A^i_\beta)) = sign(\det(A^i_{\beta'})) = sign(\det(A_{\beta'}))$. This proves (i).

For (ii), assume that there is a basis matrix $A_\gamma \in \mathbb{R}^{n \times n}$ for a non-representative set γ, so $A_\gamma^{-1} b > 0$. For the time being, assume that γ

has its first two elements out of block Π_1, no element out of block Π_2 and exactly one element out of every other block. Cramer's rule then tells us that

$$\frac{\det(A^1_\gamma)}{\det(A_\gamma)} > 0, \qquad \frac{\det(A^2_\gamma)}{\det(A_\gamma)} > 0.$$

So $sign(\det(A^1_\gamma)) = sign(\det(A^2_\gamma))$. Note that $\det(A^2_\gamma) = \det(A^2_\beta)$ for some representative set β and $\det(A^1_\gamma) = -\det(A^2_{\beta'})$ for some other representative β' (by swapping the first two columns in A^1_γ), implying $sign(\det(A^2_\beta)) \neq sign(\det(A^2_{\beta'}))$. This is a contradiction since by Cramer's rule and (i) we have $sign(\det(A^2_\beta)) = sign(\det(A^2_{\beta'}))$. Thus, A_γ is not a basis matrix and $sign(\det(A^1_\gamma)) \neq sign(\det(A^2_\gamma))$ must hold. Using this, we can pivot to the next γ (for example the one having two elements in the first and third block, none in the second and fourth, and one element in the other blocks) and derive a contradiction by the same arguments as above. Pivoting to all non-representative sets γ proves (ii). □

If A fulfills condition (i) of Observation 2.9, we say that A has *property* P.

For any $c \in \mathbb{R}^N$, a canonical Π-compatible LP is obtained by setting $b_i = 1$ and

$$A_{ij} := \begin{cases} 1, & j \in \Pi_i \\ 0, & \text{otherwise} \end{cases}, \quad i \in [n],\ j \in [N].$$

In this case, all representative submatrices are equal to the n-dimensional identity matrix, and the feasible region is the product of n simplices, where the i-th simplex is defined in the space of variables $x_j, j \in \Pi_i$, via the constraints

$$\sum_{j \in \Pi_i} x_j = 1, \quad x_j \geq 0.$$

The LP dual to (2.8) is

$$\begin{array}{ll} \text{maximize} & b^T y \\ \text{subject to} & y^T A \leq c^T. \end{array} \tag{2.10}$$

Since (2.8) is Π-compatible, an optimal solution x^* fulfills $x^*_\beta > 0$ for some representative set β. By complementary slackness (see for example [56]), the constraints corresponding to this set β in $y^T A \leq c^T$ are fulfilled with equality in an optimal solution to (2.10).

By dropping the requirement that (2.8) is Π-compatible, but still insisting on A to have property P, we get the following more general problem: given matrix A with property P and vector c, find a vector $y \in \mathbb{R}^n$ such that

$$c^T \geq y^T A, \tag{2.11}$$

and with the property that for every $i \in [n]$, there is a $j \in \Pi_i$ satisfying

$$c_j = (y^T A)_j. \tag{2.12}$$

We refer to this problem as PGLCP*(A, c), since, according to the following lemma, it is a dual form of the PGLCP.

Lemma 2.13 *Let $A \in \mathbb{R}^{n \times N}$ and $c \in \mathbb{R}^N$ such that A has property* P. *The* PGLCP*(A, c) *is equivalent to a* PGLCP, *in the sense that a solution to one problem yields a solution to the other.*

Proof. Given the PGLCP*(A, c) with A partitioned by $\Pi = (\Pi_1, \ldots, \Pi_n)$, fix a representative set β and define

$$\begin{aligned} G^T &:= A_\beta^{-1} A, \\ q^T &:= c^T - c_\beta^T A_\beta^{-1} A. \end{aligned}$$

G is a vertical block P-matrix of type $(|\Pi_1|, \ldots, |\Pi_n|)$ which easily follows from the fact that G^T has property P and contains a representative identity submatrix. We look at the following PGLCP:

$$\begin{aligned} w - Gz &= q && (2.14) \\ w, z &\geq 0 && (2.15) \\ \prod_{j \in \Pi_i} w_j &= 0, \quad i \in [n]. && (2.16) \end{aligned}$$

Since the z-variables do not appear in the complementarity condition (2.16), this is actually not a proper PGLCP. But by definition, $q_\beta = 0$ and $(G^T)_{\beta_i} = \mathbf{e}_i$ for β_i being the unique element in $\beta \cap \Pi_i$ and $\mathbf{e}_i \in \mathbb{R}^n$ being the i-th unit vector. Consequently, every solution to (2.14) must satisfy $z_i = w_{\beta_i}$ for $i \in [n]$. This means that (2.16) may be replaced with

$$z_i \prod_{j \in \Pi_i} w_j = 0, \quad i \in [n],$$

making the PGLCP proper.

Given a solution w, z to this PGLCP, a solution y to the PGLCP* is given by $y^T = (c_\beta^T - z^T)A_\beta^{-1}$. This is because (2.14) gives $w^T = c^T - y^T A$ and conditions (2.15) and (2.16) then ensure that y is as desired.

Vice versa, given a PGLCP(G, q) we enhance $G \in \mathbb{R}^{(N-n)\times n}$ by a representative identity submatrix and q accordingly by n zero entries at coordinates corresponding to the representative identity submatrix. Given a solution y to the PGLCP* with $A := G^T$ and $c := q$, a solution w, z to the (enhanced) PGLCP is given by $z^T = -y^T$ and $w^T = c^T - y^T A$. $\square$

Intuitively, PGLCP* is LP of the form (2.8) 'without a right-hand side' and therefore a generalization of Π-compatible linear programming. The generalization is proper, because given a PGLCP* instance (A, c), it is not always possible to find a right-hand side b such that A, b, c form a Π-compatible LP. We prove this in Chapter 5, where we show that b exists if and only if the matrix G, constructed from A as in the proof above, belongs to the class of *hidden* K*-matrices*.

Chapter 3

Unique Sink Orientations

The term *unique sink orientation* (USO) first appears in a paper by Szabó and Welzl [89]. There, a USO is an edge-orientation of the n-dimensional hypercube such that every face/subcube of the cube has a unique sink (a vertex with all edges within the subcube incoming). In the author's Master thesis, this concept has been generalized to orientations of *grids* (to be defined shortly) and it has been shown that the PGLCP can be mapped to finding the unique sink of a grid USO [78]. This result, which was later published as part of [33, 32], is a generalization of the work of Stickney and Watson [87], who reduced the PLCP to unique sink orientations of cubes.

We show the reduction from PGLCP to grid USO in the setting (2.6) that is different from the setting in [33, 32] (where we used the PGLCP* setting of Section 2.2), but more natural in the context of this thesis. We review the *Holt-Klee condition* in Section 3.2, and at the end of this chapter we devote a section to the analysis of special 2-dimensional grid USOs which we call *ladders*. We describe an optimal algorithm to find the sink in a ladder of size 3 and an almost optimal algorithm to find the sink in a Holt-Klee ladder of general size.

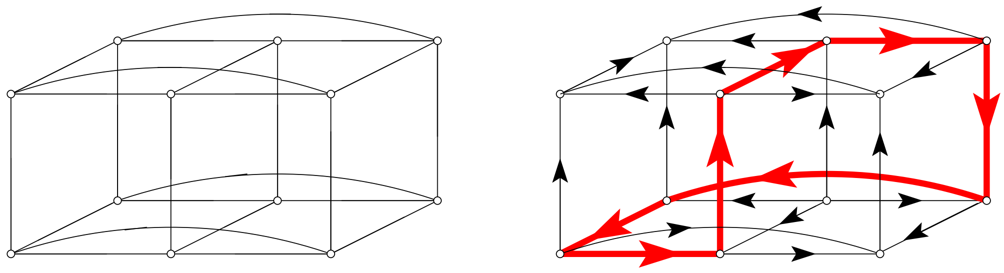


Figure 3.1: The 3-dimensional grid$([N],\Pi)$ with $N = 7$ and $\Pi = (\{1,2,3\},\{4,5\},\{6,7\})$ and a USO of it.

3.1 Reduction from PGLCP to Grid USO

An n-dimensional *grid* is given by a partition $\Pi = (\Pi_1,\dots,\Pi_n)$ of some ground set $[N]$ into n blocks. We consciously use the same letters Π, N and n as in the PGLCP(H,q), since the PGLCP with $(N-n)\times N$ matrix H, partitioned into blocks of columns according to $\Pi = (\Pi_1,\dots,\Pi_n)$, reduces to an n-dimensional grid defined by $[N]$ and Π.

Definition 3.1 *The n-dimensional* grid$([N],\Pi)$*, spanned by the set $[N]$ and a partition $\Pi = (\Pi_1,\dots,\Pi_n)$ of it with $|\Pi_i| \geq 2$ for all i, is the undirected graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$ with*

$$\begin{aligned}\mathcal{V} &:= \{V \subseteq [N]\colon |V \cap \Pi_i| = 1, i = 1,\dots,n\},\\ \mathcal{E} &:= \{\{V,V'\} \subseteq \mathcal{V}\colon |V \oplus V'| = 2\}.\end{aligned}$$

The vertices naturally correspond to the Cartesian product of the Π_i. The edges in $\mathcal{E}$ connect vertices/sets in $\mathcal{V}$ that differ in exactly one component. Every vertex therefore has $(N-n)$ neighbors. See Figure 3.1 left for an example of a grid. We ask for $|\Pi_i| \geq 2$ in the definition of the grid$([N],\Pi)$, since a block Π_{i^*} of size one, holding element j say, results in a smaller dimensional grid that is equivalent to the grid$([N \setminus j],(\Pi_1,\dots,\Pi_{i^*-1},\Pi_{i^*+1},\dots,\Pi_n))$.

Every subset $F \subseteq [N]$ defines a vertex induced *subgrid*(F,Π) of the grid$([N],\Pi)$ by restricting to vertices $V \in \mathcal{V}$ for which $V \subseteq F$ (we should actually also restrict Π to act only on elements of F, but for notational simplicity we ask the reader to keep this in mind and leave it as it is). The subgrid(F, Π) is the empty graph whenever $F \cap \Pi_i = \emptyset$ for some i. We say that such an F is *not* Π*-valid*, and it is Π*-valid* otherwise. A nonempty subgrid(F, Π) has less dimensions than the grid$([N], \Pi)$ if $|F \cap \Pi_i| = 1$ for some $i \in [n]$.

Definition 3.2 *An edge-orientation ψ of the grid($[N], \Pi$) is called a* unique sink orientation *(USO) if all nonempty subgrids have unique sinks w.r.t. ψ.*

Grid USOs can in general be cyclic, see Figure 3.1 right for an example. All 2-dimensional grid USOs are acyclic [33, 32], so the smallest cyclic grid USO is the one given in Figure 5.2 on page 92. The orientation given in Figure 3.1 right is special, since it is a smallest cyclic grid USO with all subcube USOs being acyclic.

We now show how to reduce the PGLCP to grid USO. The representative sets β in the PGLCP naturally correspond to the vertices V in the grid. There is an edge between vertex β and β' if and only if β and β' differ in exactly one component that belongs to a block Π_i, so $\beta_i \neq \beta'_i$. This is in turn the case if and only if, up to reshuffling columns, the basis matrices $B(\beta)$ and $B(\beta')$ differ in exactly one column. More precisely, there is a column with index $i_{\beta\beta'}$ in $B(\beta)$ and a column with index $i_{\beta'\beta}$ in $B(\beta')$, such that removing column $i_{\beta\beta'}$ from $B(\beta)$ results in the same matrix as removing column $i_{\beta'\beta}$ from $B(\beta')$. Note that $i_{\beta\beta'} = i_{\beta'\beta}$ if and only if $\beta_i = \beta'_i \pm 1$.

An orientation ψ of this grid graph is derived according to the following rule, where the notation

$$\beta' \xrightarrow{\psi} \beta$$

denotes that the orientation ψ induces the edge directed from β' to β:

$$\beta' \xrightarrow{\psi} \beta \quad \Leftrightarrow \quad (B(\beta)^{-1}q)_{i_{\beta\beta'}} > 0. \tag{3.3}$$

In the simpler PLCP setting, where bases and basis matrices are defined according to Definition 1.5, this can equivalently be stated as

$$\alpha' \xrightarrow{\psi} \alpha \quad \Leftrightarrow \quad (B(\alpha)^{-1}q)_i > 0, \tag{3.4}$$

where $\alpha \oplus \alpha' = \{i\}$.

Theorem 3.5 *The grid-orientation ψ induced by the* PGLCP *via (3.3) is a unique sink orientation.*

Proof. Since the PGLCP has a unique solution β that fulfills $B(\beta)^{-1}q > 0$, β is the global unique sink with all $(N-n)$ edges incoming. It thus remains to show that every proper subgrid has a unique sink, which we do by showing that the orientation of each subgrid is again PGLCP-induced. Since edges are subgrids, this also proves that each edge is directed into exactly one direction, so

$$(B(\beta)^{-1}q)_{i_{\beta\beta'}} > 0 \quad \Leftrightarrow \quad (B(\beta')^{-1}q)_{i_{\beta'\beta}} < 0.$$

Fix a nonempty subgrid(F, Π), $F \subseteq [N]$, and let β be any vertex in it. We pre-multiply the PGLCP(H, q) by $B(\beta)^{-1}$ and get the PGLCP(H', q'). This is a principal pivot transform, transforming the P-matrix G underlying H into P-matrix G' underlying H', as discussed at the end of Section 2.1. The PPT rearranges the variables. The grid orientation arising via (3.3) from the transformed PGLCP is isomorphic to the original one. The orientation in the subgrid(F, Π) is then induced by the PGLCP derived as follows. For all $j \in [N] \setminus F$ delete the corresponding column in H'. This column is zero in all entries except one, where it is 1 (that's the reason we did the PPT). Delete the row from H' that contains this 1 and delete the corresponding entry in q'. The deleted row contains the information how x_j is determined in terms of the other x-variables and a constant in q'. Since x_j does not appear in any other equation, the orientation induced by the resulting (non-standard) sub-GLCP coincides with the orientation of the subgrid$([N] \setminus j, \Pi)$ (because at every basis common to the PGLCP and the sub-GLCP, the x-variables different from x_j have the same values in the PGLCP as in the sub-GLCP).

The sub-GLCP might be non-standard because we might end up with only one column in a block, but we need two columns per block for the complementarity condition to make sense. After deletion of all columns in $[N] \setminus F$ and the corresponding rows from H', the remaining matrix is a merging of an identity matrix and a matrix G'', derived from G' by deleting rows. If there is only one column left in one block (which must be a column of G'' corresponding to a vertical block in G' that has been erased completely), it can be deleted since the corresponding variable has to be zero by the complementarity condition. This results in a standard GLCP. And the fact that G' is a P-matrix immediately implies that G'' is a P-matrix. □

The PLCP is a special case of the PGLCP where every block of the matrix H consists of two columns. Or, in the setting (2.2), every block of G has one row. The reduction thus works for the PLCP, too, in which case the grid is a cube.

Thanks to the reduction, finding the solution to a PGLCP is equivalent to finding the sink of a grid USO. Indeed, the fastest algorithms known to solve PLCP as well as PGLCP are sink-finding algorithms. We refer the reader to the literature for descriptions and analysis of algorithms working for general cube [89] and grid [33, 32] USOs. Our new contribution in this direction is the development of new algorithms for some special classes of grid USOs in Section 3.3. Moreover, the application of the *violator space framework* described in Chapter 4 results in a fast algorithm for fixed-dimensional grid USOs.

An interesting problem is to characterize USOs that are PGLCP-induced. Acyclicity is not required, since there are examples of PLCPs that reduce to cyclic USOs [87]. But it is known [33, 32], that PGLCP-induced grid USOs have a simple property known as the *Holt-Klee* condition. Before we look at this condition, we define the *outmap* of a grid USO, a concept which we will need in later sections.

Any USO can be specified by associating each vertex V with its outgoing edges. Given V and $j \in [N] \setminus V$, we define $V \rhd j$ to be the unique vertex $V' \subseteq V \cup \{j\}$ that is different from V, and we call V' the *neighbor* of V *in direction* j. Note that V is a neighbor of V' in some direction different from j.

Definition 3.6 *Given an orientation* ψ *of the grid graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *determined by the set* $[N]$ *and its partition* Π*, the function* $s_\psi : \mathcal{V} \to 2^{[N]}$*, defined by*

$$s_\psi(V) := \{j \in [N] \setminus V : V \overset{\psi}{\to} V \rhd j\}, \qquad (3.7)$$

is called the outmap *of* ψ*.*

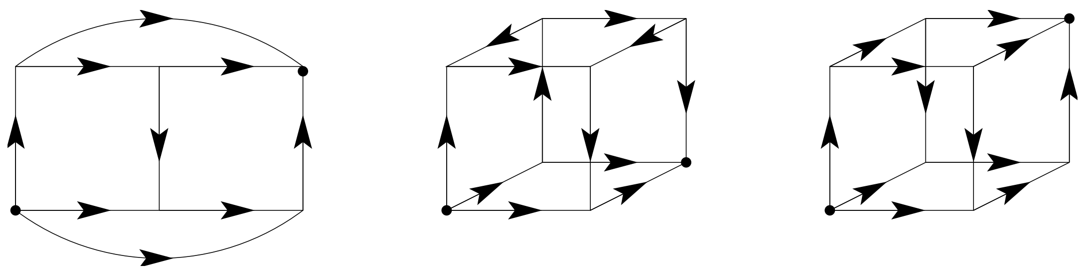


Figure 3.2: The forbidden non-HK subgrid USOs up to dimension 3.

By this definition, any sink w.r.t. ψ has empty outmap value.

3.2 The Holt-Klee Condition

A set of directed paths from the unique source to the unique sink of a grid USO is called *vertex-disjoint* if no two of the paths share any vertices other than the source or sink. A proof that a grid USO has a unique source can be found in [33, 32]. We say that a grid USO is *Holt-Klee* (HK) if the following definition applies.

Definition 3.8 *A grid* USO *satisfies the* Holt-Klee condition *if there are as many vertex-disjoint paths from the unique source to the unique sink as there are neighbors of the source, and if in addition, every nonempty subgrid* USO *satisfies the Holt-Klee condition.*

One can show that grid USOs up to 3 dimensions are HK if and only if no subgrid orientation is one of those given in Figure 3.2 [25, 33, 32]. The leftmost grid orientation in Figure 3.2 is called the *double twist*, and the two cubes are the only 3-cube USOs that are not HK [87].

This does not generalize to higher dimensions, so excluding the double twist and all cubes that are not HK is not enough for a grid USO to be HK. The smallest example is the orientation of the grid depicted in Figure 3.3, where the big arrow indicates the common orientation of all edges between the left and right subgrid. The orientation itself is not HK (try to build two vertex-disjoint paths from source to sink, starting with edges 1 and 2), but all its subgrids are. In other words, this orientation is a minimal, non-cubical obstruction for the Holt-Klee

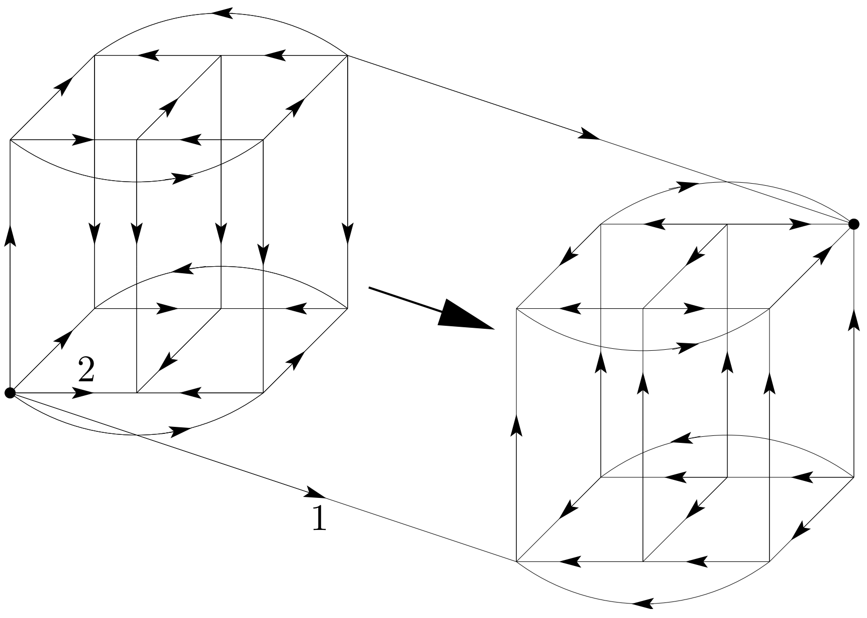


Figure 3.3: A minimal non HK 4-dimensional grid USO.

condition in dimension $n = 4$. The example can be extended to yield a minimal non-cubical obstruction in any dimension $n \geq 4$.

An open question is whether there is a finite family of forbidden subgrid orientations for given n whose absence makes any n-dimensional grid USO HK.

Let's go back to the cone point of view of the PGLCP. Remember that the columns of basis matrices $B(\beta)$ of PGLCP(H, q) span complementary $(N - n)$-dimensional cones. Further, the existence of a unique PGLCP solution for all q implies that these cones cover the whole space and that the intersection of two complementary cones is a lower dimensional cone. The dual graph underlying the cone point of view is derived by interpreting the $(N - n)$-dimensional complementary cones as vertices. Two vertices are joined by an edge if the corresponding complementary cones intersect in a $(N-n-1)$-dimensional cone. This dual graph is exactly the grid graph we get in the reduction from PGLCP to grid USO.

In a nondegenerate PGLCP, q is in general position, meaning that it is not contained in any hyperplane containing a $(N-n-1)$-dimensional intersection of two complementary cones. If a vector q is in general position, we can define an orientation of the dual graph, in which an edge joining the neighboring complementary cones K and K' is oriented from K to K' if $K \setminus K'$ and q are on opposite sides of the hyperplane containing $K \cap K'$. The digraph derived that way has a unique sink and source, which are the cones containing q and $-q$. In [33, 32] we presented the proof that these orientations fulfill the Holt-Klee condition. Since results in [33, 32] are derived in the PGLCP* setting (see Section 2.2), we give a short argument why the orientation defined in Section 3.1 through Equation (3.3) is in fact the same as the one of the dual graph just described.

The neighboring complementary cones K and K' are spanned by the columns of the basis matrices $B(\beta)$ and $B(\beta')$, respectively. Up to reshuffling columns, $B(\beta)$ differs from $B(\beta')$ only by the column with index $i_{\beta\beta'}$ in $B(\beta)$ (as discussed in the previous section). The vectors q and $B(\beta)_{\{i_{\beta\beta'}\}}$ (corresponding to $K \setminus K'$) lie on opposite sides of the hyperplane containing the cone spanned by the columns shared by $B(\beta)$ and $B(\beta')$ (corresponding to $K \cap K'$) if and only if $(B(\beta)^{-1}q)_{i_{\beta\beta'}} < 0$. PGLCP-induced grid USOs defined through (3.3) therefore fulfill the Holt-Klee condition.

The fact that PGLCP-induced USOs are HK might be exploited by sink-finding algorithms. In the next section, we design such an algorithm for 2-dimensional grids with one dimension fixed to size 2.

3.3 Ladders

The work in this section evolved from a problem posed at the 2nd Gremo's Workshop on Open Problems (GWOP) 2004[1]. We abstract from the PGLCP and focus on the problem of finding a sink in a grid USO.

A USO is usually implicitly given through a *vertex evaluation* oracle that returns for any given vertex the orientations of the incident edges. An optimal algorithm finding the sink in the 1-dimensional grid$([N], (\Pi_1))$ can easily be seen to need expected H_N vertex evaluations, where H_N is the Nth harmonic number: the runtime of the best deterministic algorithm on the input of uniformly at random dis-

[1]`http://www.ti.inf.ethz.ch/ew/workshops/gwop04/`

tributed USOs can be computed to be H_N expected vertex evaluations. By Yao's Principle [96], this is a lower bound for the expected worst-case runtime of the best randomized algorithm. Since the randomized algorithm that follows a random outgoing edge needs expected H_N vertex evaluations (independent of the USO distribution), H_N is optimal in the 1-dimensional grid.

For general grid USO, two sink finding algorithms have been presented in [33, 32]. Here, we specialize to *s-ladders*, 2-dimensional grids with one block of size 2 and the other of size s. By the previous discussion, this is the simplest nontrivial case of a (non-cubic) grid (see [89] for algorithms for cube USOs). The goal of this specialization is to reuse gained knowledge in higher dimensional grids and to show how HK can be exploited.

This section is structured as follows. After the formal definition of ladders, we analyze the behavior of a simple but general grid USO algorithm on them. Then a specific algorithm for ladders is given which seems to make a lot of sense but surprisingly turns out to be slower than the simple algorithm. Finally, we present an optimal algorithm for finding the sink in a 3-ladder and a nearly optimal algorithm for HK ladder USOs.

Definition 3.9 *The* s-ladder L_s *is the* 2*-dimensional grid*$([s+2],\Pi)$ *with* $\Pi = (\Pi_1, \Pi_2)$*,* $|\Pi_1| = 2$ *and* $|\Pi_2| = s$*.*

See Figure 3.4 for the example L_4 with 4 *steps.* In general, L_s consists of two complete graphs on s vertices where each vertex in one of these graphs is connected with exactly one in the other graph. These connections form the s steps of the ladder.

For the PGLCP, a vertex evaluation oracle can be implemented to run in time polynomial in the size of the matrix H: returning the orientations of the edges incident to vertex β amounts to computing $B(\beta)^{-1}q$. In case of a ladder, $B(\beta)$ is a square matrix of dimension s.

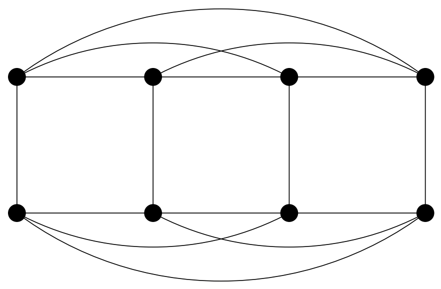


Figure 3.4: The ladder L_4 with 4 steps.

3.3.1 Simple Algorithms

Let's first analyze how the *product algorithm*, developed for general grids in [33, 32], behaves on ladders. It removes a random step of the ladder and recursively evaluates the sink of the resulting subladder L_{s-1}. In case this sink is not yet the global sink (its unique incident edge connecting it to the removed step will tell), the global sink is in the removed step. This happens with probability $1/s$, and to find the sink of the step, 3/2 evaluations suffice on average. We therefore get the recurrence

$$t(s) = t(s-1) + \frac{3}{2s}, \tag{3.10}$$

with $t(0) = 0$, for the expected total number of vertex evaluations. This yields

$$t(s) = \frac{3}{2}H_s, \quad s \geq 0.$$

Knowing that the sink of L_2 (the 2-cube) can be found with an expected number of 43/20 vertex evaluations (this is optimal) [89], we can slightly improve the product algorithm by using

$$t(2) = \frac{43}{20}$$

as a base of the recurrence. This yields

$$t(s) = \frac{3}{2}H_s - \frac{1}{10}, \quad s \geq 2. \tag{3.11}$$

Another approach to construct an algorithm is via a property of grid USO. Specialized to ladder USO, it says that for every pair $(i, j), 0 \leq i < 2, 0 \leq j < s$, there is exactly one vertex with i outgoing vertical edges and j outgoing horizontal edges [33, 32]. The pair (i, j) is the *refined index* of that vertex.

Choose one vertex v at random. If its refined index is $(0, j)$, then a vertex like v' in Figure 3.5 can not be the sink, since that would result in v and v' both being sinks in the 2-cubical face spanned by them. The global sink is therefore in the subladder L_j spanned by the j steps that contain the neighbors of v along its j outgoing edges, see Figure 3.5. Thus, we recursively evaluate the sink of this subladder and are done. If $j = 0$, the subladder is empty and v itself is the global sink.

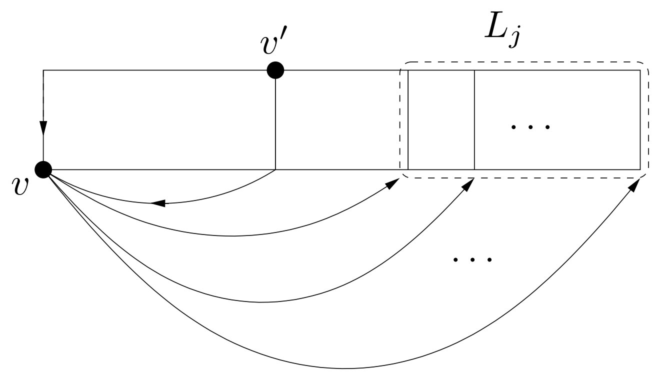


Figure 3.5: Case 1: v has refined index $(0, j)$; the global sink is in the subladder L_j (or equals v if $j = 0$).

If v's refined index is $(1, j)$, we know that the global sink is either in the complete graph K_s not containing v, or in the K_j spanned by the neighbors of v along its j outgoing edges, see Figure 3.6. Strategy 1 is to evaluate the sink of the K_s first, and if this didn't turn up the global sink, do the K_j afterwards. Strategy 2 proceeds vice versa. Taking into account that H_m evaluations are necessary and sufficient to deal with K_m, we arrive at the following runtimes of the two strategies.

	sink is in K_s	sink is in K_j
Strategy 1	H_s	$H_s + H_j$
Strategy 2	$H_s + H_j$	H_j

We choose a mixed strategy resulting from running Strategy 1 with probability p and Strategy 2 with probability $1 - p$. The best p is obtained when the mixed strategy has the same expected runtime for

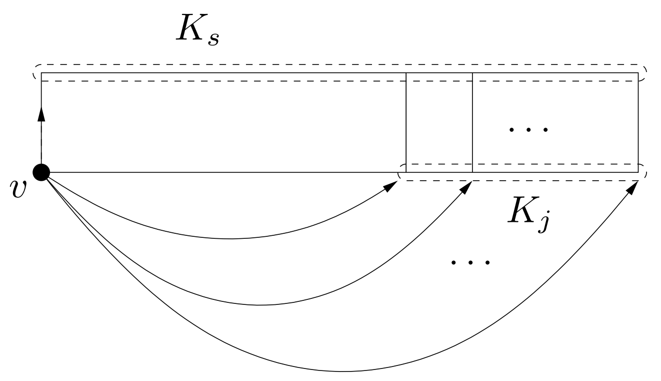


Figure 3.6: Case 2: v has refined index $(1, j)$; the global sink is in the upper K_s or in the lower K_j.

both possible locations of the sink. Therefore, we solve

$$pH_s + (1-p)(H_s + H_j) = p(H_s + H_j) + (1-p)H_j,$$

which gives

$$p = \frac{H_s}{H_s + H_j},$$

and the expected runtime of the mixed strategy is

$$H_s + H_j - \frac{H_s H_j}{H_s + H_j}.$$

Because any refined index value (i, j) has probability $1/(2s)$ of appearing, we can summarize the above in the recurrence

$$t(s) = 1 + \frac{1}{2s}\sum_{j=0}^{s-1}\left(t(j) + H_s + H_j - \frac{H_s H_j}{H_s + H_j}\right), \tag{3.12}$$

with $t(0) = 0$.

We are unable to solve this recurrence, but a rough calculation shows that it will probably not improve over the product algorithm, asymptotically. Assume we inductively want to prove $t(s) \leq cH_s$, for some constant c. For most j, H_j is very close to H_s, so let us argue that

$$H_s + H_j - \frac{H_s H_j}{H_s + H_j} \approx H_s + H_s - \frac{H_s H_s}{H_s + H_s} = \frac{3}{2}H_s.$$

In order for the inductive proof to work, we should then have

$$\begin{aligned}
1 + \frac{1}{2s}\sum_{j=0}^{s-1}\left(t(j) + H_s + H_j - \frac{H_sH_j}{H_s + H_j}\right) &\approx \frac{3}{4}H_s + \frac{c}{2s}\sum_{j=0}^{s-1}H_j \\
&\approx \frac{3}{4}H_s + \frac{c}{2}H_s \\
&\leq cH_s,
\end{aligned}$$

which gives $c \geq 3/2$.

Let us do some numerical evaluations of the bounds in (3.11) and (3.12). The following table gives some values (all values have been exactly computed first and then rounded).

s	1	2	3	13	14	50	100
Product	3/2	86/40	2.65	4.670	4.7773	6.65	7.68
Refined Index	3/2	89/40	2.71	4.672	4.7769	6.61	7.62

The product algorithm is still better than the refined index algorithm for $s \leq 13$, but starting from $s = 14$, refined index wins. The margin is not very large, though, and it is not clear whether this margin goes to infinity with s, or whether it is bounded by some constant. Exact computations of values become very slow for $s > 100$, and even when we do all computations in floating-point, Maple[2] is slow. For $s = 200$, the floating-point computations almost reach their limit; we get 8.72 for product and 8.65 for refined index.

3.3.2 A better Algorithm for $s = 3$?

Here is a specific algorithm for $s = 3$ which seems to make sense. As in the refined index algorithm, a random vertex is evaluated first, and depending on its refined index, further actions are taken. Figure 3.7 summarizes the algorithm.

Here are the runtimes in the various cases. As above, we apply the optimal mixed strategy in cases $(1, 1)$ and $(1, 2)$. In all other cases, we

[2]http://www.maplesoft.com/

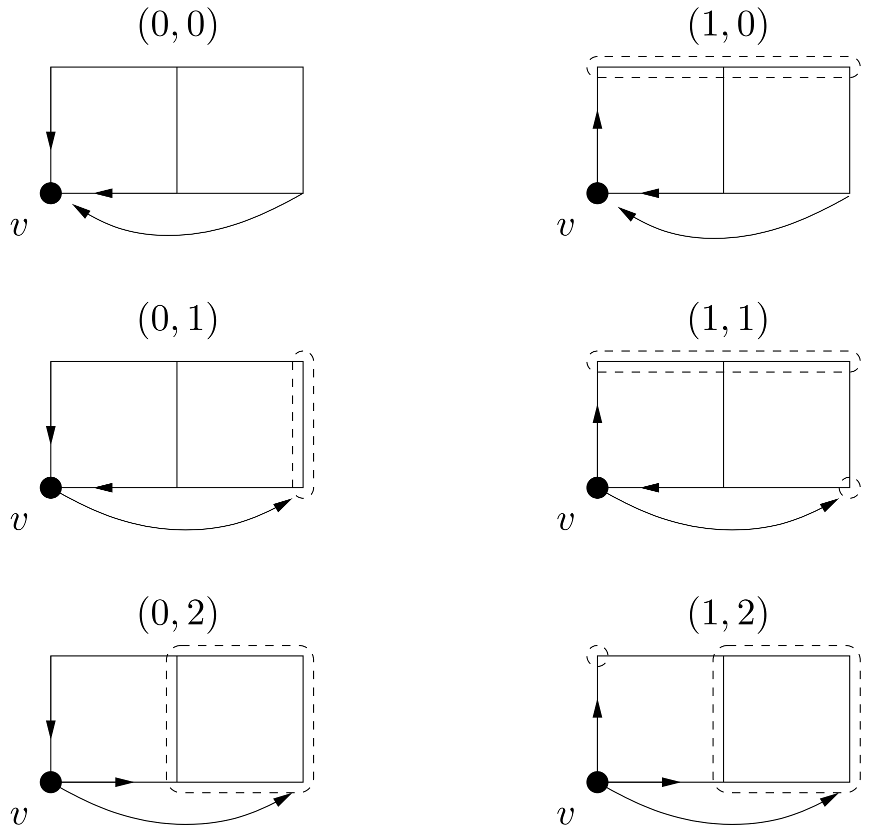


Figure 3.7: Case $(0,0)$: v is the sink; Case $(0,1)$: evaluate right column; Case $(0,2)$: evaluate right 2-cube; Case $(1,0)$: evaluate upper row; Case $(1,1)$: evaluate lower right vertex first (Strategy 1), or upper row first (Strategy 2); Case $(1,2)$: evaluate upper left vertex first (Strategy 1), or right 2-cube first (Strategy 2).

apply the optimal strategies for rows/columns and the 2-cube.

$$\begin{aligned}
(0,0) &: 1,\\
(0,1) &: 1+\frac{3}{2},\\
(0,2) &: 1+\frac{43}{20},\\
(1,0) &: 1+\frac{11}{6},\\
(1,1) &: 1+1+\frac{11}{6}-1\cdot\frac{11}{6}/\left(1+\frac{11}{6}\right)=1+\frac{223}{102}\approx 3.19,\\
(1,2) &: 1+1+\frac{43}{20}-1\cdot\frac{43}{20}/\left(1+\frac{43}{20}\right)=1+\frac{3109}{1260}\approx 3.47.
\end{aligned}$$

The average of these six values is

$$\frac{43207}{16065} \approx 2.69$$

and equals the expected number of vertex evaluations. Unfortunately, this number is larger than that obtained from the product algorithm, so what went wrong? It seems that in cases $(0,1),(0,2),(1,0)$, we can hardly do better: we know a subgrid containing the sink, but we have no information about the edge orientations within the subgrid. In this situation, the generic optimal algorithm for the subgrid should be applied. In the difficult cases $(1,1)$ and $(1,2)$, it is not clear, though, that it is best to handle the two subgrids that may contain a sink (one of them a vertex) separately. Apparently, it's not. Other subdivisions are depicted in Figure 3.8, but they are worse, as you can check. The prob-

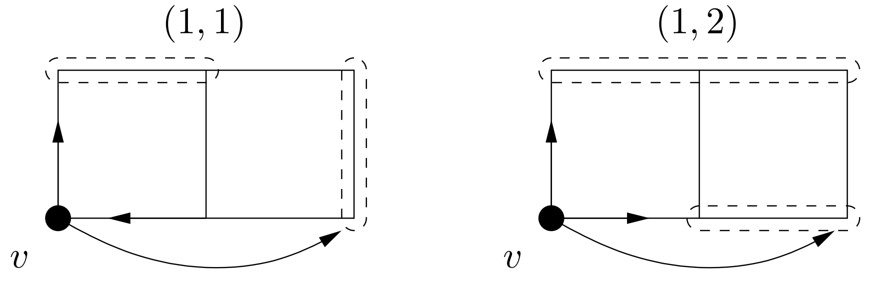


Figure 3.8: Worse ways of subdividing the possible sink locations into two subgrids.

lem is therefore to find better or even optimal strategies to deal with cases $(1,1)$ and $(1,2)$. In the following, by solving a linear program, we will develop an optimal algorithm for the 3-ladder. But first – in order to keep the LP as small as possible – we define isomorphisms of grids.

3.3.3 Isomorphisms on General Grid USOs

Remember that a vertex in the n-dimensional grid$([N],\Pi)$ is a set consisting of exactly one element out of each block Π_i, $i=1,\ldots,n$. In the following, we stick to the *bit-model*, where we represent such a set by its N-dimensional characteristic vector, partitioned into n blocks. The number of bits of a block is the size of the corresponding grid dimension. Given a USO ψ of the grid$([N],\Pi)$, the outgoing edges of a vertex V

are again described by a set of labels in $[N]$, namely the labels given by the outmap $s_\psi(V)$, see Definition 3.6. Like vertices, the outmap is represented in the bit-model by its N-dimensional characteristic vector, partitioned into n blocks.

Two USOs induced on a grid are isomorphic to each other if one can be derived from the other by swapping columns and/or rows in the grid. In the bit-model, these isomorphisms correspond to applying a number of bit-swaps, where a bit-swap swaps two bits in the same block. This is what we call an isomorphism of the first kind. Additionally, in case the grid has dimensions of the same size, isomorphisms of the second kind are described through swapping whole blocks of bits, i.e., they correspond to interchanging dimensions of the grid.

As an example take the $(3 \times 2 \times 2)$-grid. By this we mean the grid$([N], \Pi)$ with $N = 7, |\Pi_1| = 3, |\Pi_2| = 2$ and $|\Pi_3| = 2$. A possible vertex has the form 001|01|10 and we assume that it has outmap 110|00|01. An isomorphism of the first kind might swap the second and the third bit in the first block while an isomorphism of the second kind might swap the second and the third block. Combined we get the isomorphism which maps the original vertex to 010|10|01 and its outmap to 101|01|00. In this example, the number of isomorphisms of the first kind is $3! * 2! * 2!$ and the number of second kind isomorphisms is $2!$, thus all together 48 possible isomorphisms.

In the special case of the n-dimensional cubical grid, there are $2^n * n!$ isomorphisms.

The C++ program ISOS[3] returns all possible isomorphisms in the 2-dimensional array `isomorphisms`, where an isomorphism is described through bit-swaps. For example

```
isomorphisms[2] = [1,2,4,5,-1,-1,-1,-1]
```

means that applying isomorphism 2 is to swap bits 1 and 2 as well as bits 4 and 5. The -1's are there for programming reasons only.

The set of *combinatorial* USO *types* is the maximal set of USOs of a grid that are pairwise non-isomorphic. Considering only isomorphisms of the first kind, the number of combinatorial types times the number of isomorphisms is the number of possible USOs on a grid. This is not

[3]All used C++ programs are available from the author.

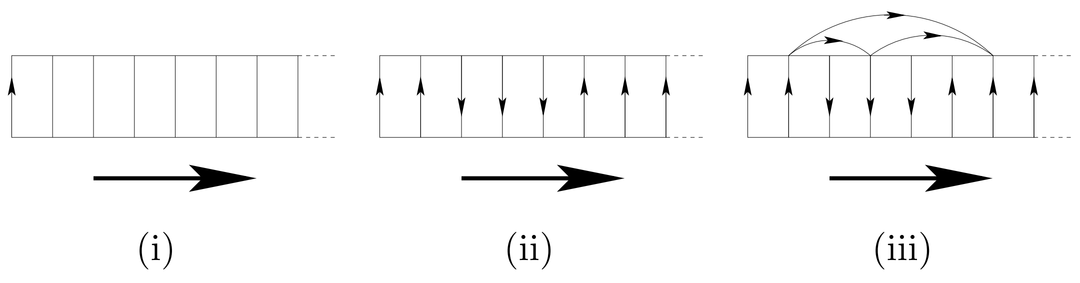


Figure 3.9: Number of combinatorial USO types of L_s.

the case with isomorphisms of the second kind (for example, regarding only second kind isomorphisms, there are 8 combinatorial USO types of the 2-cube and two isomorphisms). The following table collects some values computed by the programs ISOS and LP_GENERATOR (to which we return later). Since we'll analyze HK orientations shortly, we also include the number of combinatorial types that are HK.

	#Isos	**#Combs**	**#HK-Combs**	**#Usos**
1-cube	2	1	1	2
2-cube	8	2	2	12
3-cube	48	19	17	744
4-cube	384	14614	?	5541744
d-grid	$d!$	1	1	$d!$
(3×2)-grid	12	11	10	132
(4×2)-grid	48	47	40	2256
(5×2)-grid	240	231	192	55440
(3×3)-grid	72	82	45	5796
$(3 \times 2 \times 2)$-grid	48	2330	1322	110760

The number of combinatorial USO types for the 4-cube is due to Schurr [81]. We restrict our focus to ladders, which are $(2 \times s)$-grids. Isomorphisms of the second kind do not apply here (for $s \neq 2$).

Lemma 3.13 *The number of combinatorial* USO *types on the s-ladder, $c(s)$, obeys the following recurrence:*

$$c(s) = \sum_{i=1}^{s} c(s-i) * i!, \quad c(0) = 1.$$

Proof. Fix the leftmost edge to go upwards and all edges in the bottom row of the ladder to go from left to right. This is indicated by the thick arrow in Figure 3.9 (i). Every USO that can be achieved by completing this partial orientation to a USO defines a distinct combinatorial type. An orientation of the vertical edges gives us a partition of the ladder into blocks of upward and downward going edges. For example, in Figure 3.9 (ii), the first block consists of 2, the second of 3 and the third of 3 up-/downward directed edges. Such a partition is given by its characterization $P := (p_1, p_2, ..., p_k)$, where p_i is the number of vertical edges in the ith block. Given a characterization, the number of possible USO completions is $P! := p_1! * p_2! * \ldots * p_k!$. This can be seen through Figure 3.9 (iii), where the forced orientations of three further edges are shown. One can see that the horizontal edges between different blocks have to be oriented to the right in order to prevent cycles or multiple sinks in subgrids. Edges connecting two vertices in the same block are thus the only ones whose orientation is not yet fixed. In block i, there are $p_i!$ possibilities to orient the edges.

All together, the number of combinatorial USO types is $\sum_P P!$. Arguing in terms of the formula in the lemma, we have $i!$ possibilities to orient the edges in the first block of size i (note that the first block has size at least 1) and then $c(s-i)$ possibilities to orient the remaining $(s-i)$-ladder. □

For $s = 1, \ldots, 6$ we get $1, 3, 11, 47, 231, 1303$ combinatorial USO types. Observe that for the 2-ladder – which is the 2-cube – we get 3 combinatorial types in contrast to 2 combinatorial types in the table above. This is because we ignore isomorphisms of the second kind.

We are not able to give a closed form for the recurrence. Karttunen points out [46] that the sequence generated by the recurrence could be the invert transform of the factorial numbers, but he has no proof. The invert transform is given for example in [2].

3.3.4 Optimal Strategies and Zero-Sum Games

The problem of finding an optimal strategy, w.r.t. the number of vertex evaluations needed to find the sink in a grid USO, can be stated as a zero-sum game between two players [29]. One player, the algorithm player, wants to find a probability distribution over the possible algorithms such that the number of vertex evaluations is minimized with respect to the underlying USO played by the second player. The second player, the adversary, wants to play the USOs according to some probability distribution maximizing the number of evaluations the algorithm player needs. Using results in game theory, it is possible to get the optimal strategies for both players by solving a linear program. Although this LP is in general too big to handle, the methods of Tessaro [91] allow us to drastically reduce the size of the LP by making use of isomorphisms and thereby restricting to combinatorial USO types. We adapted Tessaro's program for cube USOs to work for grids.

The C++ program LP_GENERATOR needs as inputs the possible USOs of the grid (generated with the program GRID_USO) and all isomorphisms (generated with ISOS, see above). It outputs the LP either in Maple or CPLEX format. The latter can be processed by the CPLEX solver[4] or can be converted (using CPLEX) to MPS format which in turn can be processed by Masashi Kiyomi's EXLP solver[5]. While CPLEX is very fast but does not compute with exact arithmetic, EXLP is exact but slow (but still faster than Maple). Here is an overview of the solutions found:

Optimal expected number of vertex evaluations		
	Rational value	Floating value
1-cube	3/2	1.5
2-cube	43/20	2.15
3-cube	4074633/1369468	2.9753400591
3-cube HK	62386353/21024728	2.9672846659
d-grid	H_d	
(3×2)-grid	10001/3912	2.5564928425
(3×2)-grid HK	611/240	2.5458333333
(4×2)-grid	28491130033/9987432048	2.8526982608
(4×2)-grid HK	4827433/1704792	2.8316844518
(3×3)-grid		2.9941171939
(3×3)-grid HK		2.9654439873

The LP for the (3×3)-grid is already too large to be solved by EXLP. It is remarkable that all grids in the table need less than 3 expected vertex evaluations (the 1-dimensional d-grid for $d \leq 10$).

[4] http://www.cplex.com

[5] http://members.jcom.home.ne.jp/masashi777/exlp.html

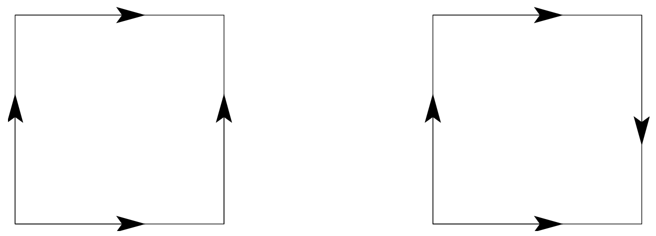


Figure 3.10: The eye (left) and the bow (right).

An optimal strategy for the algorithm player is encoded in the solution to the primal LP. For the 1-cube, the strategy is easy: evaluate one of the two vertices. If it is the sink stop, else evaluate the second vertex as well. The strategy for the 2-cube is described in [89]. Not given there is the strategy for the adversary, which can be derived by solving the dual LP. Optimally, the adversary chooses an *eye* (see Figure 3.10) with probability $\frac{1}{5}$ and a *bow* with probability $\frac{4}{5}$. The strategy for the 3-cube has first been computed by Rote (this was not published, the paper [89] contains a "personal communication" reference). It has been verified and analyzed in more detail in [91]. We restrict ourselves to the optimal strategy for the 3-ladder, optimal strategies for larger grids seem too difficult to describe.

3.3.5 The 3-Ladder

After presolving, CPLEX solves the sparse LP consisting of 48 rows, 179 columns and 1073 nonzeros in a few milliseconds. The LP is actually small enough to be solved by Maple, which yields the rational value of 10001/3912 for the objective function (expected number of vertex evaluations) and rational values for the solution values of the variables, from which an optimal strategy can be read off. The LP is degenerate, i.e., there exists more than one solution. In fact there are infinitely many solutions, since every convex combination of solutions is again a solution (Lemma 3.29 in [91]).

The optimal strategy found is given in form of a *game tree* [29] in Figures 3.11 and 3.12, where the latter figure describes the strategy when the first vertex evaluated is the source.

Going from top to bottom in the game tree, moves of the algorithm

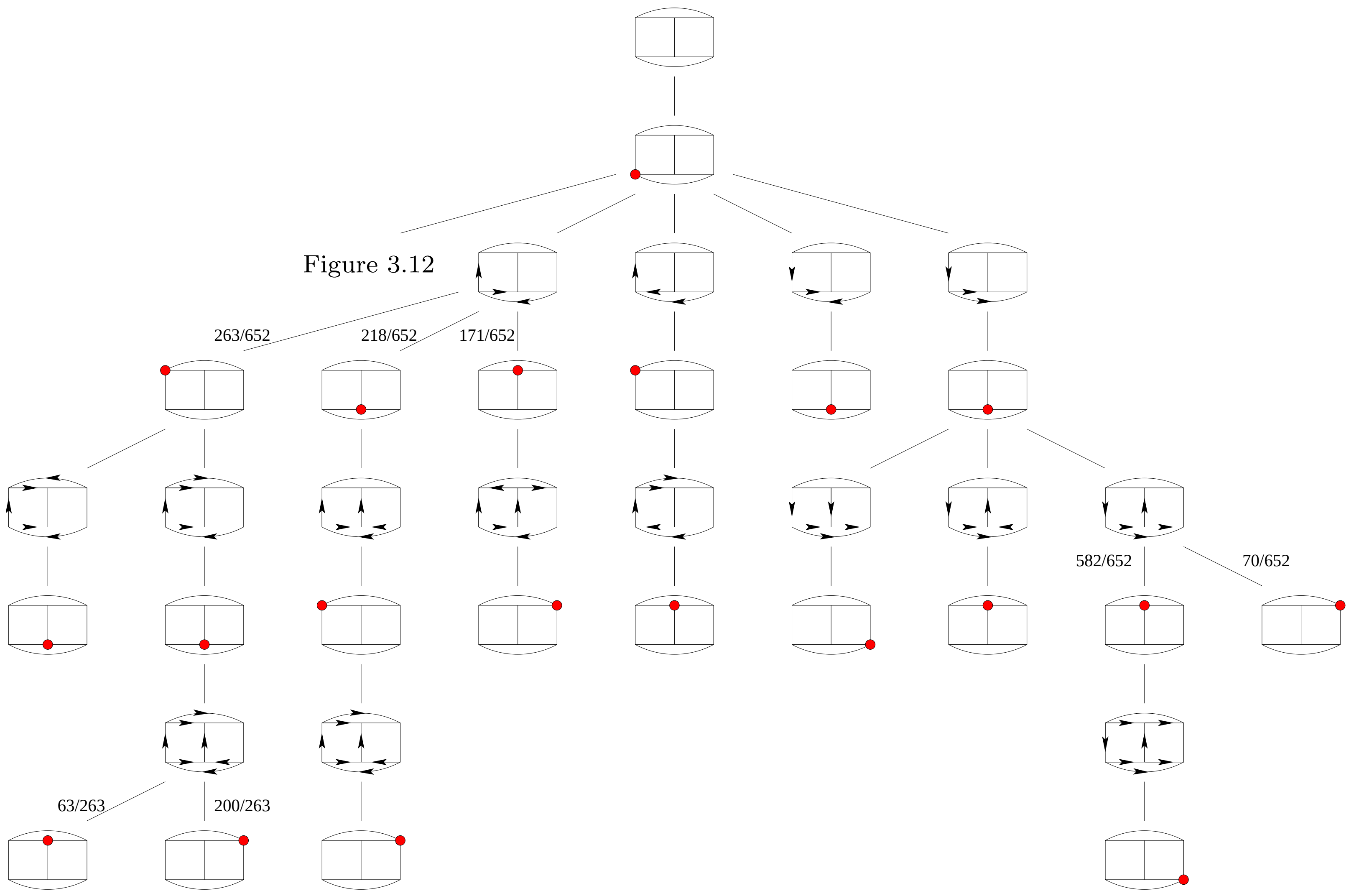


Figure 3.11: An optimal strategy. See Figure 3.12 for the case where the source is evaluated first.

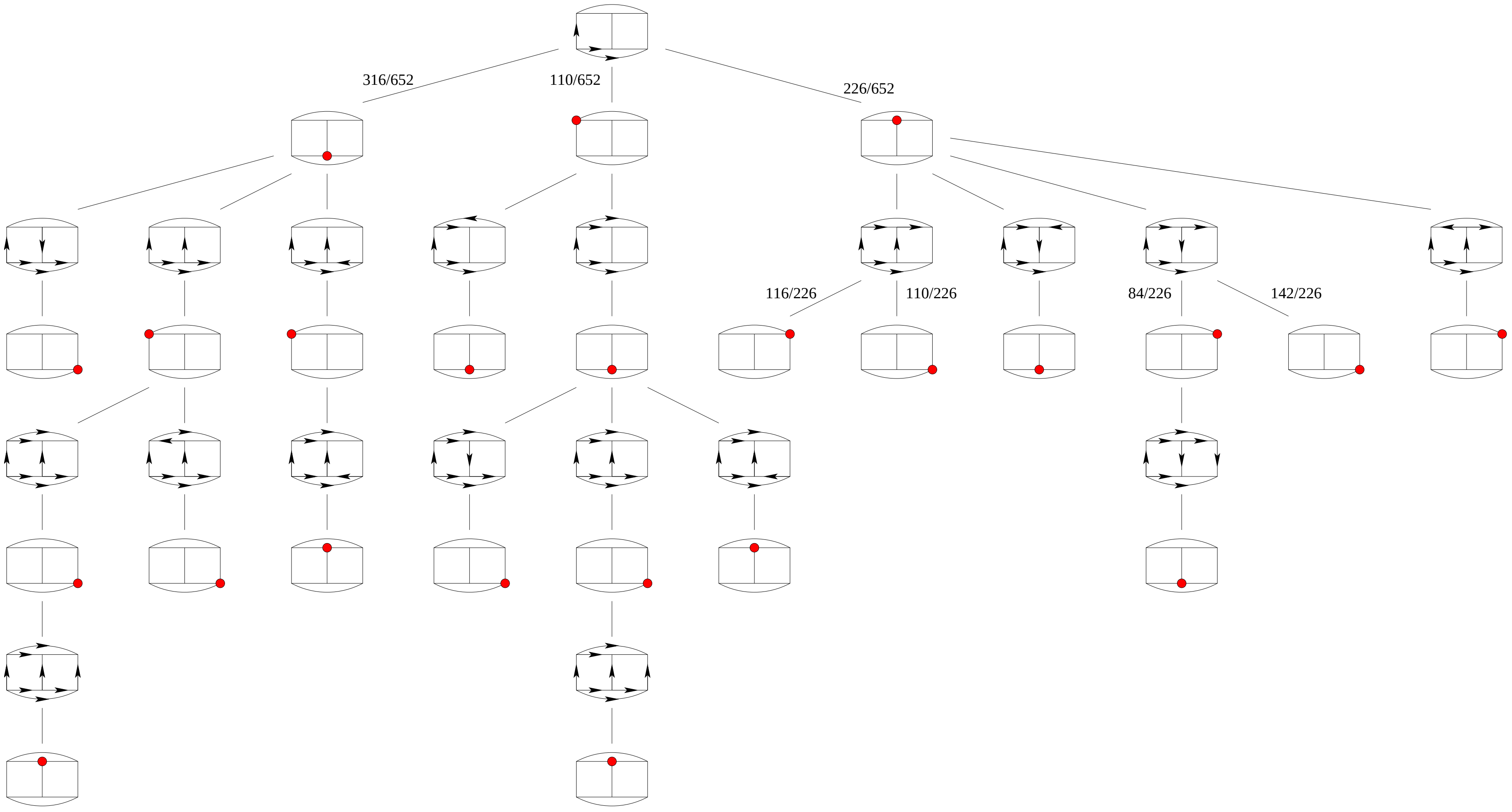


Figure 3.12: The first vertex evaluated is the source.

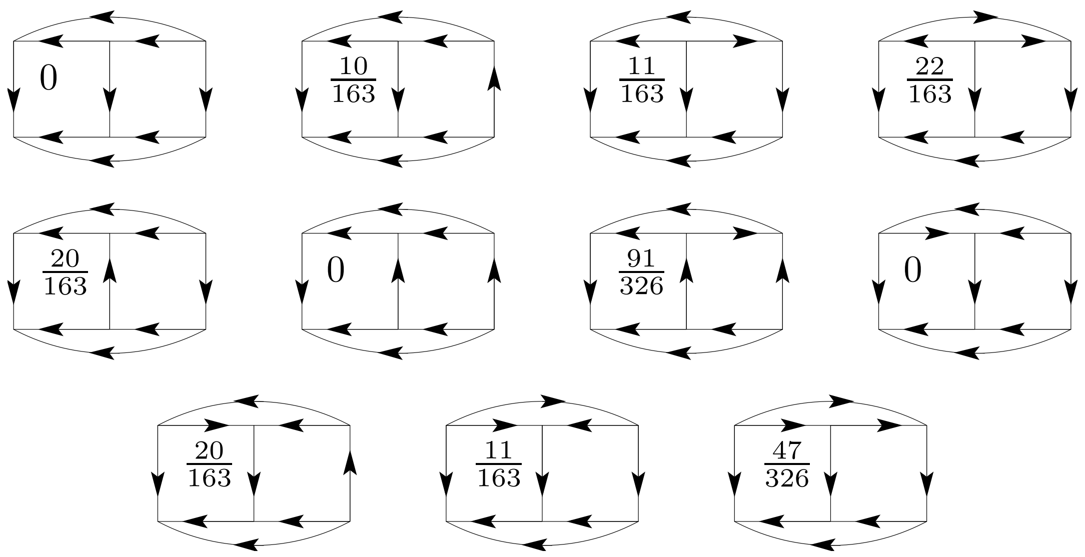
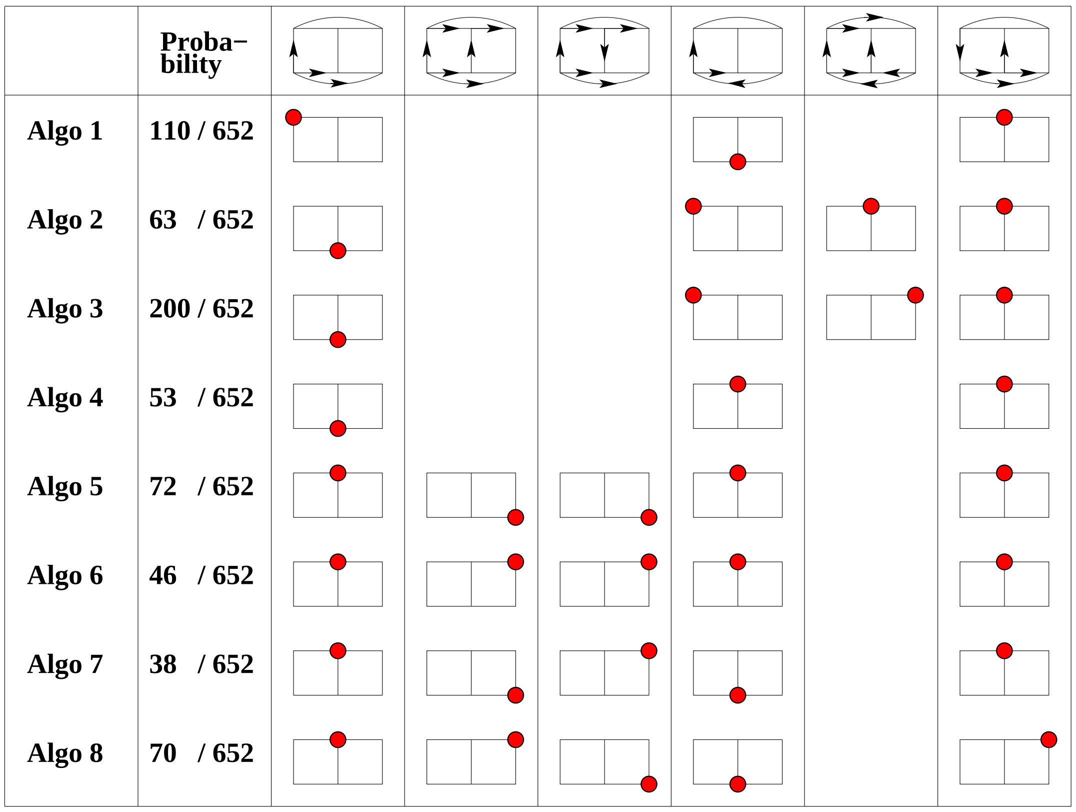


Figure 3.13: The 11 combinatorial USO types of the 3-ladder and their probabilities to be chosen by the adversary playing an optimal strategy.

player and the adversary alternate, starting with the algorithm player choosing a first vertex to evaluate (the bottom left vertex in the 3-ladder). Moves of the algorithm player not labeled by a probability are chosen with probability 1. Since the figures represent the point of view of the algorithm player, moves of the adversary have no probability: the tree shows how to react to any possible USO. In the game tree of the dual LP, the roles of the players are interchanged and the adversary's optimal strategy is visualized (not shown here). To save space, once the sink is determined, its evaluation is not given in the figure.

By our restriction to combinatorial types, the USOs given in the figures are actually classes of isomorphic USOs, thus choosing the vertex at the bottom left in the first move is equivalent to choosing any vertex. Generally, if the probability to evaluate a vertex is p in the game tree, then p is uniformly distributed among all vertices in the same *orbit* (see [91]). Vertices are in the same orbit if they are indistinguishable with respect to the partial combinatorial USO type determined so far.

An optimal strategy for the adversary is to choose a combinatorial USO type with its corresponding probability given in Figure 3.13. These probabilities can be read off a solution to the dual LP. The strategy shown is Maple's solution. Again, there are infinitely many solutions to the dual LP. Note that every optimal strategy of the adversary is a



	Proba–bility
Algo 1	110 / 652
Algo 2	63 / 652
Algo 3	200 / 652
Algo 4	53 / 652
Algo 5	72 / 652
Algo 6	46 / 652
Algo 7	38 / 652
Algo 8	70 / 652

Figure 3.14: The behavior of the eight deterministic algorithms at the six partial USOs at which the randomized algorithm chooses among several next moves with certain probabilities.

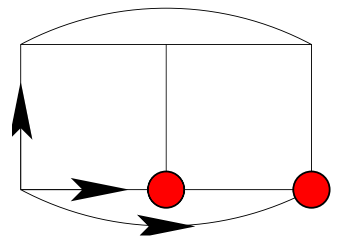
best response to every optimal strategy of the algorithm player and vice versa.

The strategy (or the randomized algorithm) of Figures 3.11 and 3.12 can be described in terms of eight deterministic algorithms that are chosen by certain probabilities. The deterministic algorithms behave exactly as given in the figures, except that they branch deterministically at vertices in the tree having more than one choice for the next move. Figure 3.14 shows the choices of the algorithms at these partial USOs and the probabilities with which an algorithm is chosen.

Let matrix M hold at position M_{ij} the number of vertex evaluations algorithm i needs to find the sink in the combinatorial USO j (enumerating them from left to right and top to bottom in Figure 3.13). Moreover, let x be the vector holding at position i the probability of taking deterministic algorithm i (the probability given in Figure 3.14) and y be the vector holding at position j the probability that the adversary chooses the combinatorial USO j (the probability given in Figure 3.13), both vectors of appropriate size. Then one can verify that $x^T My$ is indeed $10001/3912 = 2.556$, where M is the following matrix.

$$\begin{pmatrix}
11/4 & 11/4 & 11/4 & 8/3 & 31/12 & 5/2 & 29/12 & 11/4 & 8/3 & 29/12 & 5/2 \\
11/4 & 31/12 & 8/3 & 5/2 & 8/3 & 31/12 & 31/12 & 31/12 & 7/3 & 11/4 & 5/2 \\
11/4 & 31/12 & 17/6 & 5/2 & 8/3 & 31/12 & 31/12 & 31/12 & 7/3 & 31/12 & 5/2 \\
29/12 & 9/4 & 7/3 & 17/6 & 8/3 & 31/12 & 31/12 & 11/4 & 5/2 & 29/12 & 5/2 \\
25/12 & 7/3 & 2 & 31/12 & 5/2 & 31/12 & 8/3 & 29/12 & 11/4 & 5/2 & 31/12 \\
2 & 5/2 & 2 & 31/12 & 29/12 & 8/3 & 8/3 & 7/3 & 11/4 & 5/2 & 31/12 \\
29/12 & 17/6 & 5/2 & 29/12 & 9/4 & 29/12 & 5/2 & 29/12 & 11/4 & 8/3 & 11/4 \\
7/3 & 5/2 & 5/2 & 29/12 & 7/3 & 5/2 & 5/2 & 7/3 & 17/6 & 8/3 & 11/4
\end{pmatrix}$$

We remark that the eight deterministic algorithms actually don't behave purely deterministically. The first vertex to evaluate is chosen in every algorithm with probability 1/6 and in partial USOs where there is more than one vertex in the orbit, the next one is chosen u.a.r. among all of them. For example in Figure 3.12, an algorithm that branches left after the very first vertex evaluation would evaluate one of the two marked vertices in the following figure with probability 1/2 each (the single marked vertex in Figure 3.12 is a representative for the two vertices in the same orbit).



We see that an optimal algorithm is hard to describe already for small ladders. By plugging in $t(3) = 10001/3912$ as a base case for the recurrence (3.10)

$$t(s) = t(s-1) + \frac{3}{2s},$$

that we derived on page 38 for the product algorithm, gives

$$t(s) = \frac{3}{2}H_s - \frac{757}{3912}.$$

It remains open whether an optimal algorithm finds the sink in asymptotically less than the $\frac{3}{2}H_s$ vertex evaluations needed by the product algorithm.

3.3.6 Strategies for the s-Ladder

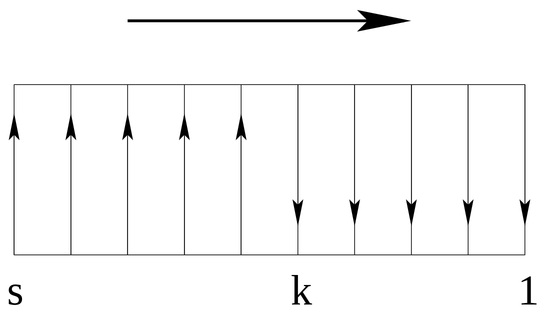
Since describing a strategy for general ladders is too complicated, we restrict our analysis to the following three distributions of the adversary:

(D1) Only combinatorial USO types that are HK are played by the adversary.

(D2) Uniform distribution (every combinatorial USO type has the same probability to be played by the adversary).

(D3) Adversary chooses HK combinatorial USO types uniformly at random.

For the following analysis, we assume that the upper horizontal edges are all directed from left to right and the leftmost vertical edge is directed upwards. It is easy to see that a combinatorial USO of a ladder is HK if and only if it has at most two blocks, one of upward the other one of downward going edges: as soon as this is not the case, the double twist of Figure 3.2 appears and the ladder is no longer HK. The number of downward going edges is denoted by $k = 0 \dots s-1$. See the next figure.



We look at the following three algorithms, all of which evaluate a random vertex in the first step:

(A1) If the vertical edge of the evaluated vertex is outgoing, follow it and evaluate the corresponding edge-neighbor. If it is incoming, evaluate a random neighbor reachable through outgoing horizontal edges. Repeat this strategy.

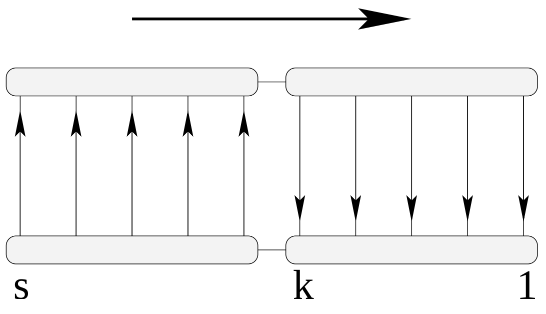
(A2) If the vertical edge of the evaluated vertex is outgoing, jump to a vertex antipodal in the 2-cubical face spanned by the vertical edge and a random outgoing horizontal edge (if there is no outgoing horizontal edge, just follow the vertical edge). If the vertical edge is incoming, evaluate a random neighbor reachable through outgoing horizontal edges. Repeat this strategy.

(A3) Proceed as in (A2). If there is more than one possible vertex to be evaluated next, choose randomly one that has fewest incoming edges evaluated so far. Repeat this strategy.

Here are some results.

	(A1)	(A2)	(A3)
(D1)	$H_s + 3/2 - 1/s$	?	?
(D2)	?	?	Optimal for $s \leq 4$
(D3)	$\leq H_s + 3/2 - 1/s$	$\leq H_s + 3/2 - 1/s$	Optimal for $s \leq 4$

The expected number of vertex evaluations (runtime) of (A1) on HK ladders depends only on the (adversary's) choice of k. For $k = 0$ it is $H_s + 1/2$. For $k > 0$ we distinguish between start vertices in one of the four areas given in the following figure. If the first vertex evaluated is



the one connected to the jth step in the upper left area ($j > k$), then the expected number of vertex evaluations is given by the recurrence

$$t(j) = 1 + \frac{1}{j-1}\sum_{i=k+1}^{j-1} t(i) + \frac{1}{j-1}\sum_{i=1}^{k}(1+H_k).$$

So $(j-1)\cdot t(j) - (j-2)\cdot t(j-1) = 1 + t(j-1)$ holds for $j \geq k+2$ which implies $t(j) = \frac{1}{j-1} + t(j-1)$. This results in $t(j) = H_{j-1} + 2$ for $j > k$.

Given this, the runtime for start vertices in the other areas are easy to derive. Since the starting vertex is picked uniformly at random, the overall runtime is

$$\begin{aligned}
T(s) &= \frac{1}{2s}\Big(\overbrace{\sum_{i=k+1}^{s} t(i)}^{\text{upper left}} + \underbrace{\sum_{i=k+1}^{s}(1+t(i))}_{\text{lower left}} + \overbrace{\sum_{i=1}^{k}(1+H_k)}^{\text{upper right}} + \underbrace{\sum_{i=1}^{k} H_k}_{\text{lower right}}\Big) \\
&= \frac{1}{s}\sum_{i=k+1}^{s} t(i) + \frac{k}{s}H_k + \frac{1}{2} \\
&= \frac{1}{s}\sum_{i=k+1}^{s}(H_{i-1}+2) + \frac{k}{s}H_k + \frac{1}{2} \\
&= \frac{1}{s}(sH_{s-1} - kH_k - (s-1) + k) + \frac{k}{s}H_k + \frac{2(s-k)}{s} + \frac{1}{2} \\
&= H_{s-1} + \frac{1-k}{s} + \frac{3}{2} \\
&= H_s - \frac{k}{s} + \frac{3}{2},
\end{aligned}$$

where we use that $\sum_{i=a}^{b} H_i = (b+1)H_b - aH_a - b + a$. This is largest for $k = 1$ which gives us the worst case runtime of $H_s + 3/2 - 1/s$ on any HK ladder.

A lower bound for the runtime on a HK ladder is derived as follows. Let the adversary choose the distribution described by the following construction: choose an orientation of the 1-dimensional s-grid (which is the complete graph on s vertices) uniformly at random among all possible orientations. Let the upper and lower part of the s-ladder be this orientation and orient all vertical edges upwards, except the one forming the step connecting the two sinks in the s-grids. This edge is directed upwards or downwards with probability 1/2. The resulting orientation is a HK ladder USO (k is 0 or 1). An optimal deterministic algorithm finds the sink by evaluating either the lower or upper s-grid first and then, if necessary (with probability 1/2), jumping to the other side for one last vertex evaluation. This results in an optimum of $H_s + 1/2$ vertex evaluations (see also the discussion on page 37). By using Yao's Principle [96] again, this is a lower bound for an optimal randomized algorithm finding the sink in a HK s-ladder. Our worst case runtime of $H_s + 3/2 - 1/s$ for HK ladders is therefore close to optimum. However, it is not optimal, which we can easily check by comparing to the values given in the table on page 47.

For general (not HK) ladders, the distribution where the adversary orients all horizontal edges from left to right and all vertical edges with probability 1/2 downwards or upwards, yields a runtime of $H_s \cdot 3/2$ for (A1). So (A1) is not better than the product algorithm in general.

Algorithm (A2) is harder to analyze, since its runtime depends on the adversary's distribution (not only on k). If the HK ladders are chosen u.a.r., then its runtime can be computed (but the formula is quite messy) and computer experiments suggest that (A2) performs slightly better than (A1) on HK ladders chosen u.a.r.

By solving a linear program as we did it for 3-ladders, we computed the optimal strategy for 4-ladders when the adversary chooses the uniform distribution, once over all ladders and once only over HK ladders. It is interesting that in both cases, (A3) is an optimal algorithm to find the sink in s-ladders with $s \leq 4$. We don't know whether this remains true for $s > 4$.

Chapter 4

Violator Spaces

Sharir and Welzl introduced an abstract framework for optimization problems, called LP-*type problems* or also *generalized linear programming problems* [84], which proved useful in algorithm design. Here we present the new framework of *violator spaces*, introduced by Matoušek and Škovroň in [85]. Violator spaces form, as we believe, a simpler and more natural framework and they constitute a proper generalization of LP-type problems. We show that grid USOs fit into this framework and that Clarkson's randomized algorithms for low-dimensional linear programming work in the context of violator spaces. Via the reduction from PGLCP to grid USO (Section 3.1), we obtain the fastest known algorithm for the PGLCP with a constant number of blocks.

4.1 LP-Type Problems

An (abstract) LP-type problem is given by a finite set H of *constraints* and a *value* $w(G)$ for every subset $G \subseteq H$. The values can be real numbers or, for technical convenience, elements of any other linearly ordered set. Intuitively, $w(G)$ is the minimum value of a solution that satisfies all constraints in G. The assignment $G \mapsto w(G)$ has to obey the axioms in the following definition.

Definition 4.1 *An* abstract LP-type problem *is a quadruple*

$$(H, w, W, \leq),$$

where H is a finite set, W is a set linearly ordered by $\leq$, and $w\colon 2^H \to W$ is a mapping satisfying the following two conditions for all $F \subseteq G \subseteq H$:

Monotonicity:	*$w(F) \leq w(G)$, and*
Locality:	*if $w(F) = w(G)$ and $w(G) < w(G \cup \{h\})$ for $h \in H$, then $w(F) < w(F \cup \{h\})$ holds.*

As our running example, we will use the smallest enclosing ball problem, where H is a finite point set in $\mathbb{R}^d$ and $w(G)$ is the radius of the smallest ball that encloses all points of G. In this case monotonicity is obvious, while verifying locality requires the nontrivial but well-known geometric result that the smallest enclosing ball is unique for every set.

It seems that the order $\leq$ of subsets is crucial; after all, LP-type problems model *optimization problems*, and indeed, the subexponential algorithm for linear programming and other LP-type problems [57] heavily relies on such an order.

A somewhat deeper look reveals that we often only care whether two subsets have the *same* value, but not how they compare under the order $\leq$. The following definition is taken from [84]:

Definition 4.2 *Consider an abstract* LP*-type problem $(H, w, W, \leq)$. We say that $B \subseteq H$ is a* basis *if for all proper subsets $F \subset B$ we have $w(F) \neq w(B)$. For $G \subseteq H$, a* basis of G *is an inclusion-minimal subset B of G with $w(B) = w(G)$.*

We observe that a minimal subset $B \subseteq G$ with $w(B) = w(G)$ is indeed a basis.

Solving an abstract LP-type problem $(H, w, W, \leq)$ means to find a basis of H. In the smallest enclosing ball problem, a basis of H is a minimal set B of points such that the smallest enclosing ball of B has the same radius (and is in fact the same) as the smallest enclosing ball of H, $w(B) = w(H)$.

In defining bases, and in saying what it means to solve an LP-type problem, we therefore do not need the order $\leq$. This is formalized by the new framework of *violator spaces*, defined in Section 4.2. Intuitively, a violator space is an LP-type problem without order. This generalization of LP-type problems is proper, and we can exactly characterize the violator spaces that “are” LP-type problems [31, 30].

Probably the most surprising insight is that Clarkson’s algorithms [11] work for violator spaces of fixed dimension, leading to an expected linear-time algorithm for “solving” the violator space. This is shown in Section 4.3. Clarkson’s algorithms were originally developed for linear programs with small dimension. They can be generalized for LP-type problems [35, 9]. The fact that the scheme also works for violator spaces may come as a surprise since the structure of violator spaces is not acyclic in general (in contrast to LP-type problems). Actually, it turns out that problems effectively solvable by Clarkson’s algorithms are in a well-defined sense exactly the violator spaces [86]; see Property 4.15 for a precise formulation.

In Section 4.4 we show that any grid USO gives rise to a violator space, but not to an LP-type problem in general. The sink of a grid USO can thus be found by solving the violator space, for example with Clarkson’s algorithms. A concrete new result is obtained by applying this to the PGLCP. Since any PGLCP gives rise to a unique sink orientation (Theorem 3.5), we may use violator spaces and Clarkson’s algorithms to solve the problem in expected linear time in the (polynomially solvable) case of a *fixed* number of blocks. This is optimal and beats all previous algorithms.

A unique sink orientation can be cyclic and we show that a cyclic orientation gives rise to a cyclic violator space and does therefore not fit into the LP-type framework. Unique sink orientations are thus non-trivial examples of possibly cyclic violator spaces. We are confident that more applications of violator spaces that are not subsumed by the LP-type framework will be discovered in the future.

4.2 The Violator Space Framework

Let $(H, w, W, \leq)$ be an abstract LP-type problem. It is natural to define that a constraint $h \in H$ *violates* a set $G \subseteq H$ of constraints if

$$w(G \cup \{h\}) > w(G).$$

For example, in the smallest enclosing ball problem, a point h violates a set G if it lies outside of the smallest ball enclosing G (which is unique).

Definition 4.3 *The* violator mapping *of* $(H, w, W, \leq)$ *is defined by* $\mathsf{V}(G) = \{h \in H\colon w(G \cup \{h\}) > w(G)\}$*. Thus,* $\mathsf{V}(G)$ *is the set of all constraints violating* G*.*

It is shown in [31, 30] that the knowledge of $\mathsf{V}(G)$ for all $G \subseteq H$ is sufficient to describe the "structure" of an LP-type problem. That is, while we cannot reconstruct W, $\leq$, and w from this knowledge, it is natural to regard two LP-type problems with the same mapping $\mathsf{V}\colon 2^H \to 2^H$ as isomorphic. Indeed, the algorithmic primitives needed for implementing the Sharir-Welzl algorithm and the other algorithms for LP-type problems mentioned above can be phrased in terms of testing violation (does $h \in \mathsf{V}(G)$ hold for a certain set $G \subseteq H$?), and they never deal explicitly with the values of w.

We now introduce the notion of *violator space*:

Definition 4.4 *A* violator space *is a pair* (H, V)*, where* H *is a finite set and* V *is a mapping* $2^H \to 2^H$ *such that*

Consistency:	$G \cap \mathsf{V}(G) = \emptyset$ *holds for all* $G \subseteq H$*, and*
Locality:	*for all* $F \subseteq G \subseteq H$*, if* $G \cap \mathsf{V}(F) = \emptyset$*, then* $\mathsf{V}(G) = \mathsf{V}(F)$*.*

A basis of a violator space is defined in analogy to a basis of an LP-type problem.

Definition 4.5 *Consider a violator space* (H, V)*. We say that* $B \subseteq H$ *is a* basis *if for all proper subsets* $F \subset B$ *we have* $B \cap \mathsf{V}(F) \neq \emptyset$*. For* $G \subseteq H$*, a* basis of G *is a minimal subset* B *of* G *with* $\mathsf{V}(B) = \mathsf{V}(G)$*.*

Observe that a minimal subset $B \subseteq G$ with $\mathsf{V}(B) = \mathsf{V}(G)$ is indeed a basis: Assume for contradiction that there is a set $F \subset B$ such that $B \cap \mathsf{V}(F) = \emptyset$. Locality then yields $\mathsf{V}(B) = \mathsf{V}(F) = \mathsf{V}(G)$, which contradicts minimality of B.

In [31, 30] it is shown that an abstract LP-type problem $(H, w, W, \leq)$ is, in a natural sense, isomorphic to an *acyclic* violator space (H, V) with V being the violator mapping. We now prepare to define the notion of an acyclic violator space. The example of a cyclic violator space at the end of this section then shows that violator spaces are a proper generalization of LP-type problems.

We fix a violator space (H, V). The set of all bases in (H, V) will be denoted by $\mathcal{B}$.

Definition 4.6 *$B, C \in \mathcal{B}$ are* equivalent*, $B \sim C$, if $\mathsf{V}(B) = \mathsf{V}(C)$.*

Clearly, the relation $\sim$ defined on $\mathcal{B}$ is an equivalence relation. The equivalence class containing a basis B will be denoted by $[B]$. The following definition of an ordering of the equivalence classes is needed for the notion of acyclicity in violator spaces.

Definition 4.7 *For $F, G \subseteq H$ in a violator space (H, V), we say that $F \leq_0 G$ (F is* locally smaller *than G) if $F \cap \mathsf{V}(G) = \emptyset$. For equivalence classes $[B], [C] \in \mathcal{B}/\sim$, we say that $[B] \leq_0 [C]$ if there exist $B' \in [B]$ and $C' \in [C]$ such that $B' \leq_0 C'$.*

The intuition of the *locally-smaller* notion comes from LP-type problems: if no element of F violates G, then $G \cup F$ has the same value as G (this is formally proved in [31, 30]), and monotonicity yields that value-wise, F is smaller than or equal to G.

Note that in the definition of $[B] \leq_0 [C]$ we do not require $B' \leq_0 C'$ to hold for *every* B' and C'. It may happen that $B' \not\leq_0 C'$ for some bases B' and C'. However, $C' \leq_0 B'$ cannot hold, as one can check.

We are now ready to define acyclic violator spaces.

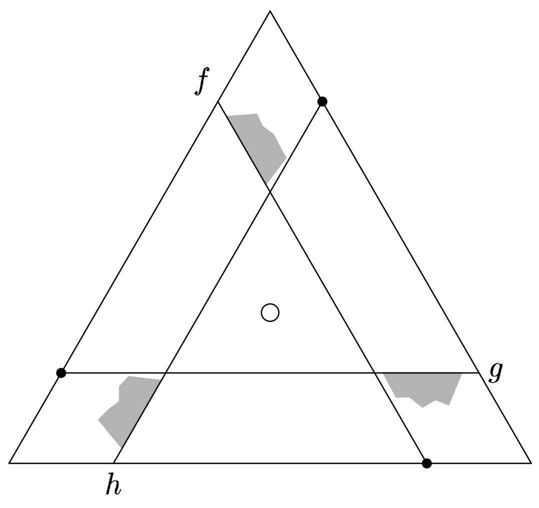


Figure 4.1: A cyclic violator space.

Definition 4.8 *Let the relation $\leq_1$ on the equivalence classes be the transitive closure of $\leq_0$. The relation $\leq_1$ is clearly reflexive and transitive. If it is antisymmetric, we say that the violator space is* acyclic, *and we define the relation $\leq$ as an arbitrary linear extension of $\leq_1$.*

To show that acyclicity need not always hold, we conclude this Section with an example of a cyclic violator space. We begin with an intuitive geometric description, see Figure 4.1, where we consider a triangle without the center point. We say that one point of the triangle is "locally smaller" than another if it is farther clockwise with respect to the center. The constraints in our violator space are the three halfplanes f, g, h.

The locally smallest point within each halfplane is marked, and a halfplane violates a set of halfplanes if it does not contain the locally smallest point in their intersection.

Now we specify the corresponding violator space formally. We have $H = \{f, g, h\}$, and V is given by the following table:

G	$\emptyset$	f	g	h	f, g	f, h	g, h	f, g, h
$\mathsf{V}(G)$	f, g, h	h	f	g	h	g	f	$\emptyset$

This (H, V) is really a violator space, since we can easily check both consistency and locality. The bases are $\emptyset$, one-element sets, and H. We have $\{f\} \leq_0 \{h\} \leq_0 \{g\} \leq_0 \{f\}$, but no two of the one-element bases are equivalent; i.e., $\leq_1$ is not antisymmetric.

4.3 Clarkson's Algorithms

We show that Clarkson's randomized reduction scheme, originally developed for linear programs with many constraints and few variables, actually works for general (possibly cyclic) violator spaces. The two algorithms of Clarkson involved in the reduction have been analyzed for LP and LP-type problems before [11, 35, 9]. The analysis we give below is almost identical on the abstract level. Our new contribution is that the combinatorial properties underlying Clarkson's algorithms also hold for violator spaces.

We start off by deriving these combinatorial properties. The analysis of Clarkson's reduction scheme is included for completeness. We note that the Sharir-Welzl algorithm is also applicable for violator spaces in a straightforward way. However, the most obvious translation of this algorithm to the setting of violator spaces is not even guaranteed to finish, since for a general violator space it may run in a cycle and the subexponential analysis thus breaks down.

4.3.1 Violator Spaces revisited

We recall that an abstract LP-type problem is defined by a quadruple $(H, w, W, \leq)$. In this subsection we will view a violator space as an "LP-type problem without the order $\leq$", i.e., we will only care whether two subsets F and G, $F \subseteq G \subseteq H$, have the same value, but not how they compare under the order $\leq$. It turns out that the order is irrelevant for Clarkson's algorithms.

Even without an order, we can talk about monotonicity in violator spaces:

Lemma 4.9 *Any violator space* (H, V) *satisfies* monotonicity *defined as follows:*

Monotonicity: $\mathsf{V}(F) = \mathsf{V}(G)$ *implies* $\mathsf{V}(F) = \mathsf{V}(E) = \mathsf{V}(G)$ *for all sets* $F \subseteq E \subseteq G \subseteq H$.

Proof. Assume $\mathsf{V}(E) \neq \mathsf{V}(F), \mathsf{V}(G)$. Then locality yields $\emptyset \neq E \cap \mathsf{V}(F) = E \cap \mathsf{V}(G)$ which contradicts consistency. $\square$

Recall Definition 4.5: a basis is a set B satisfying $B \cap \mathsf{V}(F) \neq \emptyset$ for all proper subsets F of B. A basis of G is an inclusion-minimal subset of G with the same violators. This can be used to prove Observation 4.10, well-known to hold for LP-type problems [35].

Observation 4.10 *Let* (H, V) *be a violator space. For* $G \subseteq H$ *and all* $h \in H$, *we have*

(i) $\mathsf{V}(G) \neq \mathsf{V}(G \cup \{h\})$ *if and only if* $h \in \mathsf{V}(G)$, *and*

(ii) $\mathsf{V}(G) \neq \mathsf{V}(G \setminus \{h\})$ *if and only if* h *is contained in every basis of* G.

An element h *such that (ii) holds is called* extreme *in* G.

Proof. (i) If $h \notin \mathsf{V}(G)$, we get $\mathsf{V}(G) = \mathsf{V}(G \cup \{h\})$ by locality. If $h \in \mathsf{V}(G)$, then $\mathsf{V}(G) \neq \mathsf{V}(G \cup \{h\})$ is a consequence of consistency applied to $G \cup \{h\}$. (ii) If $\mathsf{V}(G) = \mathsf{V}(G \setminus \{h\})$, there is a basis B of $G \setminus \{h\}$, and this basis, which does not contain h, is also a basis of G. Conversely, if there is some basis B of G not containing h, then $\mathsf{V}(G) = \mathsf{V}(G \setminus \{h\})$ follows from monotonicity (Lemma 4.9). $\square$

We are particularly interested in violator spaces with small bases.

Definition 4.11 *Let* (H, V) *be a violator space. The size of a largest basis is called the* combinatorial dimension $\delta = \delta(H, \mathsf{V})$ *of* (H, V).

Observation 4.10 implies that in a violator space of combinatorial dimension δ, every set has at most δ extreme elements. This in turn yields a bound for the *expected* number of violators of a random subset of constraints:

Lemma 4.12 *Let (H, V) be a violator space of combinatorial dimension δ and $W \subseteq H$ some fixed set. Let v_r be the expected number of violators of the set $W \cup R$, where $R \subseteq H \setminus W$ is a random subset of size $r < N = |H|$. Then*

$$v_r \leq \delta \frac{N-r}{r+1}.$$

The lemma can be proved using the *Sampling Lemma* of [36]. We give an independent proof here.

Proof. By definition of v_r and Observation 4.10

$$\begin{aligned} \binom{|H \setminus W|}{r} v_r &= \sum_{\substack{R \subseteq H \setminus W \\ |R|=r}} \sum_{h \in (H \setminus W) \setminus R} [h \text{ violates } W \cup R] \\ &= \sum_{\substack{R \subseteq H \setminus W \\ |R|=r}} \sum_{h \in (H \setminus W) \setminus R} [h \text{ is extreme in } W \cup (R \cup \{h\})]. \end{aligned}$$

Here $[A]$ is the *indicator variable* for the event A that has value 1 if A holds and 0 otherwise.

Substituting Q for $R \cup \{h\}$ and using the fact that any set has at most δ extreme elements, we can rewrite this as

$$\begin{aligned} \binom{|H \setminus W|}{r} v_r &= \sum_{\substack{Q \subseteq H \setminus W \\ |Q|=r+1}} \sum_{h \in Q} [h \text{ is extreme in } W \cup Q] \\ &\leq \sum_{\substack{Q \subseteq H \setminus W \\ |Q|=r+1}} \delta = \binom{|H \setminus W|}{r+1} \delta. \end{aligned}$$

This yields $v_r \leq \delta(|H \setminus W| - r)/(r+1)$, and the lemma follows. □

4.3.2 The Trivial Algorithm

Given a violator space (H, V) of combinatorial dimension δ, the goal is to find a basis of H. For this, we assume availability of the following primitive, which we use for sets G of size at most δ.

Primitive 4.13 *Given $G \subseteq H$ and $h \in H \setminus G$, decide whether $h \in \mathsf{V}(G)$.*

Given this primitive, a basis of H can be found in a brute-force manner by going through all sets of size at most δ, testing each of them for being a basis of H. More generally, $B \subseteq G$ is a basis of G if and only if

$$\begin{array}{lcll} h & \in & \mathsf{V}(B \setminus \{h\}), & \text{for all } h \in B, \text{ and} \\ h & \notin & \mathsf{V}(B), & \text{for all } h \in G \setminus B. \end{array}$$

Consequently, the number of times the primitive needs to be invoked in order to find a basis of H ($|H| = N$) is at most

$$N \sum_{i=0}^{\delta} \binom{N}{i} = O(N^{\delta+1}).$$

The next two subsections show that this can be substantially improved.

4.3.3 Clarkson's First Algorithm

Fix a violator space (H, V) of combinatorial dimension δ, implicitly specified through Primitive 4.13. Clarkson's first algorithm calls Clarkson's second algorithm (`Basis2`) as a subroutine. Given $G \subseteq H$, both algorithms compute a basis B of G.

```
Basis1($G$):
  (* computes a basis $B$ of $G$ *)
  IF $|G| \leq 9\delta^2$ THEN
    RETURN Basis2($G$)
  ELSE
    $r := \lfloor \delta \sqrt{|G|} \rfloor$
    $W := \emptyset$
    REPEAT
      choose $R$ to be a random $r$-element subset of $G \setminus W$
      $C :=$ Basis2($W \cup R$)
      $V := \{h \in G \setminus C : h \in \mathsf{V}(C)\}$
      IF $|\mathsf{V}| \leq 2\sqrt{|G|}$ THEN
        $W := W \cup V$
      END
    UNTIL $V = \emptyset$
    RETURN $C$
  END
```

Assuming `Basis2` is correct, this algorithm is correct as well: if B is a basis of $W \cup R \subseteq G$ that in addition has no violators in G, B is a basis of G. Moreover, the algorithm augments the working set W at most δ times, which is guaranteed by the following observation.

Observation 4.14 *If $C \subseteq G$ and $G \cap \mathsf{V}(C) \neq \emptyset$, then $G \cap \mathsf{V}(C)$ contains at least one element from every basis of G.*

We remark that the condition resulting from Observation 4.14 can be used instead of locality in the definition of violator spaces. More precisely, the following is proved in [86].

Property 4.15 *Consider a set H and a mapping $\mathsf{V} : 2^H \to 2^H$, where*

(i) $\mathsf{V}(G) \cap G = \emptyset$ *for every $G \subseteq H$, and*

(ii) if a set $C \subseteq G$ satisfies $G \cap \mathsf{V}(C) \neq \emptyset$, then $G \cap \mathsf{V}(C)$ contains at least one element from every inclusion-minimal subset B of G with $\mathsf{V}(B) = \mathsf{V}(G)$.

Then (H, V) is a violator space.

Proof of Observation 4.14. Let B be a basis of G. Assuming

$$\emptyset = B \cap G \cap \mathsf{V}(C) = B \cap \mathsf{V}(C),$$

consistency yields $C \cap \mathsf{V}(C) = \emptyset$, implying $(B \cup C) \cap \mathsf{V}(C) = \emptyset$. From locality and monotonicity (Lemma 4.9), we get

$$\mathsf{V}(C) = \mathsf{V}(B \cup C) = \mathsf{V}(G),$$

meaning that $G \cap \mathsf{V}(G) = G \cap \mathsf{V}(C) = \emptyset$, a contradiction. □

It is also clear that `Basis2` is called only with sets of size at most $3\delta\sqrt{|G|}$. Finally, the expected number of iterations through the `REPEAT` loop is bounded by 2δ: by Lemma 4.12 (applied to $(G, \mathsf{V}|_G)$, where $\mathsf{V}|_G$ is V restricted to elements in G) and the Markov inequality, the expected number of calls to `Basis2` before we next augment W is bounded by 2.

Lemma 4.16 *Algorithm* `Basis1` *computes a basis of G with an expected number of at most $2\delta|G|$ calls to Primitive 4.13, and an expected number of at most 2δ calls to* `Basis2`*, with sets of size at most $3\delta\sqrt{|G|}$.*

4.3.4 Clarkson's Second Algorithm

This algorithm calls the trivial algorithm as a subroutine. Instead of adding violated constraints to a working set, it gives them larger probability of being selected in further iterations. Technically, this is done by maintaining G as a multiset, where $\mu(h)$ denotes the multiplicity of h (we set $\mu(F) := \sum_{h\in F}\mu(h)$). Sampling from G is done as before, imagining that G contains $\mu(h)$ copies of the element h.

```
Basis2($G$):
  (* computes a basis $B$ of $G$ *)
  IF $|G| \leq 6\delta^2$ THEN
    RETURN Trivial($G$)
  ELSE
    $r := 6\delta^2$
    REPEAT
      choose random $R \in \binom{G}{r}$
      $C :=$ Trivial($R$)
      $V := \{h \in G \setminus C \colon h \in \mathsf{V}(C)\}$
      IF $\mu(V) \leq \mu(G)/3\delta$ THEN
        $\mu(h) := 2\mu(h), \quad h \in V$
      END
    UNTIL $V = \emptyset$
    RETURN $C$
  END
```

Here `Trivial`(G) refers to calling the brute-force search algorithm described below Primitive 4.13.

Invoking Lemma 4.12 again (which also applies to multisets as we use them), we see that the expected number of calls to `Trivial` before we next reweight elements (a *successful* iteration), is bounded by 2. It remains to bound the number of successful iterations.

Lemma 4.17 *Let k be a positive integer. After $k\delta$ successful iterations, we have*

$$2^k \leq \mu(B) \leq |G|e^{k/3},$$

for every basis B of G. In particular, $k < 3 \ln |G|$.

Proof. Every successful iteration multiplies the total weight of elements in G by at most $(1 + 1/3\delta)$, which gives the upper bound (not only for $\mu(B)$ but actually for $\mu(G)$). For the lower bound, we use Observation 4.14 again to argue that each successful iteration doubles the weight of some element in B, meaning that after $k\delta$ iterations, one element has been doubled at least k times. Because the lower bound exceeds the upper bound for $k \geq 3 \ln |G|$, the bound on k follows. □

Summarizing, we get the following lemma.

Lemma 4.18 *Algorithm* `Basis2` *computes a basis of G with an expected number of at most $6\delta|G| \ln |G|$ calls to Primitive 4.13, and expected number of at most $6\delta \ln |G|$ calls to* `Trivial`*, with sets of size $6\delta^2$.*

4.3.5 Combining the Algorithms

Theorem 4.19 *Using a combination of the above two algorithms, a basis of H in a violator space (H, V) can be found by calling Primitive 4.13 expected*

$$O\left(\delta N + \delta^{O(\delta)}\right)$$

many times.

Proof. Using the above bound for the trivial algorithm, `Basis2` can be implemented to require an expected number of at most

$$O\left(\delta \log |G|(|G| + \delta^{O(\delta)})\right)$$

calls to the primitive. Applying this as a subroutine in `Basis1`(H) with $|H| = N$, $|G|$ is bounded by $3\delta\sqrt{N}$, and we get an overall expected complexity of

$$O\left(\delta N + \delta^2(\log N(\delta\sqrt{N} + \delta^{O(\delta)}))\right)$$

in terms of the number of calls to Primitive 4.13. The terms $\delta^2 \log N\ \delta\sqrt{N}$ and $\delta^2 \log N\ \delta^{O(\delta)}$ are asymptotically dominated by either δN or $\delta^{O(\delta)}$, and we get the simplified bound of $O\left(\delta N + \delta^{O(\delta)}\right)$. □

4.4 Grid USO as Models for Violator Spaces

We show in this section that the problem of finding the sink in a USO of the n-dimensional grid$([N], \Pi)$ can be reduced to the problem of finding the (unique) basis of a violator space (H, V) with $H = [N]$ and combinatorial dimension n. For a better understanding, we recommend to refresh the definitions of grid, subgrid, USO and outmap, given in Section 3.1 starting on page 30.

A vertex J in the grid graph is a subset of $[N]$ consisting of one element out of each block Π_i. There is an edge between two vertices J and J' if and only if J and J' differ in exactly one element. In order to reduce a grid USO ψ, we need the outmap $s_\psi(J)$ of a vertex J. It is important that a sink w.r.t. ψ has empty outmap value.

Let us fix a USO ψ of the n-dimensional grid graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Given a Π-valid subset $G \subseteq [N]$ (a subset consisting of at least one element out of each block Π_i), we define $\text{sink}(G) \in \mathcal{V}$ to be the unique sink vertex in the subgrid(G, Π). For a subset G that is not Π-valid, let

$$\bar{G} := \bigcup_{i\colon G \cap \Pi_i = \emptyset} \Pi_i.$$

Thus $\bar{G}$ is the set of elements occurring in blocks of Π disjoint from G.

Definition 4.20 *For $G \subseteq [N]$, define*

$$\mathsf{V}(G) = \begin{cases} s_\psi(\mathrm{sink}(G)), & \text{if } G \text{ is } \Pi\text{-valid} \\ \bar{G}, & \text{if } G \text{ is not } \Pi\text{-valid.} \end{cases}$$

Theorem 4.21 *The pair $([N], \mathsf{V})$ from Definition 4.20 is a violator space of combinatorial dimension n. Moreover, for all Π-valid $G \subseteq [N]$, the unique sink of the subgrid(G, Π) corresponds to the unique basis of G in $([N], \mathsf{V})$.*

Proof. For every $G \subseteq [N]$, consistency holds by definition of $\mathrm{sink}(G)$, $s_\psi(J)$ and $\bar{G}$. In order to prove locality for $F \subseteq G \subseteq [N]$, we look at three different cases. To shortcut notation, we abbreviate subgrid(F, Π) by $\mathcal{G}(F)$ for any $F \subseteq [N]$.

G is not Π-valid. Then, $F \subseteq G$ is not Π-valid either. The condition $\emptyset = G \cap \mathsf{V}(F) = G \cap \bar{F}$ means that F is disjoint from the same blocks as G. This implies $\bar{G} = \bar{F}$, hence $\mathsf{V}(G) = \mathsf{V}(F)$.

G and F are both Π-valid. Then $\mathcal{G}(F)$ is a nonempty subgrid of $\mathcal{G}(G)$, and $G \cap \mathsf{V}(F) = \emptyset$ means that the sink of $\mathcal{G}(F)$ has no outgoing edges into $\mathcal{G}(G)$. Thus the unique sink of $\mathcal{G}(F)$ is also a sink of $\mathcal{G}(G)$ and therefore the unique one. This means that $\mathrm{sink}(G) = \mathrm{sink}(F)$, from which $\mathsf{V}(G) = \mathsf{V}(F)$ follows.

G is Π-valid, F is not Π-valid. Then the condition $G \cap \mathsf{V}(F) = \emptyset$ can never be satisfied since $\mathsf{V}(F) = \bar{F}$ contains at least one full block Π_i, and $G \cap \Pi_i \neq \emptyset$.

Next we prove that a largest basis in $([N], \mathsf{V})$ has at most n elements. For this, let $G \subseteq [N]$ be a set of size larger than n. If G is Π-valid, we have

$$\mathsf{V}(G) := s_\psi(\mathrm{sink}(G)) = s_\psi(\mathrm{sink}(\mathrm{sink}(G))) =: \mathsf{V}(\mathrm{sink}(G)),$$

since $J = \mathrm{sink}(J)$ for any vertex J. This means that G has a subset of size n with the same violators, so G is not a basis.

If G is not Π-valid, we consider some subset B that contains exactly one element from every block intersected by G. By definition, we have $\bar{G} = \bar{B}$ and $\mathsf{V}(G) = \mathsf{V}(B)$. Since B has less than n elements, G cannot be a basis in this case, either.

It remains to prove that for G being Π-valid, the vertex sink(G) is the unique basis of G in $([N], \mathsf{V})$. We have already shown that $\mathsf{V}(G) = \mathsf{V}(\text{sink}(G))$ must hold in this case. Moreover, $\mathsf{V}(\text{sink}(G))$ contains no full block Π_i. On the other hand, any proper subset F of sink(G) is not Π-valid, so its violator set *does* contain at least one full block. It follows that $V(F) \neq V(\text{sink}(G))$, so sink$(G)$ is a basis of G. The argument is complete when we can prove that no other vertex $J \subseteq G$ is a basis of G. Indeed, such a vertex J is not a sink in $\mathcal{G}(G)$, meaning that $G \cap \mathsf{V}(J) \neq \emptyset$. This implies $\mathsf{V}(J) \neq \mathsf{V}(G)$. □

Note that the global sink of the grid USO corresponds to the unique n-element (and Π-valid) set B with $\mathsf{V}(B) = \emptyset$. This is exactly the set output by the call `Basis1`$([N])$ of Clarkson's algorithms, when we apply it to the violator space constructed in Definition 4.20.

Primitive 4.13 corresponds to one edge evaluation in the USO setting. With Theorem 4.19 we then get the following result:

Theorem 4.22 *The sink in a* USO *of the n-dimensional grid*$([N], \Pi)$ *can be found by evaluating expected $O\left(nN + n^{O(n)}\right)$ edges.*

For small n, the running time given in the theorem is faster than the one from the product algorithm [33, 32] which needs expected $O(n!N + H_N^n)$ edge evaluations, where H_N, as used earlier, is the N-th harmonic number. Via the reduction from PGLCP to grid USO (Section 3.1) this yields the fastest known algorithm for PGLCP with fixed number of blocks.

The PGLCP has in general a cyclic structure and therefore gives rise to a cyclic USO. The following lemma shows that a violator space obtained from a cyclic USO is cyclic as well, proving that grid USO (and PGLCP) can not be reduced to LP-type problems in general (since LP-type problems are equivalent to acyclic violator spaces [31, 30]).

Lemma 4.23 *The violator space arising from a cyclic grid* USO *via*

Definition 4.20 is cyclic.

Proof. Remember that given a grid vertex J and $j \in [N] \backslash J$, $J \triangleright j$ is the unique vertex $J' \subseteq J \cup \{j\}$ that is different from J. J' is the neighbor of J in direction j. Assume that the edge $\{J, J'\}$ is directed from vertex J' to vertex J in the grid USO ψ. We want to show that J' is locally smaller than J, meaning that $J' \cap \mathsf{V}(J) = \emptyset$ (Definition 4.7). This is indeed true, since by consistency, the only element that could possibly be in the intersection of J' and $\mathsf{V}(J)$ is j. But $j \in \mathsf{V}(J)$ would imply that $j \in s_\psi(J)$, meaning that J has an outgoing edge into direction j, a contradiction to the assumption that the edge is directed from J' to J.

We have shown above that every vertex J is a basis in the violator space. From the fact that the outmap function s_ψ is an injection [33, 32], it follows that no two vertices in the grid belong to the same equivalence class in the violator space. The result that an edge directed from J' to J implies that J' is locally smaller than J, means that a cycle in the grid USO immediately implies a cycle in the violator space, see Definition 4.8. $\square$

It is interesting that there are no cycles in a 2-dimensional grid USO [33, 32]. Whether the same is true for nondegenerate (no two bases have the same violators) violator spaces of combinatorial dimension 2 is an open question, see also [86].

Part II: Specializations

In this second part of the thesis, we present two specializations of the PGLCP. The first is achieved by restricting to a subclass of P-matrices called *hidden* K-*matrices* in Chapter 5. The GLCP with a hidden K-matrix has been proved to be solvable in polynomial time. Using the concept of Π-compatible LPs defined in Section 2.2, we give an alternative proof for this fact and strengthen the statement by showing that the arising grid USO is LP-induced and therefore acyclic. Further, a characterization for vertical block matrices that are not hidden K is derived. Towards the end of the chapter, we introduce a nontrivial and large subclass of matrices that are not hidden K and show that 3-dimensional P-matrices in this class give rise to a cyclic USO in general.

The second specialization is actually an application. We show that *simple stochastic games* (SSG) can be reduced to the PGLCP. It is open to characterize the P-matrices we get from the games, but we give arguments indicating that not all P-matrices are SSG-induced (that's why we speak of a specialization). At the end of the thesis, we give examples of SSGs, showing that the resulting matrix in the PGLCP is neither (transpose) hidden K nor *well-conditioned* in general.

Chapter 5

Hidden K-Matrices

We focus on the class of *hidden* K-*matrices* which also appear under the name *hidden Minkowski matrices* or *mime matrices* (see [93]). Hidden K-matrices form a proper subclass of P-matrices. The importance of the hidden K-matrix GLCP comes from the fact that the solution can be computed by a linear program. In order to show how this LP looks like, we need some definitions of matrix classes first, given in Section 5.1.

We then reprove that the hidden K GLCP can be solved by LP and concurrently show that the grid USO arising from a hidden K-matrix GLCP is LP-induced and therefore acyclic for any q (Section 5.2).

Early computer experiments suggested that it might hold that a right-hand side q making the USO cyclic (a *cyclic* q) exists if and only if the P-matrix is not hidden K. But this can't be true, since the set of hidden K-matrices is open as well as the set of P-matrices having a cyclic q (slightly perturbing such a matrix does not destroy the cycle). Indeed, exploiting this knowledge, we found a 3-dimensional non hidden K P-matrix that does not have a cyclic q, see Subsection 5.2.1.

However, the matrix M_1 found is very unstable, in the sense that if one perturbs it only slightly, then it either becomes hidden K, or there will pop up a cyclic q. Our updated conjecture is therefore the following.

Conjecture 5.1 *There exists a vector q such that the grid* USO *arising*

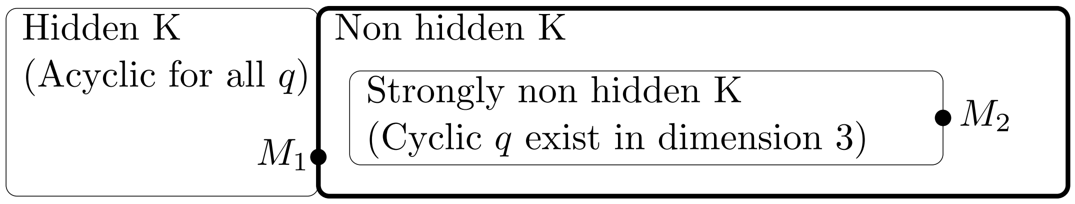


Figure 5.1: The picture shows the partition of the P-matrix class. The classes of hidden K-matrices and strongly non hidden K-matrices are open, while the class of non hidden K-matrices is closed. The text in brackets says what we know about the existence of cyclic q-vectors for the given classes. For matrix M_1 there exists no cyclic q, for M_2 there are cyclic q-vectors.

from PGLCP(G, q) *is cyclic if and only if the* P*-matrix* G *is* interior *non hidden* K*.*

A P-matrix G is *interior non hidden* K if all P-matrices in an arbitrarily small neighborhood of it are not hidden K as well. In other words, we can slightly perturb the matrix within the set of P-matrices and it stays non hidden K.

The problem we have is that we have no simple characterization for interior non hidden K-matrices, which would help us in order to analyze whether there exists a cyclic q. We present a characterization for non hidden K-matrices in Section 5.3 and a characterization for a nontrivial subclass of them, called *strongly* non hidden K-matrices in Subsection 5.3.1. Intuitively, the class of strongly non hidden K-matrices seems to be very close to the class of interior non hidden K-matrices. It is included in the latter class but we give a 4-dimensional P-matrix M_2 that is not strongly non hidden K but nevertheless interior non hidden K. We know that it is interior non hidden K because we have q-vectors generating cyclic orientations for this matrix M_2. Perturbing the matrix slightly won't destroy such a cycle, which proves that the matrix is interior non hidden K (by Theorem 5.20, which states that the hidden K-matrix GLCP gives rise to acyclic orientations). Moreover, we can show that there are matrices arbitrarily close to M_2 that are strongly non hidden K, so the situation is as depicted in Figure 5.1.

We conclude the chapter by proving in Subsection 5.3.2 that a cyclic

q always exists for 3-dimensional strongly non hidden K P-matrices.

5.1 Matrix Classes

The first class of matrices we look at is the class of square Z-matrices, for which the corresponding LCP is solvable in polynomial time [8].

Definition 5.2 *A square matrix is a* Z-matrix *if all off-diagonal entries are nonpositive.*

This generalizes to vertical block matrices.

Definition 5.3 *A vertical block matrix is a* vertical block Z-matrix *if all representative submatrices are* Z*-matrices.*

The Z-matrix GLCP is not known to be solvable in polynomial time. See for example [24] for more information about the Z-matrix GLCP. A proper subclass of (vertical block) Z-matrices is the following.

Definition 5.4 *A (vertical block) matrix is a* (vertical block) K-matrix *if it is a (vertical block)* Z*-matrix as well as a (vertical block)* P*-matrix.*

In [7], Borici and Lüthi give an application for the K-matrix LCP. We will need the following results about K-matrices, all of them proved for example in [26].

Lemma 5.5 *The following are equivalent for a square* Z*-matrix* $M \in \mathbb{R}^{n\times n}$*.*

(i) M *is a* K*-matrix.*

(ii) M^{-1} *exists and is nonnegative.*

(iii) There exists a positive vector $x \in \mathbb{R}^n$ *such that* $Mx > 0$*.*

We now define a class of matrices for which the corresponding LCP is solvable by linear programming. The class was introduced by Mangasarian [52] and generalized for vertical block matrices by Mohan and Neogy [61].

Definition 5.6 *A matrix $G \in \mathbb{R}^{(N-n)\times n}$ is called a* vertical block hidden Z-matrix *if there exists a square* Z*-matrix $X \in \mathbb{R}^{n\times n}$ and a vertical block* Z*-matrix $Y \in \mathbb{R}^{(N-n)\times n}$ of the same type as G, such that*

(i) $GX = Y$,

and if there exist nonnegative vectors $r \in \mathbb{R}^n$ and $s \in \mathbb{R}^N$ such that

(ii) $r^T X + s^T Y > 0$.

Along Mangasarian's lines, Mohan and Neogy show how the hidden Z-matrix GLCP can be solved by a linear program if the *witness* X of conditions (i) and (ii) above is known. The problem is that such an X is in general hard to find, since condition (ii) is nonlinear. But if the hidden Z-matrix G is also a P-matrix, then this is easy according to Characterizations 5.9 and 5.10 below, for which we need the following definition and theorem taken from [61].

Definition 5.7 *A matrix is a* vertical block hidden K-matrix *if it is a vertical block hidden* Z*-matrix as well as a vertical block* P*-matrix.*

Theorem 5.8 *A matrix $G \in \mathbb{R}^{(N-n)\times n}$ is a vertical block hidden* K*-matrix if and only if*

(i) G is vertical block hidden Z *and*

(ii) there exists a positive vector $x \in \mathbb{R}^n$ such that $Gx > 0$.

For the square case, Pang [71] derived the following characterization.

Characterization 5.9 *A matrix $M \in \mathbb{R}^{n\times n}$ is hidden* K *if and only if there exist* Z*-matrices $X \in \mathbb{R}^{n\times n}$ and $Y \in \mathbb{R}^{n\times n}$ such that* *(i)* $MX = Y$,

(ii) $Xe > 0$, *where* $e \in \mathbb{R}^n$ *is the all-one vector, and*

(iii) there exists a positive vector $x \in \mathbb{R}^n$ *such that* $Mx > 0$.

A positive vector x as in (iii) always exists for P-matrices [17], so (i) and (ii) are enough for a P-matrix to be hidden K. It is known that condition (iii) can be replaced by $Ye > 0$, see [93] and [65]. We generalize this to vertical block matrices.

Characterization 5.10 *A vertical block matrix* $G \in \mathbb{R}^{(N-n)\times n}$ *is hidden* K *if and only if there exist* Z*-matrices* X *and* Y *as in* (i) *of Definition 5.6 such that*

(i) $GX = Y$,

(ii) $Xe > 0$, *and*

(iii) $Ye > 0$, *where* $e \in \mathbb{R}^n$ *is the all-one vector.*

Proof. If G is vertical block hidden K, then one can prove [61] that there exists a positive vector $v \in \mathbb{R}^n$ such that for the witnesses X and Y in Definition 5.6 it holds that $Xv > 0$ and $Yv > 0$. Replacing X by XD and Y by YD where $D \in \mathbb{R}^{n\times n}$ is the diagonal matrix $diag(v^T)$, the Z-matrix property is preserved and conditions (i), (ii) and (iii) are satisfied.

For the other direction, assume that conditions $(i)-(iii)$ hold. Condition (ii) assures by Lemma 5.5 that the Z-matrix X is a P-matrix. Its transpose, X^T, is therefore also a P-matrix and as noted before, it is well-known that a positive vector $r \in \mathbb{R}^n$ exists such that $X^T r > 0$. Using (i) and setting $s \in \mathbb{R}^N$ to zero we get that G is a hidden Z-matrix by Definition 5.6. By Theorem 5.8 it remains to show that $Ye > 0$ implies existence of a vector $x \in \mathbb{R}^n$ such that $Gx > 0$. This is achieved by setting x to be the positive vector Xe, since $Gx = GXe = Ye > 0$. □

The nice thing about hidden K-matrices is that the witness P-matrix X can be found by solving a linear program. Since hidden K-matrices (and their witnesses) are closed under scaling, the strict positivity constraints in Characterization 5.10 can easily be realized by nonnegativity constraints using standard tricks [71], see also Section 5.3.

Pang showed that square hidden K-matrices are closed under principal pivot transforms [70]. We conclude this section by generalizing this result to vertical block matrices. Principal pivot transforms for vertical block matrices have been defined for example in [39].

Theorem 5.11 *Let G' be a vertical block matrix derived from G by doing a principal pivot transform. Then G' is hidden* K *if and only if G is hidden* K*.*

Proof. A PPT corresponds to an interchange of variables in the equation

$$w - Gz = q \tag{5.12}$$

of the GLCP. Let's without loss of generality look at the PPT that interchanges variables w_1^1 and z_1 and let's assume (by scaling G and q) that the entry G_{11}^1 is 1. The first row of (5.12) tells us that $w_1^1 - z_1 - G_{12}^1 z_2 - \ldots - G_{1n}^1 z_n = q_1^1$. After the PPT, the first row of (5.12) will be $z_1 - w_1^1 + G_{12}^1 z_2 + \ldots + G_{1n}^1 z_n = -q_1^1$ and all occurrences of z_1 in the other rows will be replaced by $w_1^1 - G_{12}^1 z_2 - \ldots - G_{1n}^1 z_n$ (we forget about the $-q_1^1$ term, since it affects the right-hand side of the GLCP and we are interested in G' only). The matrix G' derived by this PPT is therefore G^*T where G^* is G with the first row replaced by the first row of the $n \times n$ identity matrix and $T \in \mathbb{R}^{n \times n}$ is

$$\begin{pmatrix} 1 & -G_{12}^1 & -G_{13}^1 & \ldots & -G_{1n}^1 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix}.$$

The inverse of T is

$$\begin{pmatrix} 1 & G_{12}^1 & G_{13}^1 & \ldots & G_{1n}^1 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix}.$$

The first row of T^{-1} is exactly the first row of G. Let X be the hidden K witness of G and $Y = GX$ as in Characterization 5.10. Then $T^{-1}X$ is a hidden K witness for G' since

(i) $G'T^{-1}X = G^*TT^{-1}X = G^*X$ is a Z-matrix because the first row is the first row of X and the other rows are the same as in Y, and

(ii) $T^{-1}X$ is Z and $T^{-1}Xe > 0$ because the first row of $T^{-1}X$ is the first row of Y (Y is Z and $Ye > 0$), and because the other rows are the same as in X (X is Z and $Xe > 0$).

In general, let G be hidden K with witnesses X and Y and let G' be derived through a PPT interchanging variables z_i and w^i_j ($i \in [n]$ and $j \in [g_i]$). Then G' is hidden K with witnesses X' and Y', where X' is X with the ith row replaced by the jth row of Y^i, and Y' is Y with the jth row of Y^i replaced by the ith row of X. $\square$

5.2 Hidden K-Matrix GLCP and LP

Here comes the central theorem in hidden Z-matrix theory. It's proved in [61] by a straightforward generalization of the square case, one direction of which was proved in [19] and the other in [72], using the theory developed in Mangasarian's papers [52, 54, 55]. We state it for hidden K-matrices, i.e., we restrict it to P-matrices.

Theorem 5.13 *Let $G \in \mathbb{R}^{(N-n)\times n}$ be a vertical block P-matrix and q any $(N-n)$ vector. There is a vector $p \in \mathbb{R}^n$ (independent from q) such that the GLCP(G, q) has the same solution z as the linear program*

$$\begin{array}{ll} \textit{minimize} & p^Tz \\ \textit{subject to} & Gz + q \geq 0 \\ & z \geq 0 \end{array} \tag{5.14}$$

if and only if G is hidden K. If G is hidden K w.r.t. witness X, then p can be chosen as any n-vector for which $X^Tp > 0$.

As mentioned in the proof of Characterization 5.10, X is a P-matrix and it is well-known that a vector p with $X^Tp > 0$ thus exists. Note that the constraints of the LP (5.14) assure that w and z in the GLCP are nonnegative. The theorem states that for a hidden K-matrix G and suitable p, the LP gives us complementarity of w and z for free.

In this section, we reprove one direction of the theorem. Using the notion of a Π-compatible linear program derived in Section 2.2, we show that the GLCP(G,q) with hidden K-matrix G can be solved by (5.14). Moreover, we prove that the grid USO arising from the hidden K-matrix GLCP (Section 3.1) and the orientation of the following linear program are the same (the orientation derived from a linear program will be defined later).

$$\begin{array}{ll} \text{minimize} & q^T x \\ \text{subject to} & G^T x \leq p \\ & x \geq 0, \end{array}$$

where p is any vector such that $X^T p > 0$ (X a hidden K witness of G). This linear program is in fact the dual to (5.14) and it is in turn equivalent to

$$\begin{array}{ll} \text{minimize} & \begin{pmatrix} 0 \\ q \end{pmatrix}^T \begin{pmatrix} x' \\ x \end{pmatrix} \\ \text{subject to} & (I \mid G^T) \begin{pmatrix} x' \\ x \end{pmatrix} = p, \\ & \begin{pmatrix} x' \\ x \end{pmatrix} \geq 0, \end{array} \tag{5.15}$$

where 0 is the vector of all 0's of appropriate size and x' an n-vector of slack variables. We still require $X^T p > 0$.

We induce a partition Π on the set $[N]$ of column indices of $(I \mid G^T)$ by letting Π_i be the set of column indices corresponding to the rows of G^i (the ith block of G) plus the index i of column $I_{\{i\}}$ (the ith column of I). A representative set β is then a set having exactly one element out of each block of Π and $(I \mid G^T)_\beta$ is the matrix $(I \mid G^T)$ restricted to columns in β.

In the following, we will write A^T_α with the interpretation $(A^T)_\alpha$ for some matrix A. For every β, $(I \mid G^T)_\beta$ is equal to $(I_\alpha \| \bar{G}^T_{\bar{\alpha}})$ for some representative submatrix $\bar{G}$ of G and some $\alpha \subseteq [n]$. It's an easy observation that every representative submatrix $\bar{G}$ of the hidden K-matrix G is hidden K w.r.t. the same witness X. For such a submatrix, let $\bar{Y} = \bar{G}X$.

The hidden K property of G allows us to prove that the LP (5.15) is Π-compatible, see Section 2.2. This is because using (i) of Characterization 5.10, we can write

$$(I_\alpha \| \bar{G}^T_{\bar{\alpha}}) = (X^T)^{-1}(X^T_\alpha \| \bar{Y}^T_{\bar{\alpha}})$$

and derive that

$$((I \mid G^T)_\beta)^{-1}p = (I_\alpha \| \bar{G}^T_{\bar{\alpha}})^{-1}p = (X^T_\alpha \| \bar{Y}^T_{\bar{\alpha}})^{-1}X^T p > 0, \tag{5.16}$$

where the inequality follows from $X^T p > 0$ and the fact that $(X^T_\alpha \| \bar{Y}^T_{\bar{\alpha}})$ is a K-matrix ([17, Theorem 3.11.19]) and its inverse therefore nonnegative (Lemma 5.5). The proof that $(X^T_\alpha \| \bar{Y}^T_{\bar{\alpha}})$ is a K-matrix makes use of the existence of a positive vector v for which $Xv > 0$ and $\bar{Y}v > 0$ (as mentioned in the proof of Characterization 5.10). So $(X^T_\alpha \| \bar{Y}^T_{\bar{\alpha}})^T v > 0$ for all α implying that $(X^T_\alpha \| \bar{Y}^T_{\bar{\alpha}})^T$ is a K-matrix by Lemma 5.5 (it is obviously Z). Since K-matrices are closed under taking transposes (because P- and Z-matrices are), the result follows.

Observation 2.9 therefore tells us that the matrix $(I \mid G^T)$ has property P (which in this case is easy to see because G is a P-matrix) and that the representative sets β are exactly the bases of the LP (5.15). The bases of the LP are thus in one to one correspondence with the bases of the PGLCP(G, q) defined (via the PGLCP(H, q)) in Definition 2.7 on page 23.

An optimal solution x^* to (5.15) fulfills $x^*_\beta > 0$ for some representative set β, and by complementary slackness, an optimal solution z^* to (5.14) therefore fulfills the constraints corresponding to β with equality, thus giving us exactly the complementarity constraints for the GLCP. This finishes the alternative proof for the statement that the hidden K-matrix GLCP(G, q) can be solved by solving (5.14) and we also get

Corollary 5.17 *There is a vector $p \in \mathbb{R}^n$ such that the* LP *(5.15) with vertical block* P*-matrix G is* Π*-compatible if and only if G is hidden* K.

Proof. We have just seen the proof for the sufficiency part according to Equation (5.16). For the necessity part, assume that (5.15) is Π-compatible for some p but G is not hidden K. By complementary slackness, the solution of the dual to (5.15), which is (5.14), is also the solution to the GLCP(G, q) for all q. This is a contradiction to Theorem 5.13. □

We note that p can be chosen to be positive (because X^T is a P-matrix). For a square hidden K-matrix M^T, according to (5.16), p fulfills

$$(I_\alpha \| M_{\bar{\alpha}})^{-1} p > 0$$

for all α. Up to reshuffling rows and/or columns, this is equivalent to

$$\begin{pmatrix} I_{\alpha,\alpha} & -M_{\alpha,\bar{\alpha}}(M_{\bar{\alpha},\bar{\alpha}})^{-1} \\ 0_{\bar{\alpha},\alpha} & (M_{\bar{\alpha},\bar{\alpha}})^{-1} \end{pmatrix} p > 0$$

for all α, where 0 is the matrix of all zeros and the notation $A_{\alpha,\alpha'}$ for $\alpha, \alpha' \subseteq [n]$ denotes the matrix we get from $A \in \mathbb{R}^{n\times n}$ if we restrict to rows with indices in α and columns with indices in α'. The positive vector p thus satisfies the following property:

$$(M_{\alpha,\alpha})^{-1} p_\alpha > 0 \text{ for all } \alpha. \tag{5.18}$$

By making use of this positive vector p, Lemke's algorithm solves the LCP(M, q) in at most n iterations [14]. Our result states that if M^T is hidden K, then a positive vector p fulfilling (5.18) exists and the LCP(M, q) can therefore be solved in n steps of Lemke's algorithm. This has been proved before [73], and the converse is also known [62] (if a p fulfilling (5.18) exists, then M^T is hidden K).

We now proceed to show that the orientations underlying the hidden K GLCP and the LP (5.15) are the same. As noted above, the bases β of the LP are in one to one correspondence with the bases of the PGLCP(H, q) which implies that the underlying grid graph is the same. According to (3.3) on page 31, the orientation of edges incident to vertex β in the PGLCP(H, q) can be read off the $(N-n)$-dimensional vector $B(\beta)^{-1}q$, where a negative component corresponds to an outgoing edge and a positive component to an incoming edge. For the LP (5.15) it is known that the *reduced cost vector*

$$\bar{c}(\beta) := \begin{pmatrix} 0 \\ q \end{pmatrix}_{\bar{\beta}} - \begin{pmatrix} I \\ G \end{pmatrix}_{\bar{\beta},[n]} \left(\begin{pmatrix} I \\ G \end{pmatrix}_{\beta,[n]} \right)^{-1} \begin{pmatrix} 0 \\ q \end{pmatrix}_{\beta} \tag{5.19}$$

encodes the orientation at vertex β in the same way, see for example Chvátal's book [10]. It thus suffices to prove that $\bar{c}(\beta) = B(\beta)^{-1}q$. For that we fix a basis β and define k_I to be the number of columns of the identity matrix in $(I \,|\, G^T)_\beta$. Similarly, k_G is the number of columns of the matrix G^T in $(I \,|\, G^T)_\beta$. It holds that $k_I + k_G = n$. For notational convenience we assume without loss of generality that the matrix $(I \,|\, G^T)$ (and the vector $\begin{pmatrix} 0 \\ q \end{pmatrix}$) is sorted according to β, meaning that the first k_I columns in I (entries in the vector 0) and the first k_G columns in G^T (entries in q) are those covered by β. Moreover, we define $\overline{[k_I]}$ to be the set $[n] \setminus [k_I]$ and $\overline{[k_G]}$ to be the set $[N-n] \setminus [k_G]$. Note that $\overline{[k_I]}$ consists of k_G elements. In the following, we omit information concerning the dimensions of the identity- and zero-matrices, but the reader is able to figure out from the context of which dimension the matrices/vectors are. Equation (1) below holds because

$$\begin{pmatrix} I_{[k_I],[k_I]} & 0_{[k_I],\overline{[k_I]}} \\ G_{[k_G],[k_I]} & G_{[k_G],\overline{[k_I]}} \end{pmatrix}^{-1} = \begin{pmatrix} I_{[k_I],[k_I]} & 0_{[k_I],\overline{[k_I]}} \\ -(G_{[k_G],\overline{[k_I]}})^{-1} G_{[k_G],[k_I]} & (G_{[k_G],\overline{[k_I]}})^{-1} \end{pmatrix}.$$

$$\begin{aligned}
\bar{c}(\beta) &= \begin{pmatrix} 0 \\ q \end{pmatrix}_{\bar\beta} - \begin{pmatrix} I \\ G \end{pmatrix}_{\bar\beta,[n]} \left(\begin{pmatrix} I \\ G \end{pmatrix}_{\beta,[n]} \right)^{-1} \begin{pmatrix} 0 \\ q \end{pmatrix}_{\beta} \\
&= \begin{pmatrix} 0 \\ q \end{pmatrix}_{\bar\beta} - \begin{pmatrix} I \\ G \end{pmatrix}_{\bar\beta,[n]} \begin{pmatrix} I_{[k_I],[n]} \\ G_{[k_G],[n]} \end{pmatrix}^{-1} \begin{pmatrix} 0_{[k_I]} \\ q_{[k_G]} \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ q \end{pmatrix}_{\bar\beta} - \begin{pmatrix} I \\ G \end{pmatrix}_{\bar\beta,[n]} \begin{pmatrix} I_{[k_I],[k_I]} & 0_{[k_I],\overline{[k_I]}} \\ G_{[k_G],[k_I]} & G_{[k_G],\overline{[k_I]}} \end{pmatrix}^{-1} \begin{pmatrix} 0_{[k_I]} \\ q_{[k_G]} \end{pmatrix} \\
&\overset{(1)}{=} \begin{pmatrix} 0 \\ q \end{pmatrix}_{\bar\beta} - \begin{pmatrix} I \\ G \end{pmatrix}_{\bar\beta,[n]} \begin{pmatrix} 0_{[k_I]} \\ (G_{[k_G],\overline{[k_I]}})^{-1} q_{[k_G]} \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ q \end{pmatrix}_{\bar\beta} - \begin{pmatrix} 0_{\overline{[k_I]},[k_I]} & I_{\overline{[k_I]},\overline{[k_I]}} \\ G_{\overline{[k_G]},[k_I]} & G_{\overline{[k_G]},\overline{[k_I]}} \end{pmatrix} \begin{pmatrix} 0_{[k_I]} \\ (G_{[k_G],\overline{[k_I]}})^{-1} q_{[k_G]} \end{pmatrix} \\
&= \begin{pmatrix} 0_{\overline{[k_I]}} \\ q_{\overline{[k_G]}} \end{pmatrix} - \begin{pmatrix} (G_{[k_G],\overline{[k_I]}})^{-1} q_{[k_G]} \\ G_{\overline{[k_G]},\overline{[k_I]}} (G_{[k_G],\overline{[k_I]}})^{-1} q_{[k_G]} \end{pmatrix} \\
&= \begin{pmatrix} -(G_{[k_G],\overline{[k_I]}})^{-1} & 0_{\overline{[k_I]},\overline{[k_G]}} \\ -G_{\overline{[k_G]},\overline{[k_I]}} (G_{[k_G],\overline{[k_I]}})^{-1} & I_{\overline{[k_G]},\overline{[k_G]}} \end{pmatrix} \begin{pmatrix} q_{[k_G]} \\ q_{\overline{[k_G]}} \end{pmatrix} \\
&= \begin{pmatrix} -G_{[k_G],\overline{[k_I]}} & 0_{\overline{[k_I]},\overline{[k_G]}} \\ -G_{\overline{[k_G]},\overline{[k_I]}} & I_{\overline{[k_G]},\overline{[k_G]}} \end{pmatrix}^{-1} \begin{pmatrix} q_{[k_G]} \\ q_{\overline{[k_G]}} \end{pmatrix} \\
&= (H_{\bar\beta})^{-1} q = B(\beta)^{-1} q,
\end{aligned}$$

where the sorting of $(I \,|\, G^T)$ according to β resulted in H being $(G \,|\, I)$ in the PGLCP(H, q) with the first columns in G and $I \in \mathbb{R}^{(N-n)\times(N-n)}$ being the ones covered by β.

An LP orientation is acyclic, since following an outgoing edge means improving the objective function. We have thus proved

Theorem 5.20 *The grid-orientation (given by (3.3)) of the* GLCP *with hidden* K*-matrix G and right-hand side vector q is the same as the acyclic grid-orientation (given by the reduced cost vector in (5.19)) of the linear program (5.15).*

So the orientation induced by a hidden K-matrix GLCP is acyclic for every q. The converse is in general not true. In the following subsection we give a family of examples of P-matrices whose members are not hidden K and for which there is no q yielding a cyclic orientation. We refer to a q yielding a cyclic orientation as a *cyclic* q in the following.

5.2.1 A Family of Acyclic Non Hidden K-Matrices

We show that the following family of square matrices are P-matrices that are not hidden K and for which there is no cyclic q, if the entries a, b, c are chosen in a suitable way.

$$M_1 = \begin{pmatrix} 1 & a & 0 \\ 0 & 1 & b \\ c & 0 & 1 \end{pmatrix}$$

Matrices in this family generalize the matrix of [87] yielding a cyclic 3-cube USO (we get this matrix by setting $a = b = c = 2$). We restrict our analysis to P-matrices in this family. M_1 is a P-matrix if and only if its determinant $1 + abc$ is positive.

Lemma 5.21 *The* P*-matrix M_1 is a hidden* K*-matrix if and only if one of the following is true.*

1. *At least one of the three entries a, b, c is nonpositive.*
2. *All entries a, b, c are positive and $abc < 1$.*

Proof. If all three entries a, b, c are nonpositive, then M_1 is a Z-matrix and therefore a (hidden) K-matrix. For the other cases, we use that conditions (*i*) and (*ii*) of Characterization 5.9 are enough for a P-matrix to be hidden K. If exactly two entries are nonpositive (let's say a and b, wlog), then the following X is a hidden K witness:

$$\begin{pmatrix} 1 & 0 & 0 \\ bc & 1-bc & 0 \\ -c & 0 & 1+c \end{pmatrix}.$$

For exactly one entry being nonpositive (let's say a, wlog), the following X is a hidden K witness:

$$\begin{pmatrix} 1-ab & 0 & ab(1+c) \\ 0 & 1+b+bc & -b(1+c) \\ -c(1-ab) & 0 & 1+c \end{pmatrix}.$$

It remains to show that in the case where a, b and c are positive, M_1 is hidden K if and only if $abc < 1$. If $abc < 1$ then the following X is a witness for M_1 being hidden K.

$$\begin{pmatrix} 1+a+ab & -a(1+b+bc) & 0 \\ 0 & 1+b+bc & -b(1+c+ac) \\ -c(1+a+ab) & 0 & 1+c+ac \end{pmatrix}$$

If M_1 is hidden K, then (i) and (ii) of Characterization 5.9 imply (among others) the two constraints

$$\begin{aligned} x_{12} + ax_{22} &\leq 0 \\ x_{11} + x_{12} + x_{13} &> 0 \end{aligned}$$

for a Z-matrix X. The second constraint implies that $x_{11} + x_{12} > 0$ (since $x_{13} \leq 0$; note that $x_{12} \leq 0$ also implies $x_{11} > 0$). Together with the first constraint this yields $x_{11} > ax_{22}$. By the same argument we also get $x_{22} > bx_{33}$ and $x_{33} > cx_{11}$. Taking the three constraints together (and making use of $a, b, c > 0$) gives $abc < 1$. □

Lemma 5.22 *The following two statements are equivalent.*

1. *There is a right-hand side vector q such that the* LCP USO *(given by (3.3)) induced by the* P*-matrix M_1 and q is cyclic.*

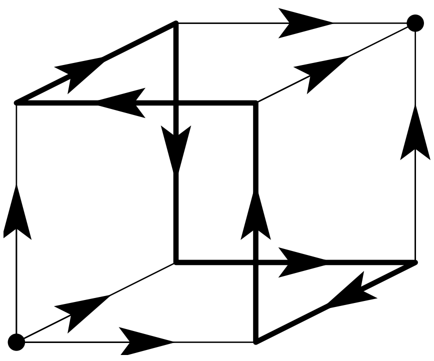


Figure 5.2: The cyclic USO of the 3-cube.

2. All entries a, b, c of M_1 are positive and $abc > 1$.

Proof. The only cyclic USO of the 3-dimensional cube (up to isomorphic ones derived by rotation and reflection of the cube, see also Subsection 3.3.3) is the one given in Figure 5.2 [87]. Note that the orientation has the following two properties.

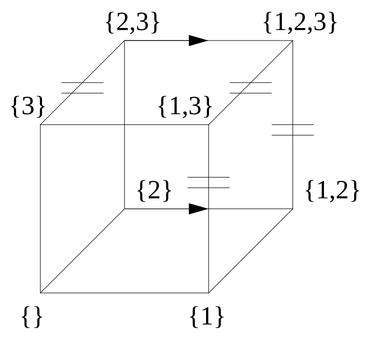
(P1) All 2-dimensional faces are *bows*, i.e. the sink and the source in these faces are neighbors.

(P2) In every dimension, there are three edges having the same orientation and one edge going into the opposite direction.

Let $q = (q_1, q_2, q_3)^T$. The orientation induced by M_1 is determined by the following table, where $B(\alpha)$ are the basis matrices of the PLCP(M_1, q) as defined in Definition 1.5 on page 14.

α	$(B(\alpha)^{-1}q)_1$	$(B(\alpha)^{-1}q)_2$	$(B(\alpha)^{-1}q)_3$
$\{1,2,3\}$	q_1	q_2	q_3
$\{2,3\}$	$-q_1$	q_2	$-cq_1 + q_3$
$\{1,3\}$	$q_1 - aq_2$	$-q_2$	q_3
$\{3\}$	$-q_1 + aq_2$	$-q_2$	$-cq_1 + acq_2 + q_3$
$\{1,2\}$	q_1	$q_2 - bq_3$	$-q_3$
$\{2\}$	$-q_1$	$bcq_1 + q_2 - bq_3$	$cq_1 - q_3$
$\{1\}$	$q_1 - aq_2 + abq_3$	$-q_2 + bq_3$	$-q_3$
$\{\}$	$-q_1 + aq_2 - abq_3$	$-bcq_1 - q_2 + bq_3$	$cq_1 - acq_2 - q_3$

The entry q_2 at position $(\{2,3\}, (B(\alpha)^{-1}q)_2)$ for example tells us that $(B(\{2,3\})^{-1}q)_2 = q_2$. That is, according to (3.4), the edge starting



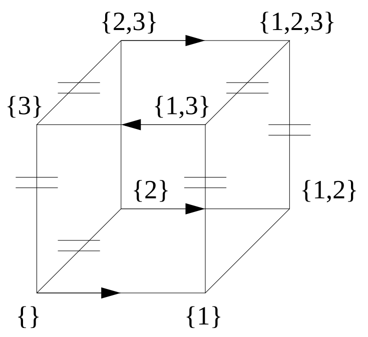


Figure 5.3: Construction of a cyclic orientation induced by M_1.

at vertex $\{2,3\}$ is outgoing into the direction of the second dimension if $q_2 < 0$ and incoming if $q_2 > 0$ (the ith dimension is the one along which we go when switching from vertex α to $\alpha \oplus i$).

The entries in the row where $\alpha = \{\}$ are missing a positive (since M_1 is a P-matrix) factor $1/(1+abc)$. But since we will be interested in the sign of the entries only, we left it away.

How does a q yielding a cycle look like? We can assume that $q_1 > 0$, otherwise we look at $-q$ which yields the orientation with every edge reversed and thus preserves cycles ($q_1 = 0$ would yield a degenerate orientation). According to the table above, this fixes the orientation of the two edges in Figure 5.2.1 left. Also from the table, we can read that the edges from $\{1,2,3\}$ and $\{2,3\}$ into direction 2 have the same orientation. The same holds for the edges from $\{1,2,3\}$ and $\{1,3\}$ into direction 3. This is indicated by an equality sign on the edges.

Properties (P1) and (P2) let us fix the edges from $\{1,3\}$ to $\{3\}$ and $\{1\}$ to $\{\}$ in Figure 5.2.1 right. (P1) tells us that the edge from $\{3\}$ to $\{\}$ should have the same orientation as the one from $\{1,3\}$ to $\{1\}$. By (P1) again, the edge from $\{1,2\}$ to $\{1\}$ has opposite direction to the edges with equality signs in dimension 2, which is equivalent (by (P2)) to stating that the edge from $\{2\}$ to $\{\}$ gets an equality sign.

If the edges with equality signs in dimension 3 would all be directed downwards (and the unique one without the equality sign upwards), then it wouldn't be possible to set sink and source antipodal, because the only possibilities left for the sink would be $\{1\}$ and $\{1,2\}$ and the vertices antipodal to them would both already have an incoming edge and could therefore not be the source. However, in order to get a 3-dimensional cyclic USO it is necessary to have sink and source antipodal. Thus, the edges with equality sign in dimension 3 have to be oriented upwards. There is only one way to complete the USO such that sink and source are antipodal, resulting in the USO of Figure 5.2.

The USO constrains the entries in the above table. Leaving away redundant constraints (the constraint for an edge in one direction is the constraint for the edge into the opposite direction multiplied by -1), the table can be shrunken to the following system of constraints which is feasible if and only if there exists a q yielding a cycle.

$$\begin{array}{rcll}
q & > & 0 & \{1,2,3\} \text{ is the sink} \\
q_1 - aq_2 & < & 0 & \text{edge } (\{1,3\},\{3\}) \\
q_2 - bq_3 & < & 0 & \text{edge } (\{1,2\},\{1\}) \\
q_3 - cq_1 & < & 0 & \text{edge } (\{2,3\},\{2\}) \\
q_1 & > & 0 & \text{edge } (\{1,2\},\{2\}) \\
q_2 & > & 0 & \text{edge } (\{2,3\},\{3\}) \\
q_3 & > & 0 & \text{edge } (\{1,3\},\{1\}) \\
q_1 - aq_2 + abq_3 & > & 0 & \{\} \text{ is the source} \\
bcq_1 + q_2 - bq_3 & > & 0 & \{\} \text{ is the source} \\
-cq_1 + acq_2 + q_3 & > & 0 & \{\} \text{ is the source}
\end{array}$$

One can check that given the first four constraints (implying $a, b, c > 0$), the last six are redundant. The proof of the lemma is therefore finished once we show that the system

$$\begin{array}{rcl}
q & > & 0 \\
q_1 - aq_2 & < & 0 \\
q_2 - bq_3 & < & 0 \\
q_3 - cq_1 & < & 0
\end{array} \qquad (5.23)$$

is feasible if and only if the system

$$\begin{array}{rcl}
a, b, c & > & 0 \\
abc & > & 1
\end{array} \qquad (5.24)$$

is feasible. It's easy to see that (5.23) implies (5.24). For the other direction, having $a, b, c > 0$ and $abc > 1$, the following q is feasible for (5.23):

$$q = \begin{pmatrix} ab + a + 1 \\ bc + b + 1 \\ ac + c + 1 \end{pmatrix}.$$

□

Lemmas 5.21 and 5.22 imply that P-matrix M_1 with $a, b, c > 0$ and $abc = 1$ is a non hidden K-matrix for which there does not exist a q yielding a cyclic orientation. This M_1 is a boundary case, in the sense that if one perturbs its entries only a little bit, it either becomes hidden K or there appears a cycle for some q.

In the next section, we derive a characterization for vertical block non hidden K-matrices, a generalization of a characterization first shown in [65]. By strengthening the conditions in the characterization, we are then able to determine a nontrivial subclass of 3-dimensional P-matrices for which a cyclic q exists.

5.3 Non Hidden K-Matrices

We first state the linear program that arises from Characterization 5.10. It is feasible if and only if G is hidden K.

$$\begin{array}{lll} \text{maximize} & 0 & \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} X_{ij} \geq 1 & \text{for all } i \\ & \displaystyle\sum_{l=1}^{n} G^i_{kl} X_{lj} \leq 0 & \text{for all } i, k \text{ and } j \neq i \\ & \displaystyle\sum_{j=1}^{n}\sum_{l=1}^{n} G^i_{kl} X_{lj} \geq 1 & \text{for all } i, k \\ & X_{ij} \leq 0 & \text{for all } i \neq j \end{array} \tag{5.25}$$

By "for all i, k" we mean all $i \in [n]$ and, depending on i, all $k \in [g_i]$. The index j is also in $[n]$. Note that by scaling the witness X in

Characterization 5.10, the constraints $Xe > 0$ ($Ye > 0$) are feasible if and only if the constraints $Xe \geq 1$ ($Ye \geq 1$) are feasible. The second set of constraints in the above LP assures that $Y^i_{kj} \leq 0$ for all i, k and $j \neq i$, implying that Y is a Z-matrix, and the third set of constraints assures that $Ye > 0$. The linear program (5.25) is thus feasible if and only if $G \in \mathbb{R}^{(N-n)\times n}$ is a hidden K-matrix. Its dual is derived by linking the first n constraints to variables R_{ii} for all i, the next $(N-n)(n-1)$ constraints to variables S^i_{kj} for all i, k and $j \neq i$, and the next $(N-n)$ constraints to variables S^i_{ki} for all i, k. The S-variables thus form a vertical block matrix of the same type as G.

$$\begin{array}{lll}
\min & \sum_{i=1}^{n}(R_{ii} + \sum_{k=1}^{n} S^i_{ki}) & \\
\text{s.t.} & R_{ii} + \sum_{l\neq j}\sum_{k=1}^{g_l} G^l_{ki}S^l_{kj} + \sum_{l=1}^{n}\sum_{k=1}^{g_l} G^l_{ki}S^l_{kl} \leq 0 & \text{for all } i \neq j \\
& R_{ii} + \sum_{l\neq i}\sum_{k=1}^{g_l} G^l_{ki}S^l_{ki} + \sum_{l=1}^{n}\sum_{k=1}^{g_l} G^l_{ki}S^l_{kl} = 0 & \text{for all } i \\
& R_{ii} \leq 0, S^i_{ki} \leq 0 & \text{for all } i, k \\
& S^i_{kj} \geq 0 \quad \text{for all } i, k \text{ and } j \neq i. &
\end{array} \tag{5.26}$$

The first $n^2 - n$ constraints in (5.26) correspond to the variables X_{ij}, $i \neq j$, in (5.25) and the next n constraints to the variables X_{ii}.

We replace variables R_{ii} by $-R_{ii}$ and variables S^i_{kj} by $-S^i_{kj}$ for all i, j, k, move some terms from the right double sum to the left double sum, and get

$$\begin{array}{lll}
\max & \sum_{i=1}^{n}(R_{ii} + \sum_{k=1}^{n} S^i_{ki}) & \\
\text{s.t.} & R_{ii} + \sum_{l=1}^{n}\sum_{k=1}^{g_l} G^l_{ki}S^l_{kj} + \sum_{l\neq j}\sum_{k=1}^{g_l} G^l_{ki}S^l_{kl} \geq 0 & \text{for all } i \neq j \\
& R_{ii} + \sum_{l=1}^{n}\sum_{k=1}^{g_l} G^l_{ki}S^l_{ki} + \sum_{l\neq i}\sum_{k=1}^{g_l} G^l_{ki}S^l_{kl} = 0 & \text{for all } i \\
& R_{ii} \geq 0, S^i_{ki} \geq 0 & \text{for all } i, k \\
& S^i_{kj} \leq 0 & \text{for all } i, k \text{ and } j \neq i.
\end{array}$$

By defining R' and S' to be the matrices with $R'_{ij} := R_{ii}$ and

$$S'^i_{kj} := \begin{cases} S^i_{ki} & \text{if } j \neq i \\ 0 & \text{if } j = i \end{cases}$$

for all i, j, k, the previous LP can be written as

$$\begin{array}{lll} \text{maximize} & \sum_{i=1}^{n}(R_{ii} + \sum_{k=1}^{n}(S^i_{ki} + S'^i_{ki})) & \\ \text{subject to} & R' + G^T(S + S') = T & \\ & R_{ii} \geq 0, S^i_{ki} \geq 0, T_{ii} = 0 & \text{for all } i, k \\ & S^i_{kj} \leq 0, T_{ij} \geq 0 & \text{for all } i, k \text{ and } j \neq i. \end{array}$$

Replacing $S + S'$ by a new matrix S and merging the constraints for the matrix R' and the slack matrix T into constraints for the matrix R, we can equivalently state this as

$$\begin{array}{lll} \text{maximize} & \sum_{i=1}^{n}(R_{ii} + \sum_{k=1}^{n} S^i_{ki}) & \\ \text{subject to} & R + G^T S = 0 & \\ & R_{ii} \geq 0, S^i_{ki} \geq 0 & \text{for all } i, k \\ & R_{ij} \leq R_{ii}, S^i_{kj} \leq S^i_{ki} & \text{for all } i, k \text{ and } j \neq i. \end{array} \tag{5.27}$$

Since (5.27) is always feasible, it is unbounded if and only if (5.25) is infeasible (a standard fact in linear programming theory) which is in turn the case if and only if G is not hidden K. We therefore get the following characterization for non hidden K-matrices:

Characterization 5.28 *A vertical block matrix $G \in \mathbb{R}^{(N-n)\times n}$ of type $(g_1, \ldots, g_n)$ is* not hidden K *if and only if*

(i) $R + G^T S = 0$ *for some matrix $R \in \mathbb{R}^{n\times n}$ and some vertical block matrix S of the same size and type as G with*

(ii) $R_{ii} \geq 0, S^i_{ki} \geq 0, R_{ii} \geq R_{ij}, S^i_{ki} \geq S^i_{kj}$ *for all $i, j \in [n]$ and $k \in [g_i]$, and*

(iii) at least one diagonal element of R or at least one diagonal element of some representative submatrix of S is nonzero.

For the remainder of this chapter, we focus on square matrices. The specialization of Characterization 5.28 to square matrices is:

Characterization 5.29 *A matrix $M \in \mathbb{R}^{n\times n}$ is* not hidden K *if and only if*

(i) $R + M^T S = 0$ *for some matrices R and S in $\mathbb{R}^{n\times n}$ with*

(ii) $R_{ii} \geq 0, S_{ii} \geq 0, R_{ii} \geq R_{ij}, S_{ii} \geq S_{ij}$ *for all i, j, and*

(iii) at least one diagonal element of R or S is nonzero.

This is the same characterization for square matrices as the one given in [65], except that condition (*iii*) is replaced by:

(*iii′*) R and S are not both zero.

However, if R and S satisfy (*i*), (*ii*) and (*iii′*), then the matrices R' and S' we get from R and S by post-multiplying them with the square $n \times n$ matrix with all off-diagonal entries being -1 and all diagonal entries being n satisfy (*i*), (*ii*) and (*iii*). The two characterizations are therefore equivalent.

The characterization of [65] has been derived using different methods than dualization of linear programs. Characterization 5.29 can be tightened for *minimal* non hidden K P-matrices.

Definition 5.30 *A square non hidden* K *matrix is called* minimal *if all proper principal submatrices of all principal pivot transforms are hidden K-matrices.*

Given a P-matrix M that is minimal non hidden K with witnesses R and S as in Characterization 5.29, we first show that S does not have a row with all entries being zero. For that remember the notation $A_{\alpha,\alpha'}$ introduced on page 88. Assume for contradiction that the ith row of S is zero. Then $R_{[n]\setminus i,[n]\setminus i}$ and $S_{[n]\setminus i,[n]\setminus i}$ are witnesses for $M_{[n]\setminus i,[n]\setminus i}$ being non hidden K and therefore yield a contradiction to minimality, given that at least one diagonal element of $R_{[n]\setminus i,[n]\setminus i}$ or $S_{[n]\setminus i,[n]\setminus i}$ is nonzero. This is indeed the case: assume for contradiction that all diagonal elements in R and S are zero, except R_{ii}. By (*ii*) in Characterization 5.29, the ith column of S is nonpositive ($S_{\{i\}} \leq 0$ and $S_{ii} = 0$), and $R_{\{i\}} = -M^T S_{\{i\}} \leq 0$ at all components except R_{ii}. M^T

then reverses the sign of the nonzero (because $R_{ii} \neq 0$) vector $S_{\{i\}}$, meaning that $(S_{\{i\}})_j(M^T S_{\{i\}})_j \leq 0$ for all $j \in [n]$. The existence of this *sign reversing vector* is well-known [17] to contradict the P-matrix property of M^T and with this also of M. We conclude that S has no zero row.

All we needed to prove that S has no zero row was minimality and the P-matrix property of M which implies that no sign reversing vector exists. By pre-multiplying $R + M^T S = 0$ with $(M^{-1})^T$ we get $S + (M^{-1})^T R = 0$. Since the inverse of a minimal P-matrix is again a minimal P-matrix (the inverse is derived by a PPT), and since R and S share the same properties, this situation is completely symmetric to the original one and we can prove along the same lines that R can not have a zero row as well.

By post-multiplying R and S with the square $n \times n$ matrix with all off-diagonal entries being -1 and all diagonal entries being n, we get new R and S matrices with all diagonal entries being strictly positive, while R and S keep the properties of Characterization 5.29. We thus derive

Lemma 5.31 *For a minimal non hidden* K P*-matrix* $M \in \mathbb{R}^{n\times n}$ *the witnesses* $R \in \mathbb{R}^{n\times n}$ *and* $S \in \mathbb{R}^{n\times n}$ *as in Characterization 5.29 do not have zero rows. Moreover, there exist* R *and* S *matrices with the following properties.*

(i) $R + M^T S = 0$,

(ii) $R_{ii} > 0$, $S_{ii} > 0$, *for all* i *and*

(iii) $R_{ii} \geq R_{ij}$, $S_{ii} \geq S_{ij}$, *for all* i, j.

We note that it is known that every 3-dimensional P-matrix that is not hidden K is minimal, since all P-matrices of dimension at most 2 are hidden K. For general minimal non hidden K P-matrices it is not possible to find witnesses R and S satisfying the constraints (iii) with strict inequalities. We give an example of a minimal non hidden K P-matrix for which no R and S with strict constraints exist in the next subsection.

5.3.1 A Nontrivial Subclass

The example in Subsection 5.2.1 proves that there are non hidden K-matrices for which there exists no cyclic q. The question is how to tell matrices with cyclic q-vectors and others apart. Looking at Lemma 5.31, a first idea might be to tighten all the constraints in Characterization 5.28. We define the following nontrivial and large subclass of non hidden K-matrices.

Definition 5.32 *A matrix $M \in \mathbb{R}^{n\times n}$ is* strongly not hidden K *if*

(i) $R + M^T S = 0$ *for some matrices R and S in $\mathbb{R}^{n\times n}$ with*

(ii) $R_{ii} > 0, S_{ii} > 0, R_{ii} > R_{ij}, S_{ii} > S_{ij}$ *for all i and j, $i \neq j$.*

A non hidden K*-matrix not fulfilling* (*i*) *and* (*ii*) *is* weakly not hidden K*.*

One might hope that strongly not hidden K-matrices are exactly those non hidden K-matrices which have a cyclic q. Although this could be true in 3 dimensions (see also the next subsection) where we don't have a counterexample, it is not true in general: the 4-dimensional P-matrix

$$M_2 = \begin{pmatrix} 45 & -44 & 81 & -9 \\ 76 & 17 & -21 & 81 \\ -76 & 9 & 83 & -81 \\ -83 & -79 & 61 & 53 \end{pmatrix}$$

is minimal weakly not hidden K (checked by computer) and there exist cyclic q-vectors, an example is

$$q = \begin{pmatrix} 1 \\ 6 \\ 3 \\ 2 \end{pmatrix}.$$

The matrix M_2 was found by Maple doing a brute force search (using the LP formulation arising from Lemma 5.36 below) and q was found by a C++ program computing all the possible cells generated by the hyperplanes containing the 3-dimensional intersections of the complementary 4-dimensional cones of the PGLCP (only later we were pointed to an existing software doing this: TOPCOM, written by Jörg Rambau [76] and available on the web).

We need some definitions in order to give a certificate for the fact that the above matrix M_2 is weakly not hidden K.

Definition 5.33 *A matrix $M \in \mathbb{R}^{n\times n}$ is a* Z°-matrix *if all off-diagonal entries are negative.*

Definition 5.34 *A matrix $M \in \mathbb{R}^{n\times n}$ is* strongly hidden K *if there exist* Z°*-matrices $X \in \mathbb{R}^{n\times n}$ and $Y \in \mathbb{R}^{n\times n}$ such that*

(i) $MX = Y$,

(ii) $Xe > 0$, *and*

(iii) $Ye > 0$, *where* $e \in \mathbb{R}^n$ *is the all-one vector.*

One can prove [65] that a matrix is hidden K if and only if it is strongly hidden K, so the two classes are in fact the same.

Definition 5.35 *A matrix $M \in \mathbb{R}^{n\times n}$ is* weakly hidden K *if there exist* Z*-matrices $X \in \mathbb{R}^{n\times n}$ and $Y \in \mathbb{R}^{n\times n}$, X nonzero, such that*

(i) $MX = Y$,

(ii) $Xe \geq 0$, *and*

(iii) $Ye \geq 0$, *where* $e \in \mathbb{R}^n$ *is the all-one vector.*

Lemma 5.36 *A matrix $M \in \mathbb{R}^{n\times n}$ is strongly not hidden* K *if and only if it is not weakly hidden* K*. The matrices in $\mathbb{R}^{n\times n}$ are therefore partitioned into three classes:*

(1) (strongly) hidden K*-matrices,*

(2) strongly not hidden K*-matrices,*

(3) matrices that are weakly hidden K *as well as weakly not hidden* K.

Proof. According to Definition 5.35, it is easy to see that the matrix M is not weakly hidden K if and only if the following LP is bounded (note that it is always feasible).

$$\begin{array}{ll} \text{maximize} & \sum_{i=1}^{n} X_{ii} + Y_{ii} \\ \text{subject to} & MX = Y \\ & Xe \geq 0 \\ & Ye \geq 0 \\ & X_{ij} \leq 0, Y_{ij} \leq 0, \text{ for all } i \text{ and } j \neq i \end{array} \tag{5.37}$$

On the other hand, according to Definition 5.32, M is strongly not hidden K if and only if the following LP is feasible (it is always bounded).

$$\begin{array}{lll} \text{minimize} & 0 & \\ \text{subject to} & -\sum_{k=1}^{n} M_{ki}S_{ki} = 1 + a_i & \text{for all } i \\ & -\sum_{k=1}^{n} M_{ki}S_{kj} \leq a_i & \text{for all } i \text{ and } j \neq i \\ & S_{ii} = 1 + b_i & \text{for all } i \\ & S_{ij} \leq b_i & \text{for all } i \text{ and } j \neq i \\ & a_i \geq 0, b_i \geq 0 & \text{for all } i \end{array} \tag{5.38}$$

One can check that these two LPs are dual to each other, the constraints $MX = Y$, $Xe \geq 0$, $Ye \geq 0$ in (5.37) correspond to variables $-S$, $-a$, $-b$ in (5.38). From LP theory, we get that (5.37) is bounded if and only if (5.38) is feasible, which proves the lemma. □

The matrix M_2 above belongs to class (3) of Lemma 5.36. It is weakly not hidden K:

$$\begin{pmatrix} 9576 & -18088 & 9576 & 9576 \\ 2142 & 2142 & -62695 & 2142 \\ -2646 & 19754 & 87300 & -2646 \\ 10206 & -19278 & 10206 & 10206 \end{pmatrix} + M_2^T \begin{pmatrix} 0 & 0 & -827 & 0 \\ -126 & 0 & 0 & -126 \\ 0 & -238 & 0 & 0 \\ 0 & 0 & -333 & 0 \end{pmatrix} = 0.$$

And it is weakly hidden K:

$$M_2 \begin{pmatrix} 81 & 0 & 0 & -81 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -76 & 0 & 0 & 76 \end{pmatrix} = \begin{pmatrix} 4329 & 0 & 0 & -4329 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -10751 & 0 & 0 & 10751 \end{pmatrix}.$$

These witness matrices can be found using the Maple hidden K library provided by the author. Also, the above q can be tested to be cyclic for M_2. This is done by solving a linear program that tries to find a flow of maximal value in the corresponding USO between source and sink. The value of the flow is infinite if there exists a cycle.

5.3.2 Dimension 3

We prove that 3-dimensional strongly non hidden K P-matrices have a cyclic q.

Theorem 5.39 *Let $M \in \mathbb{R}^{3\times 3}$ be a non hidden* K P*-matrix for which matrices $R \in \mathbb{R}^{3\times 3}$ and $S \in \mathbb{R}^{3\times 3}$ with*

(i) $R + M^T S = 0$,

(ii) $R_{ii} > 0$, $S_{ii} > 0$,

(iii) $R_{ii} > R_{ij}$ *and* $S_{ii} > S_{ij}$

for all i and j, $i \neq j$ exist. Then S can be assumed to be invertible and $q := (S^T)^{-1}e$ is cyclic (e is the 3-dimensional all-one vector), meaning that the 3-cube USO *arising from the* PLCP(M, q) *has a cycle.*

Proof. Note first that the properties of R and S allow us to perturb S by small amounts (and adjust R) while preserving $R + M^T S = 0$ and the strict inequality constraints. This implies existence of an invertible S.

By pre-multiplying the equation $w - Mz = q$ in the PLCP with S^T, we arrive at $S^T w - S^T M z = S^T q$ which evaluates according to (i) to $S^T w + R^T z = S^T q$. This is not a standard PLCP anymore, but we still have bases $\alpha \subseteq [n]$ (determining which variables are to be set to zero in order to fulfill the complementarity conditions) and basis matrices $C(\alpha) := (S^T_\alpha \| R^T_{\bar\alpha})$. The underlying USO stays the same, since

$$\begin{aligned} C(\alpha)^{-1} S^T q &= ((S^T)^{-1} C(\alpha))^{-1} q \\ &= ((S^T)^{-1} (S^T_\alpha \| R^T_{\bar\alpha}))^{-1} q \\ &= (I_\alpha \| - M_{\bar\alpha})^{-1} q \\ &= B(\alpha)^{-1} q, \end{aligned}$$

or simply because for bases in the two settings it holds that $S^T B(\alpha) = C(\alpha)$ which implies $C(\alpha)^{-1} S^T q = B(\alpha)^{-1} q$ (it's clear that $C(\alpha)$ is regular).

Up to now, we haven't used the fact that we are in 3 dimensions. But now we fix an $\alpha \subseteq [3]$ and define $C := C(\alpha)$. The matrix C consists of some columns out of S^T and some out of R^T. We label its entries as follows

$$\begin{pmatrix} C_{11} & C_{21} & C_{31} \\ C_{12} & C_{22} & C_{32} \\ C_{13} & C_{23} & C_{33} \end{pmatrix},$$

because this way, the transposes are directly taken care of, and we have $C_{ii} > 0$ and $C_{ii} > C_{ij}$ for all $i \neq j$. We look at the orientations of edges incident to the vertex α if q is $(S^T)^{-1}e$. We define $\bar{q} := C(\alpha)^{-1}S^T q = C^{-1}e$. By Cramer's rule, the components of $\bar{q}$ can be computed as

$$\bar{q}_1 = \frac{\det \begin{pmatrix} 1 & C_{21} & C_{31} \\ 1 & C_{22} & C_{32} \\ 1 & C_{23} & C_{33} \end{pmatrix}}{\det(C)},$$

$$\bar{q}_2 = \frac{\det \begin{pmatrix} C_{11} & 1 & C_{31} \\ C_{12} & 1 & C_{32} \\ C_{13} & 1 & C_{33} \end{pmatrix}}{\det(C)},$$

$$\bar{q}_3 = \frac{\det \begin{pmatrix} C_{11} & C_{21} & 1 \\ C_{12} & C_{22} & 1 \\ C_{13} & C_{23} & 1 \end{pmatrix}}{\det(C)}.$$

Now take the two entries $\bar{q}_1$ and $\bar{q}_2$. Expanding the numerator in $\bar{q}_1$ along column 2 yields

$$C_{21}\underbrace{(C_{32} - C_{33})}_{<0} + C_{22}\underbrace{(C_{33} - C_{31})}_{>0} + C_{23}(C_{31} - C_{32}) \tag{5.40}$$

and expanding the numerator in $\bar{q}_2$ along column 1 yields

$$C_{11}\underbrace{(C_{33} - C_{32})}_{>0} + C_{12}\underbrace{(C_{31} - C_{33})}_{<0} + C_{13}(C_{32} - C_{31}). \tag{5.41}$$

The terms in brackets are the same except for reversed signs in both determinants. We do a case analysis on the sign of $(C_{31} - C_{32})$ (note that the signs of the other terms in brackets are determined by the property that $C_{ii} > C_{ij}$ for all $i \neq j$). If $(C_{31} - C_{32}) \leq 0$ then (5.40) can be shown to be positive by making use of $C_{21} < C_{22}$ and $C_{23} < C_{22}$. With the same argument, $(C_{31} - C_{32}) > 0$ implies positivity of (5.41).

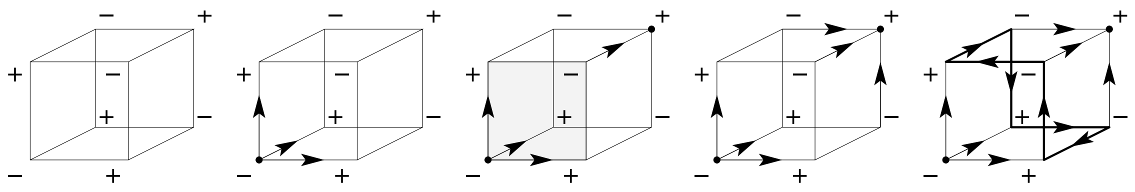


Figure 5.4: The cyclic USO enforced by the pattern.

So, if (5.40) is negative (it's never zero by the nondegeneracy assumption), then $(C_{31} - C_{32}) > 0$ which in turn implies that (5.41) is positive. On the other hand, if (5.41) is negative, then $(C_{31} - C_{32}) \leq 0$ implying that (5.40) is positive. This means that if $\det(C) > 0$, then if one of the edges in dimensions 1 and 2 adjacent to vertex α is outgoing, the other edge has to be incoming. And if $\det(C) < 0$, then if one of the edges in dimensions 1 or 2 is incoming, the other edge has to be outgoing. Since this argumentation works for any two indices of $\bar{q}$ and any set α, we get the following pattern for all α:

$\det(C(\alpha)) > 0$	An outgoing edge forces the other two to be incoming
$\det(C(\alpha)) < 0$	An incoming edge forces the other two to be outgoing

By the P-matrix property of M, it is easy to see that $\det(B(\alpha))$ is positive if and only if $\bar{\alpha}$ has an even number of elements. Determinants of basis matrices differing in one column therefore have opposite signs and since $C(\alpha) = S^T B(\alpha)$ this is also true for $C(\alpha)$. So $sign(\det(C(\alpha))) \neq sign(\det(C(\alpha')))$ whenever $|\alpha \oplus \alpha'| = 1$. We illustrate this by pluses and minuses in Figure 5.4, meaning that the determinant of basis matrix $C(\alpha)$ is positive (negative) if there is a plus (minus) at vertex α.

By the pattern, the source has to sit at a minus-vertex, wlog the one at the bottom left corner in Figure 5.4. Since the source is unique, the other minus-vertices have to have one incoming and two outgoing edges. Look at the minus-vertex antipodal to the source in the front 2-dimensional facet (shaded in Figure 5.4). The edge pointing away from the facet has to be oriented as in the figure, since the other direction would imply two sources (and two sinks) in the facet (by the pattern). The same argument holds for the remaining two minus-vertices, forcing the sink to be antipodal to the source in the 3-cube. It's easy to see that the pattern forces the remaining edges to form a cycle, either the one given in the figure or the one with opposite direction. $\square$

Using our Maple library, it is easy to find examples proving that Theorem 5.39 does not generalize to higher dimensional strongly non hidden K P-matrices (we tried dimensions 4 and 5). However, cyclic q-vectors are in all examples (where we could find a cyclic q-vector by different methods) very close to the vector $(S^T)^{-1}e$, so the problem might be that the properties of our S are not strong enough.

In three dimensions, we haven't found any cyclic q-vectors for matrices in the class (3) of Lemma 5.36. Trying $q := (S^T)^{-1}e$ resulted in a degenerate orientation always (we always found an invertible witness S, although it is not clear that such a witness has to exist). Here is an example of such a matrix:

$$\begin{pmatrix} 49 & -53 & 63 \\ 27 & 74 & -71 \\ -5 & -64 & 85 \end{pmatrix}.$$

We don't know whether Theorem 5.39 can be generalized in any way to the GLCP. One problem is that the witness S is a vertical block matrix in the GLCP case and therefore not square (invertible) in general.

Chapter 6

Simple Stochastic Games

In this chapter, we show that the problem of finding optimal strategies for both players in a *simple stochastic game* (SSG) reduces to the PGLCP. This makes the rich PGLCP theory and algorithms available for SSG. As a special case, we get a reduction from binary SSG to PLCP.

SSG are two-player games on directed graphs, with certain random moves. If both players play optimally, their respective strategies assign values $v(i)$ to the vertices i, with the property that the first player wins with probability $v(i)$, given the game starts at vertex i. For a given start vertex s, the optimization problem associated with the SSG is to compute the *game value* $v(s)$; the decision problem asks whether the game value is at least 1/2. The formal setup of SSG is given in Section 6.1.

Section 6.2 then shows the actual reduction to PGLCP and concludes with an argument why P-matrices are probably more general than the matrices we get in SSG-induced PGLCPs. Section 6.3 gives some negative results, for example that the matrix in the SSG-induced PGLCP is not a hidden K-matrix in general.

6.1 The Setup

We are given a finite directed graph G whose vertex set has the form

$$V = \{\mathbf{1}, \mathbf{0}\} \cup V_{max} \cup V_{min} \cup V_{avg},$$

where $\mathbf{1}$, the *1-sink*, and $\mathbf{0}$, the *0-sink*, are the only two vertices with no outgoing edges.

Vertices in V_{max} belong to the first player which we call the *max player*, while vertices in V_{min} are owned by the second player, the *min player*. Vertices in V_{avg} are *average* vertices. For $i \in V \setminus \{\mathbf{1}, \mathbf{0}\}$, we let $\mathcal{N}(i)$ be the set of neighbors of i along the outgoing edges of i. The elements of $\mathcal{N}(i)$ are $\{\eta_1(i), \ldots, \eta_{|\mathcal{N}(i)|}(i)\}$. An average vertex i is associated with a probability distribution $\mathcal{P}(i)$ that assigns to each outgoing edge (i,j) of i a probability $p_{ij} > 0$, $\sum_{j \in \mathcal{N}(i)} p_{ij} = 1$.

The SSG defined by G is played by moving a token from vertex to vertex, until it reaches either the 1-sink or the 0-sink. If the token is at vertex i, it is moved according to the following rules.

vertex type	rule
$i = \mathbf{1}$	the game is over and the max player wins
$i = \mathbf{0}$	the game is over and the min player wins
$i \in V_{max}$	the max player moves the token to a vertex in $\mathcal{N}(i)$
$i \in V_{min}$	the min player moves the token to a vertex in $\mathcal{N}(i)$
$i \in V_{avg}$	the token moves to a vertex in $\mathcal{N}(i)$ according to $\mathcal{P}(i)$

A SSG is called *stopping*, if no matter what the players do, the token eventually reaches $\mathbf{1}$ or $\mathbf{0}$ with probability 1, starting from *any* vertex. In a stopping game, there are no directed cycles involving only vertices in $V_{max} \cup V_{min}$. The following is well-known and has first been proved by Shapley [83], see also the papers by Condon [12, 13]. Our reduction yields an independent proof of part (i).

Lemma 6.1 *Let G define a stopping* SSG.

(i) *There are unique numbers* $v(i), i \in G$, *satisfying the equations*

$$v(i) = \begin{cases} 1, & i = \mathbf{1} \\ 0, & i = \mathbf{0} \\ \max\limits_{j \in \mathcal{N}(i)} v(j), & i \in V_{max} \\ \min\limits_{j \in \mathcal{N}(i)} v(j), & i \in V_{min} \\ \sum\limits_{j \in \mathcal{N}(i)} p_{ij} v(j), & i \in V_{avg} \end{cases} . \tag{6.2}$$

(ii) *The value* $v(i)$ *is the probability for reaching the 1-sink from vertex* i, *if both players play optimally.*

For a discussion about what it means that 'both players play optimally', we refer to Condon's paper [13]. The important point here is that computing the numbers $v(i)$ solves the optimization version of the SSG in the sense that for every possible start vertex s, we know the *value* $v(s)$ of the game. It also solves the decision version which asks whether $v(s) \geq 1/2$. Additionally, the lemma shows that there are *pure* optimal strategies that can be read off the numbers $v(i)$: if v is a solution to (6.2), then an optimal pure strategy is given by moving from vertex i along one outgoing edge to a vertex j with $v(j) = v(i)$.

The stopping assumption can be made without loss of generality: in a non-stopping game, replace every edge (i, j) by a new average vertex t_{ij} and new edges (i, t_{ij}) (with the same probability as (i, j) if $i \in V_{avg}$), (t_{ij}, j) with probability $1 - \epsilon$ and $(t_{ij}, \mathbf{0})$ with probability ϵ. Optimal strategies to this stopping game (which are given by the $v(i)$ values) correspond to optimal strategies in the original game if ϵ is chosen small enough [12].

Our original reduction is described in [34]. There we also assume that the SSG is stopping, but we don't insist on the replacement of every edge with the average vertices t_{ij} as above. Here we do that, which simplifies the reduction. Since consecutive average vertices can be merged into one average vertex, we can assume bipartiteness of the game graph, in the sense that all neighbors of player vertices are average vertices and all neighbors of average vertices are player vertices or sinks.

6.2 Reduction from SSG to PGLCP

This section describes the core result of this chapter. As a help for the reader, we provide the reduction step by step for a specific example in Subsection 6.3.1.

In the following, G defines a bipartite, transformed (as described above) stopping SSG and every non-sink vertex of G has at least two outgoing edges (a vertex of outdegree 1 can be removed from the game without affecting the values of other vertices). In order to solve (6.2), we first write down an equivalent system of linear equations and inequalities, along with (nonlinear) *complementarity* conditions for certain pairs of variables. The system has one variable x_i for each vertex i and one *slack variable* w_{ij} for each edge (i, j) with $i \in V_{max} \cup V_{min}$. It has equality constraints

$$x_i = \begin{cases} 1, & i = \mathbf{1} \\ 0, & i = \mathbf{0} \\ w_{ij} + x_j, & i \in V_{max},\ j \in \mathcal{N}(i) \\ -w_{ij} + x_j, & i \in V_{min},\ j \in \mathcal{N}(i) \\ \sum\limits_{j \in \mathcal{N}(i)} p_{ij} x_j, & i \in V_{avg}, \end{cases} \tag{6.3}$$

inequality constraints

$$w_{ij} \geq 0, \quad i \in V_{max} \cup V_{min},\ j \in \mathcal{N}(i), \tag{6.4}$$

and complementarity constraints

$$\prod_{j \in \mathcal{N}(i)} w_{ij} = 0, \quad i \in V_{max} \cup V_{min} \tag{6.5}$$

to model the max- and min-behavior in (6.2).

The statement of Lemma 6.1 (i) is equivalent to the statement that the system consisting of (6.3), (6.4) and (6.5) has a unique solution $x = (x_1, \ldots, x_n)$, and we will prove the latter statement. Actually, we prove that the variables x are redundant and that there exists a unique solution for the variables w. From this we can recover the unique solution for the x_i, which are the game values $v(i)$. Note that edges with $w_{ij} = 0$ in the solution correspond to *strategy edges* of the players, i.e., if $w_{ij} = 0$ then $x_i = x_j$ and the best strategy for the player at vertex i is to move the token to vertex j.

In order to obtain a proper GLCP formulation, we remove the redundant x-variables (the fact that we can remove them implies that they are redundant). For variables $x_i, i \notin V_{avg}$, this is easy. For each player vertex i we choose one arbitrary neighbor along an outgoing edge and call it $\eta_1(i)$. In order to reveal connections to the final GLCP already now, we call edges going from i to $\eta_1(i)$ *z-edges* and rename their slack variables to z_i, so $z_i := w_{i\eta_1(i)}$. Because of bipartiteness of the game graph, every edge originating from a player vertex connects to an average vertex j. For $i \in V_{max}$ we can therefore replace all occurrences of x_i in (6.3) by $z_i + x_{\eta_1(i)}$, and for $i \in V_{min}$ by $-z_i + x_{\eta_1(i)}$, and end up with equalities consisting of x_i for $i \in V_{avg}$ only ($x_\mathbf{0}$ and $x_\mathbf{1}$ can be replaced by 0 and 1 immediately):

$$\begin{array}{rcll} w_{ij} - z_i - x_{\eta_1(i)} + x_j &=& 0, & i \in V_{max},\ j \in \mathcal{N}(i) \setminus \eta_1(i) \\ w_{ij} - z_i + x_{\eta_1(i)} - x_j &=& 0, & i \in V_{min},\ j \in \mathcal{N}(i) \setminus \eta_1(i) \\ -x_i + \sum\limits_{j \in \mathcal{N}(i)} p_{ij} S_j &=& 0, & i \in V_{avg}, \end{array} \qquad (6.6)$$

where S_j in the last equation is $z_j + x_{\eta_1(j)}$ if $j \in V_{max}$, $-z_j + x_{\eta_1(j)}$ if $j \in V_{min}$ and 0 or 1 if j is a sink. Note that the right-hand sides of the first two equations are zero since player vertices do not connect to sinks (after the stopping transformation). Nevertheless, allowing connections from player vertices to sinks is no problem, resulting in constant values on the right-hand side.

6.2.1 A Non-Standard GLCP

Let us assume that $V_{max} \cup V_{min} = \{1, \dots, u\}$, $V_{avg} = \{u+1, \dots, n\}$. We define vectors

$$z = (w_{1\eta_1(1)}, \dots, w_{u\eta_1(u)})^T = (z_1, \dots, z_u)^T$$

and

$$w = (w^1, \dots, w^u)^T, \quad x = (x_{u+1}, \dots, x_n)^T,$$

where $w^i = (w_{i\eta_2(i)}, \dots, w_{i\eta_{|\mathcal{N}(i)|}(i)})$ is the vector consisting of the w_{ij} for all $j \in \mathcal{N}(i) \setminus \eta_1(i)$. Conditions (6.4), (6.5) and (6.6) – and therefore the problem of computing the $v(i)$ – can now be written as

$$\begin{array}{ll}\text{find } w, z & \\ \text{subject to} & w, z \geq 0 \\ & z_i \displaystyle\prod_{j=1}^{|\mathcal{N}(i)|-1} w^i_j = 0, \quad i \in V_{max} \cup V_{min} \\ & \begin{pmatrix} w \\ 0 \end{pmatrix} - \begin{pmatrix} I & C \\ A & B \end{pmatrix} \begin{pmatrix} z \\ x \end{pmatrix} = \begin{pmatrix} s \\ t \end{pmatrix}, \end{array} \tag{6.7}$$

where

$$P = \begin{pmatrix} I & C \\ A & B \end{pmatrix}, \quad q = \begin{pmatrix} s \\ t \end{pmatrix}$$

are a suitable matrix and a suitable vector. The vertical block matrix I is partitioned according to w and all representative submatrices of it are the identity matrix in $\mathbb{R}^{u \times u}$. The square matrix B of dimension $n - u$ encodes the connections between average vertices along z-edges. A and C describe how player and average vertices interconnect.

6.2.2 The Structure of the Matrix P

After the transformation to stopping SSG and the merging of the average vertices, all average vertices i have an *escape-edge* to the $\mathbf{0}$-sink, $p_{i\mathbf{0}} > 0$. The matrix B is therefore *row diagonally dominant* with positive diagonal entries, meaning that $B_{ii} > \sum_{j \neq i} |B_{ij}|$. This implies that B is a P-matrix [93]. Since it is a Z-matrix by construction, it is even a K-matrix, see Definition 5.4.

Property 6.8 *An $(n \times n)$ representative submatrix*

$$\bar{P} = \begin{pmatrix} \bar{I} & \bar{C} \\ A & B \end{pmatrix}$$

of P is given by a representative submatrix $\bar{I}$ of I and $\bar{C}$ which consists of the rows of C corresponding to the rows of $\bar{I}$. $\bar{P}$ corresponds to a subgame where (non-z-)edges have been deleted such that every player vertex has exactly two outgoing edges.

Such a subgame is a slightly generalized binary SSG, since average vertices can have more than two outgoing edges and arbitrary probability distributions on them.

Lemma 6.9 *Using elementary row operations, we can transform a representative submatrix $\bar{P}$ of P into a matrix $\bar{P}'$ of the form*

$$\bar{P}' = \begin{pmatrix} \bar{I} & \bar{C} \\ 0 & B' \end{pmatrix}$$

with B' being a P*-matrix.*

Proof. We process the rows of the lower part (AB) of $\bar{P}$ one by one. In the following, we restrict to the subgame corresponding to $\bar{P}$, where all player vertices have two outgoing edges.

For $k \in \{1, \dots, n\}$, let R_k be the kth row of $\bar{P}$ and assume that we are about to process $R_i, i \in \{u+1, \dots, n\}$. According to (6.6), we have

$$R_i \begin{pmatrix} z \\ x \end{pmatrix} = x_i - \sum_{j \in \mathcal{N}(i) \setminus \{\mathbf{0},\mathbf{1}\}} p_{ij}(\pm z_j + x_{\eta_1(j)}), \tag{6.10}$$

with $+z_j$ if $j \in V_{max}$ and $-z_j$ if $j \in V_{min}$. We will eliminate the contribution of $\pm z_j + x_{\eta_1(j)}$ for all $j \in \mathcal{N}(i) \setminus \{\mathbf{0}, \mathbf{1}\}$, by adding suitable multiples of rows $R_j, j \in \{1, \dots, u\}$. For such a j, (6.6) implies

$$R_j \begin{pmatrix} z \\ x \end{pmatrix} = \begin{cases} z_j + x_{\eta_1(j)} - x_{\eta_2(j)}, & j \in V_{max} \\ z_j - x_{\eta_1(j)} + x_{\eta_2(j)}, & j \in V_{min} \end{cases},$$

where $\eta_2(j)$ is the second neighbor of j in the subgame corresponding to $\bar{P}$. Thus, adding a fraction (either $-p_{ij}$ or p_{ij}) of such a row to (6.10) for all $j \in \mathcal{N}(i) \setminus \{\mathbf{0}, \mathbf{1}\}$ transforms our current matrix into a new matrix whose ith row has changed and yields

$$R_i' \begin{pmatrix} z \\ x \end{pmatrix} = x_i - \sum_{j \in \mathcal{N}(i) \setminus \{\mathbf{0},\mathbf{1}\}} p_{ij} x_{\eta_2(j)}. \tag{6.11}$$

This transformation is realized through elementary row operations. Because (6.11) does not contain any z-variables anymore, we get the claimed structure after all rows $R_i, i \in \{u+1, \dots, n\}$, have been processed.

B' is again a P-matrix (even a K-matrix) since it is row diagonally dominant and positive on the diagonals. □

Lemma 6.12 *P is a vertical block* P*-matrix.*

Proof. We show that every representative submatrix $\bar{P}$ of P is a P-matrix. By Lemma 6.9, $\det(\bar{P}) = \det(\bar{P}') = \det(\bar{I})\det(B') = \det(B')$, so $\bar{P}$ has positive determinant since B' is a P-matrix. To see that all proper principal minors are positive, we can observe that any principal submatrix of $\bar{P}$ is again the matrix resulting from a SSG. The subgame corresponding to a principal submatrix can be derived from the SSG by deleting vertices and redirecting edges. The deletion of a player vertex is described in Figure 6.1. Squares correspond to average and trian-

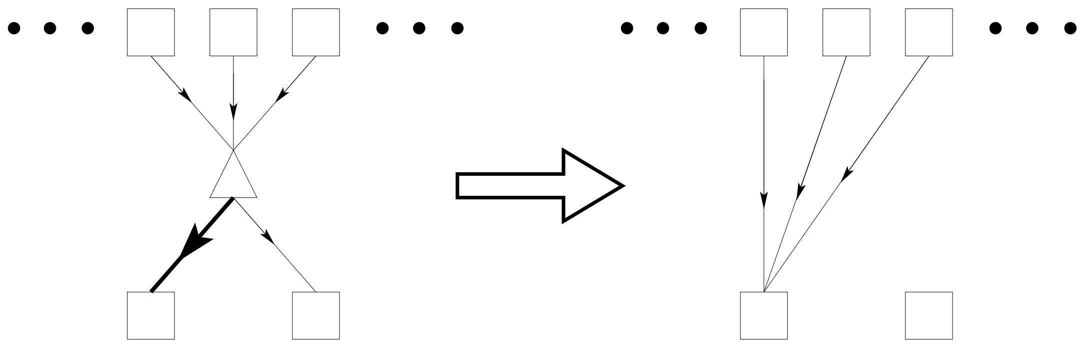


Figure 6.1: Removing a player vertex.

gles to player vertices. The dots indicate that there can be arbitrarily many average vertices having a directed edge to the vertex we want to delete. The bold edge indicates the z-edge of the player vertex (remember that player vertices in the subgame described by $\bar{P}$ have outdegree 2). Strictly speaking, the resulting game is not bipartite anymore, but looking through the reduction once more, we see that consecutive average vertices are handled easily: there are in fact less entries in the submatrix A of $\bar{P}$ we have to get rid of in order to prove that $\bar{P}$ is a P-matrix along the lines of Lemma 6.9.

Figure 6.2 shows that the deletion of an average vertex is achieved by redirecting all incoming edges to the 0-sink. Since we now have sinks after player vertices, the game is not of the original form anymore. But again, the reduction deals with that (some entries in the matrix C of $\bar{P}$ are zero instead of 1 or -1 and the s-vector in (6.7) might no longer be zero, see also the remark after (6.6)) and the matrix corresponding to the subgame has positive determinant. □

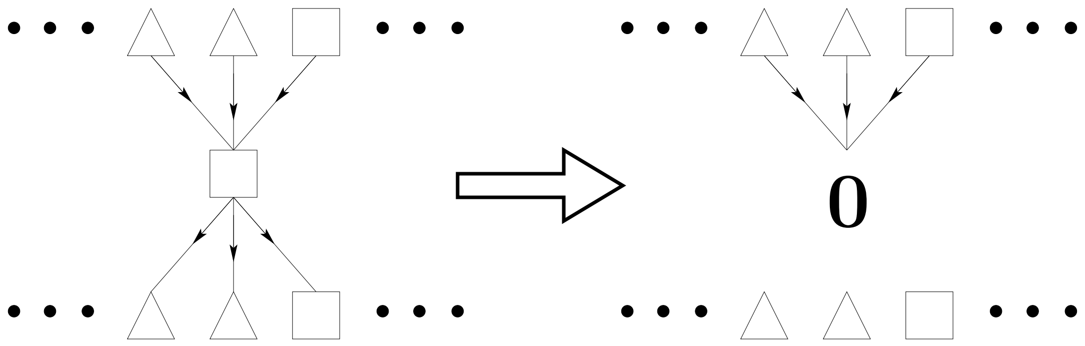


Figure 6.2: Removing an average vertex.

6.2.3 A Standard GLCP

Problem (6.7) is a non-standard GLCP because there are variables x with no complementarity conditions. But knowing that B is regular (since B is a P-matrix), we can express x in terms of z and obtain an equivalent standard GLCP.

$$\begin{array}{ll} \text{find } w, z & \\ \text{subject to} & w, z \geq 0 \\ & z_i \displaystyle\prod_{j=1}^{|\mathcal{N}(i)|-1} w_j^i = 0, \quad i \in V_{max} \cup V_{min} \\ & w - (I - CB^{-1}A)z = s - CB^{-1}t \end{array} \tag{6.13}$$

Lemma 6.14 *The matrix $I - CB^{-1}A$ is a* P*-matrix.*

Proof. We have to show that every representative submatrix of $I - CB^{-1}A$ is a P-matrix. Such submatrices are derived through $\bar{I} - \bar{C}B^{-1}A$. It thus suffices to show that $\bar{I} - \bar{C}B^{-1}A$ is a P-matrix, given that $\bar{P}$ (as defined in Property 6.8) is a P-matrix. This is well-known, $\bar{I} - \bar{C}B^{-1}A$ is called the *Schur complement* of B in $\bar{P}$ (Tsatsomeros [92]). □

We therefore have

Theorem 6.15 *A simple stochastic game is reducible in polynomial time to a generalized linear complementarity problem with a* P*-matrix.*

This theorem also provides a proof of Lemma 6.1: going through our chain of reductions again yields that the equation system in Lemma 6.1 (i) for the values $v(i)$ has a unique solution if and only if the PGLCP (6.13) has a unique solution for the slack variables w and z. The latter holds because the matrix of (6.13) is a P-matrix, see Theorem 2.5.

As mentioned earlier in Section 1.1, the reduction works for a superclass of SSG in which edges are associated with a payoff. But for general stochastic games as introduced by Shapley [83], our reduction is not possible. This follows from two facts. First, optimal strategies for stochastic games are generally non-pure. Second, it is possible to get irrational solutions (vertex values) for the stochastic game even if all input data is rational [74]. This is not possible for GLCP.

Projecting the grid USO we get from a SSG-induced PGLCP to dimensions of one player only (see [33, 32] for details about projected USOs), results in the *acyclic* grid USO that Björklund et al. derive via their reduction [3]. This phenomenon makes us believe that PGLCP is more general than SSG since we don't expect to observe it for general P-matrices. We record this observation in a lemma.

Lemma 6.16 *Consider a* SSG*-induced* PGLCP *instance. The grid* USO *evolving from it has the property that there is at least one subset of dimensions (corresponding to one of the players in the* SSG*) such that the* USO *resulting from projecting w.r.t. the dimensions in the subset as well as the* USO *resulting from projecting w.r.t. all dimensions not in the subset is acyclic.*

An interesting open question is whether the acyclic grid USOs of Björklund et al. satisfy the Holt-Klee condition. The result that PGLCP-induced grid USO fulfill it does not answer the question in the affirmative, since the Holt-Klee property might get lost in a projected USO, see Figure 1.1 on page 9 for an example. There, the left orientation is LP-induced and therefore satisfies the Holt-Klee condition while the right, projected orientation does not (it's the double twist from Figure 3.2 on page 34).

6.3 Negative Results

We show that the matrix in a SSG-induced PGLCP is in general not (transpose) hidden K and not positive semi-definite. Moreover, it is possible that the SSG reduces to a PLCP that in turn reduces to a USO of the cube known as the *Klee-Minty orientation*. In such an orientation, there is a directed path visiting all vertices of the cube. A potential algorithm following this path would therefore need an exponential number of steps to find the sink.

6.3.1 A SSG whose Matrix is not Hidden K

In order to illustrate the reduction, we provide an example here. Moreover, the matrix we get through the reduction is not hidden K, thus destroying one possible hope of getting a polynomial time algorithm for SSG via PGLCP. The example is an adaptation of Condon's original game designed to show that switching algorithms can cycle [13]. The cycle translates to the PGLCP USO in a straightforward way (switching the strategy at a player vertex is to move along one edge in the USO), which proves that the matrix is not hidden K (Theorem 5.20).

The reason we look at an adaptation and not at the original game is that we also get a negative result in the setting of Svensson and Vorobyov [88]. We come back to this after the reduction.

Figure 6.3 shows the SSG, with upper triangles denoting max player and lower triangles denoting min player vertices. Squares are average vertices and bold numbers correspond to sink values. The arrow pointing to the bold number 9/10, for instance, is an abbreviation for an arrow pointing to an average vertex having two outgoing edges, one of them connecting to the 1-sink with probability 9/10 and the other connecting to the 0-sink with probability 1/10. Since this average vertex will have value 9/10, we can disregard it in the reduction (by replacing all occurrences of its x-variable by 9/10). The game consists of 4 player vertices, the resulting vertical block matrix in the PGLCP will thus be of dimension 4 (consist of 4 vertical blocks). We don't know of any game giving a 3-dimensional non hidden K-matrix.

We first write down the equations according to (6.3), nonnegativity

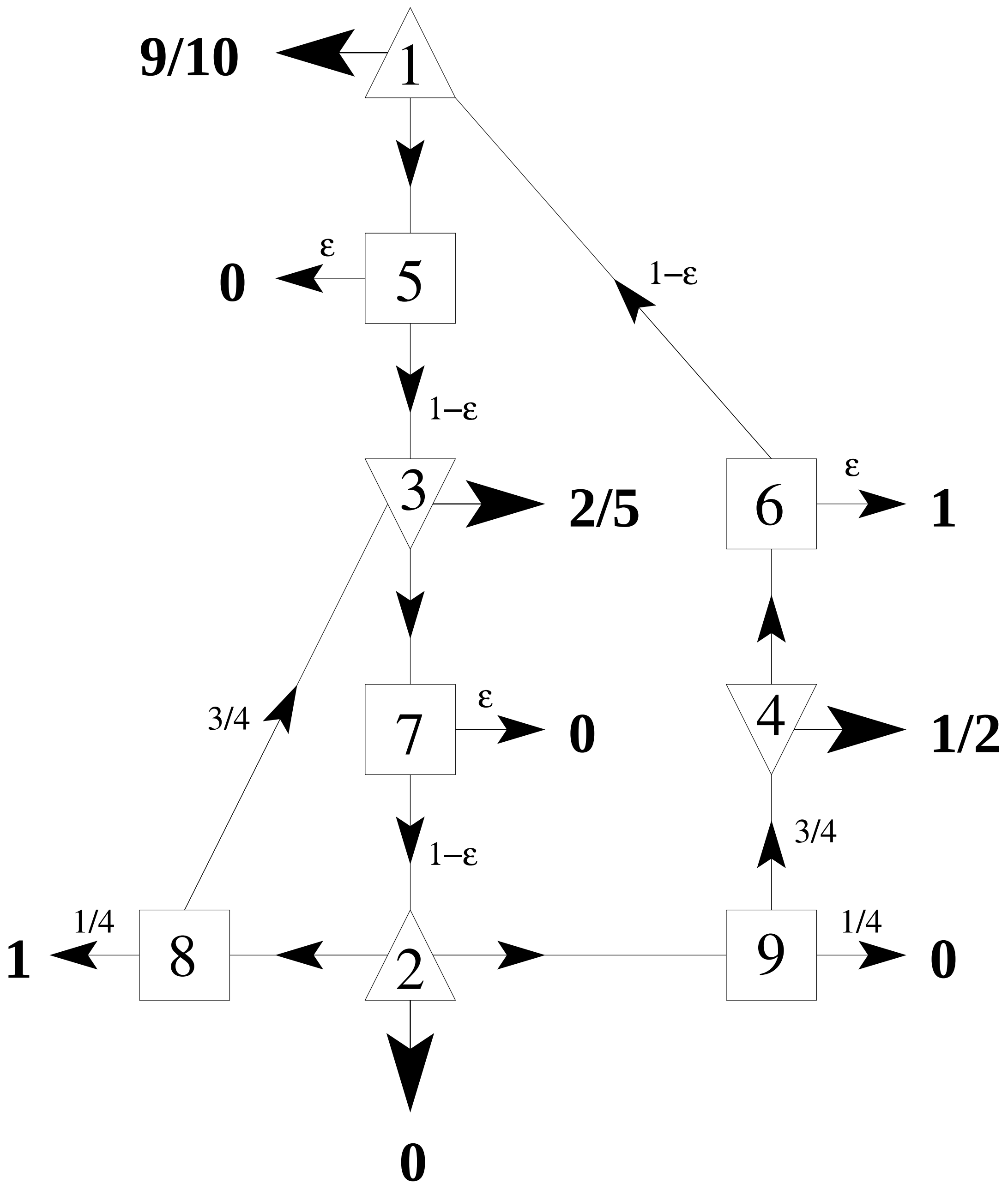


Figure 6.3: A SSG yielding a non hidden K-matrix.

and complementarity constraints omitted. The variables w_{ij} are the slacks for the edges from vertex i to vertex j. The bold arrows in the figure denote the z-edges, so z_1 for instance, is the slack of the edge going from vertex 1 to the value 9/10.

$$\begin{array}{lcl@{\qquad\qquad}lcl}
x_1 & = & z_1 + 9/10 & x_5 & = & (1-\epsilon)(-z_3 + 2/5) \\
x_1 & = & w_{15} + x_5 & x_6 & = & \epsilon + (1-\epsilon)(z_1 + 9/10) \\
x_2 & = & z_2 & x_7 & = & (1-\epsilon)z_2 \\
x_2 & = & w_{28} + x_8 & x_8 & = & 3/4(-z_3 + 2/5) + 1/4 \\
x_2 & = & w_{29} + x_9 & x_9 & = & 3/4(-z_4 + 1/2) \\
x_3 & = & -z_3 + 2/5 & & & \\
x_3 & = & -w_{37} + x_7 & & & \\
x_4 & = & -z_4 + 1/2 & & & \\
x_4 & = & -w_{46} + x_6 & & &
\end{array}$$

Eliminating the x_i for the player vertices (see (6.6)) yields

$$\begin{array}{lcl@{\qquad}lcl}
w_{15} - z_1 + x_5 & = & 9/10 & x_5 & = & (1-\epsilon)(-z_3) + 2/5 - 2/5\epsilon \\
w_{28} - z_2 + x_8 & = & 0 & x_6 & = & (1-\epsilon)z_1 + 9/10 + 1/10\epsilon \\
w_{29} - z_2 + x_9 & = & 0 & x_7 & = & (1-\epsilon)z_2 \\
w_{37} - z_3 - x_7 & = & -2/5 & x_8 & = & 3/4(-z_3) + 11/20 \\
w_{46} - z_4 - x_6 & = & -1/2 & x_9 & = & 3/4(-z_4) + 3/8
\end{array} .$$

Written in matrix form, this is

$$\begin{pmatrix} w_{15} \\ w_{28} \\ w_{29} \\ w_{37} \\ w_{46} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} - P \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{pmatrix} = \begin{pmatrix} 9/10 \\ 0 \\ 0 \\ -2/5 \\ -1/2 \\ -2/5 + 2/5\epsilon \\ -9/10 - 1/10\epsilon \\ 0 \\ -11/20 \\ -3/8 \end{pmatrix}$$

with P being the matrix

$$\left(\begin{array}{cccc|ccccc}
1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
\hline
0 & 0 & (1-\epsilon) & 0 & 1 & 0 & 0 & 0 & 0 \\
-(1-\epsilon) & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & -(1-\epsilon) & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 3/4 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 3/4 & 0 & 0 & 0 & 0 & 1
\end{array}\right) .$$

The partition of P is according to its submatrices I, A, B, C as given in (6.7). Note that columns in A and rows in C corresponding to the max player have nonpositive entries (all the other entries are nonnegative). This fact assures that the matrix in the standard GLCP as in (6.13) is nonnegative:

$$\begin{pmatrix} w_{15} \\ w_{28} \\ w_{29} \\ w_{37} \\ w_{46} \end{pmatrix} - \left(\begin{array}{cccc} 1 & 0 & (1-\epsilon) & 0 \\ \hline 0 & 1 & 3/4 & 0 \\ 0 & 1 & 0 & 3/4 \\ \hline 0 & (1-\epsilon) & 1 & 0 \\ \hline (1-\epsilon) & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = q,$$

where

$$q = \begin{pmatrix} 1/2 + 2/5\epsilon \\ -11/20 \\ -3/8 \\ -2/5 \\ 2/5 + 1/10\epsilon \end{pmatrix}.$$

This vertical block matrix of type $(1, 2, 1, 1)$, as indicated by the horizontal lines, is easily seen to be a P-matrix, since both representative submatrices are row diagonally dominant with ones on the diagonal. The matrix has in fact a very simple structure. There are two reasons for that. The first is the choice of the z-edges in the reduction. All of these edges lead directly to (sink) values. The second reason is that the game is tripartite, meaning that every max vertex is followed by average vertices that are followed by min vertices. These min vertices are followed by average vertices that are followed by max vertices.

Any SSG can without loss of generality be transformed in a tripartite one where every player vertex is connected to a sink. This can be achieved by introducing additional vertices and introducing an edge to the 1-sink for every min vertex and an edge to the 0-sink for every max vertex. Such edges are called *retreat edges* and they are redundant, since it makes no sense for the players to choose such an edge. Applying the same reduction as we do, Svensson and Vorobyov [88] show that on such games, the matrix in the final standard GLCP (the matrix $I - CB^{-1}A$ in (6.13)) is a so called D-*matrix.* The drawback of reducing to a D-matrix is that binary SSG don't reduce to LCP in general (since we introduce the retreat edges). Here is the definition of D-matrices:

Definition 6.17 *A vertical block matrix Q consisting of n blocks is a* D-matrix *if it is nonnegative, strictly row diagonally dominant and if there is a set $\alpha \subseteq [n]$ such that in every representative submatrix $\bar{Q} \in \mathbb{R}^{n\times n}$*

(*i*) *there are ones on the diagonal, and*

(*ii*) $\bar{Q}_{\alpha,\alpha}$ *and* $\bar{Q}_{\bar{\alpha},\bar{\alpha}}$ *are identity matrices of appropriate size.*

Remember that $\bar{Q}_{\alpha,\alpha}$, as defined on page 88, is $\bar{Q}$ restricted to rows and columns with indices in α. After having reduced SSG to D-matrix GLCP, the set α corresponds to the set of max player vertices. The reader is encouraged to check that our matrix above is indeed a D-matrix for $0 < \epsilon \leq 1$.

Every D-matrix is a P-matrix. It is interesting that every square D-matrix Q is hidden K. A possible witness is $X := 2I - Q$, that is Q with all off-diagonal entries turned into their negative values. Since Q is row diagonally dominant, we have $Xe > 0$ and according to Characterization 5.9 it thus suffices to check that QX is a Z-matrix. This is easy.

Despite these nice properties, D-matrices are not hidden K in general. The matrix above is a counterexample for instance for $\epsilon = 1/100$. This can be checked by formulating Characterization 5.28 as an LP that can be solved for example by Maple. More elegantly, computing the $(2\times3\times2\times2)$-grid USO corresponding to the PGLCP derived above, we see that it is cyclic. The cycle corresponds exactly to Condon's original cycling-sequence, i.e., it appears in a 4-dimensional subcube of the grid. The size 3 of the second dimension in our example was only needed to reduce to a D-matrix. The changes we made to Condon's example were to add a retreat edge for vertex 2 (this gives size 3 of the $2nd$ dimension) and to insert the average vertices $5, 6, 7$ to make the game tripartite.

The reader might wonder why the ϵ-edge of vertex 6 goes to the 1-sink, whereas the ϵ-edges of vertices 5 and 7 go to the 0-sink. Although, in order to solve the game, adding average vertices with an ϵ-edge to any sink value works if ϵ is small enough, the reason of the values here is to avoid a degenerate USO. Indeed, if we would direct the ϵ-edge of vertex 6 to the 0-sink as well, the following two strategies would have the same values for all vertices (which is not a bad thing for SSG): choose the non-z-edges for player vertices $1, 3, 4$ and for vertex 2 the edge going to 9 (first strategy) or the retreat edge (second strategy). This results in an undirected edge in the USO.

Our result that every submatrix being hidden K is not enough for a vertical block matrix to be hidden K is a strengthening of the same result for vertical block Z-matrices by Mohan and Neogy [60].

Here is the solution to the above PGLCP with $\epsilon = 1/100$:

$$\begin{pmatrix} w_{15} \\ w_{28} \\ w_{29} \\ w_{37} \\ w_{46} \end{pmatrix} = \begin{pmatrix} 63/125 \\ 0 \\ 7/40 \\ 289/2000 \\ 401/1000 \end{pmatrix}, \qquad \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 11/20 \\ 0 \\ 0 \end{pmatrix}.$$

From this we can read off that the optimal strategy is to choose the z-edge for all player vertices except vertex 2. There, the edge going to vertex 8 should be taken. Given this strategy, the values $v(i)$ in (6.2), or equivalently the values x_i in (6.3), can be computed and checked to fulfill the max and min constraints in (6.2).

The outgoing edges of a player vertex correspond to one block of variables in the PGLCP. The choice of the z-edges determines the z-variable in one block. If we choose different z-edges, then the new PGLCP we get can be derived by doing a principal pivot transform on the old one (moving the z-variables at the right place). Since hidden K-matrices are closed under principal pivot transforms (Theorem 5.11), the matrix resulting from the SSG in Figure 6.3 is not hidden K independent of the choice of z-edges.

If we remove the edge going from vertex 2 to the 0-sink in Figure 6.3 and choose the z-edge at vertex 2 to be the one going to vertex 8, then the reduction yields the square matrix

$$M = \begin{pmatrix} 1 & 0 & (1-\epsilon) & 0 \\ 0 & 1 & -3/4 & 3/4 \\ 0 & (1-\epsilon) & (1/4 + 3/4\epsilon) & 0 \\ (1-\epsilon) & 0 & 0 & 1 \end{pmatrix}$$

and right-hand side vector

$$q = \begin{pmatrix} 1/2 + 2/5\epsilon \\ 7/40 \\ 3/20 - 11/20\epsilon \\ 2/5 + 1/10\epsilon \end{pmatrix}.$$

Setting ϵ to 1/100, the USO arising from this LCP is cyclic, implying that the matrix M is not hidden K. Moreover, the transpose of M is not hidden K either, so the polynomial time algorithms known for the

LCP(M, q) with M^T being hidden K (see [17] and also the discussion on page 88) can not be used for solving games in general. The transpose of the above matrix M (with $\epsilon = 1/100$) is not hidden K, because the following matrix is a witness S as in Characterization 5.29 on page 98 (witness R can be computed as $-MS$):

$$\begin{pmatrix} 1 & 1 & 1 & \frac{-60497}{29403} \\ 1 & 1 & \frac{-503}{396} & 1 \\ -3 & 1 & 1 & 1 \\ 1 & \frac{-602}{297} & 1 & 1 \end{pmatrix}.$$

We finish the example by stating our negative findings as a lemma:

Lemma 6.18 *Simple stochastic games do in general not reduce to* PGLCP *with a hidden* K*-matrix or with the transpose of a hidden* K*-matrix.*

6.3.2 A SSG whose Matrix is Ill-Conditioned

In his book [97], Ye presents an interior point method to solve PLCP in time polynomial in the input size, given that the *condition number* of the P-matrix M is bounded by a polynomial in the dimension of M. The condition number of a P-matrix M is defined as

$$\frac{-\lambda}{\theta(M)},$$

where λ is the smallest eigenvalue of $(M + M^T)/2$ and

$$\theta(M) = \min_{||x||=1} \max_j x_j(M^T x)_j.$$

Note that $\theta(M)$ is always positive, since M is a P-matrix and does not reverse the sign of any nonzero vector [17]. We look at the game given in Figure 6.4. Doing the reduction yields matrix

$$M = \begin{pmatrix} 1 & \frac{1}{\epsilon} - 1 \\ 0 & 1 \end{pmatrix}.$$

The smallest eigenvalue of $(M + M^T)/2$ is $\lambda = \frac{3}{2} - \frac{1}{2\epsilon}$ and $\theta(M)$ can be seen to be at most 1 by choosing $x = (0, 1)^T$. The condition number is then at least $\frac{1}{2\epsilon} - \frac{3}{2}$ which is exponential in the dimension of M if $1/\epsilon$ is. Since ϵ can be chosen arbitrarily small, we get the following result.

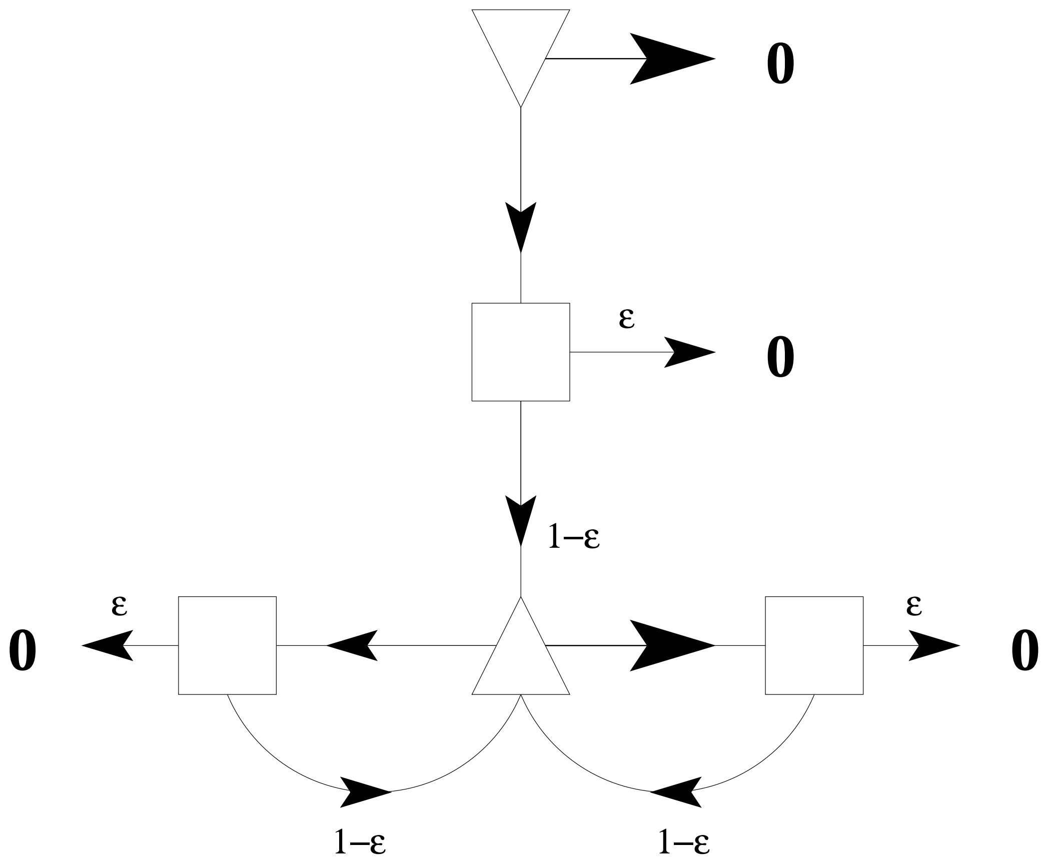


Figure 6.4: A SSG yielding an ill-conditioned matrix.

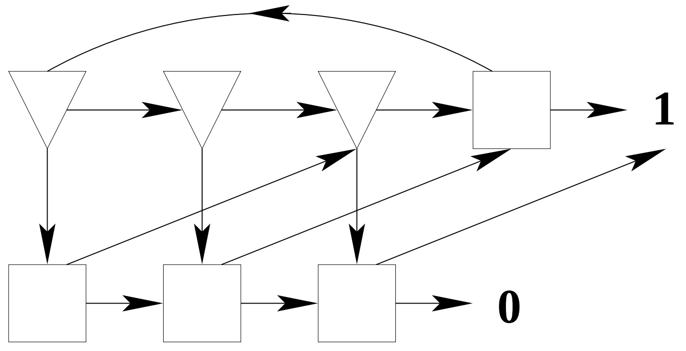


Figure 6.5: A SSG yielding a *Klee-Minty cube.*

Lemma 6.19 *Simple stochastic games do not reduce to* PLCP *with well-conditioned matrix in general.*

By Ye's definition, every P-matrix with positive eigenvalues is well-conditioned. A matrix $M \in \mathbb{R}^{n \times n}$ is *positive semi-definite* if $x^T M x \geq 0$ for all $x \in \mathbb{R}^n$. It's an easy observation that M is positive semi-definite if and only if $(M + M^T)/2$ is as well. Moreover, it is known that a symmetric matrix is positive semi-definite if and only if all eigenvalues are nonnegative (see [17] for example). Positive semi-definite P-matrices are therefore well-conditioned and Ye's algorithm solves such LCPs in polynomial time (polynomial time algorithms are also known for the LCP with general positive semi-definite matrices [67]). The above lemma therefore includes the result that simple stochastic games do not reduce to PLCP with positive semi-definite matrix in general.

6.3.3 Exponential Lower Bound for SSG

Melekopoglou and Condon give examples of binary SSG needing an exponential number of switches in the worst case [59]. Basically, such a game is a generalization of the example with 3 player vertices given in Figure 6.5, where edges at average vertices have probability 1/2.

Since switching an edge at a player vertex amounts to following an outgoing edge in the corresponding PLCP USO, their result states that the cube USO resulting from the game belongs to a class of orientations described by Klee and Minty in [48]. A necessary condition for members in their class of orientations is the existence of a path from the source to the sink visiting all vertices of the cube. And indeed, certain switching algorithms follow this path.

A Klee-Minty cube USO is actually generated by LP and was the first example to show that the simplex method with Dantzig's pivot rule may require an exponential number of pivot steps [48]. For the example of Figure 6.5, a longest possible switching sequence is given in Figure 6.7. Numbers at vertices denote the vertex values for the strategy determined by thick arrows.

Interestingly, the game uses only min player vertices. The path of the 3-cube USO, corresponding to the switching sequence, is given in Figure 6.6. Numbers at cube-vertices denote the strategy of Figure 6.7 corresponding to the vertex.

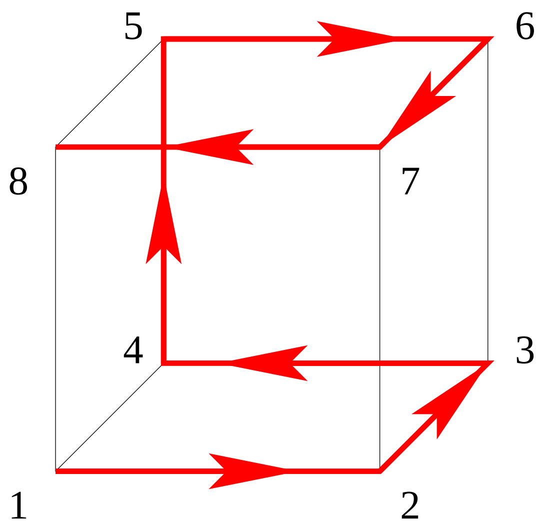


Figure 6.6: The resulting Klee-Minty path.

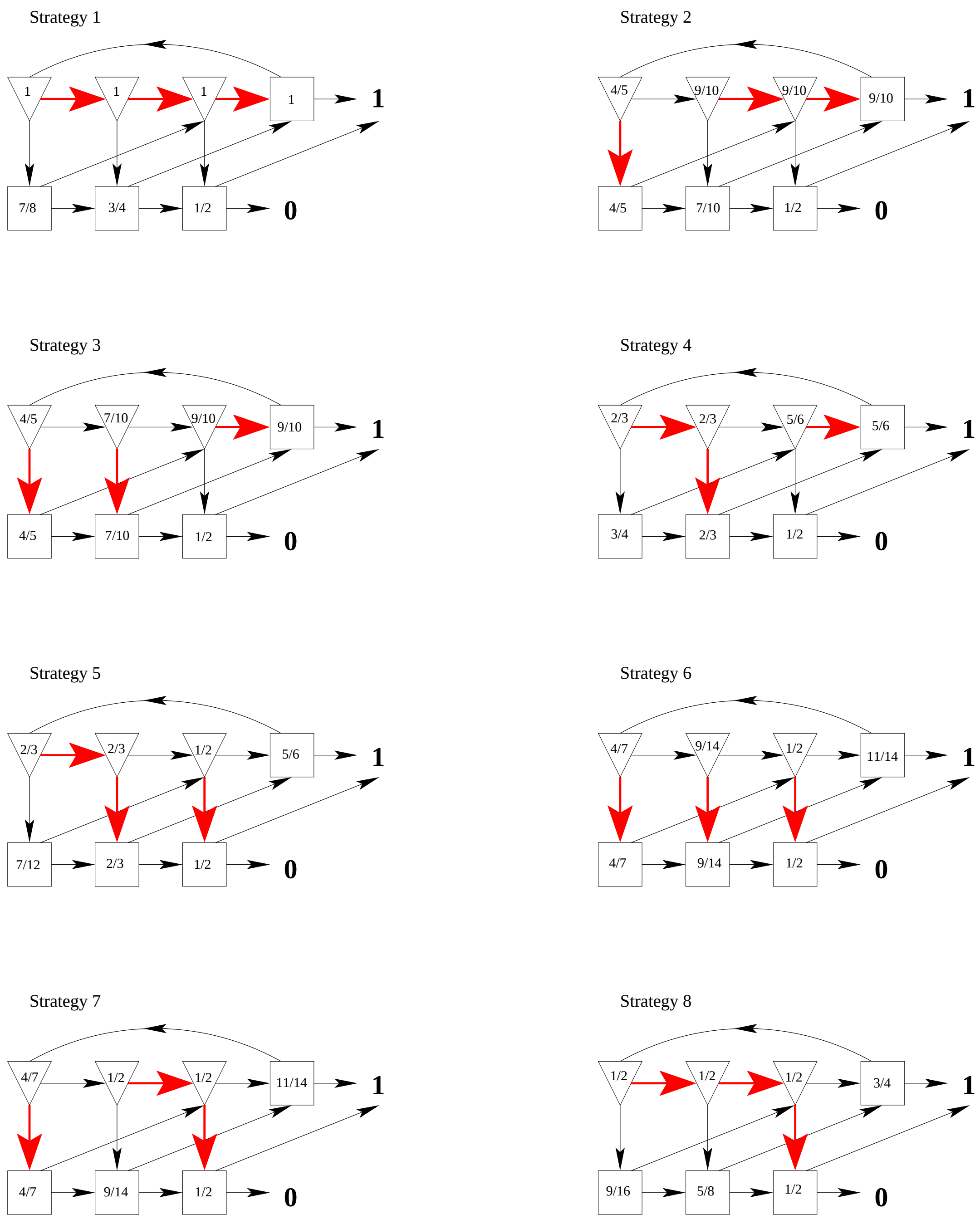


Figure 6.7: A longest possible switching sequence.

Bibliography

[1] E. W. Barankin and R. Dorfman, A method for quadratic programming, *Econometrica* **24** (1956), 340.

[2] M. Bernstein and N. J. A. Sloane, Some canonical sequences of integers, *Linear Algebra and its Applications* **226/228**, 1-3 (1995), 57–72.

[3] H. Björklund, S. Sandberg, and S. Vorobyov, Randomized subexponential algorithms for infinite games, Technical Report 2004-09, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, 2004.

[4] H. Björklund, O. Svensson, and S. Vorobyov, Controlled linear programming for infinite games, Technical Report 2005-13, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, 2005.

[5] H. Björklund, O. Svensson, and S. Vorobyov, Linear complementarity algorithms for mean payoff games, Technical Report 2005-05, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, 2005.

[6] A. Björner, M. Las Vergnas, B. Sturmfels, N. White, and G. M. Ziegler, *Oriented Matroids*, Cambridge University Press, Cambridge, 1999.

[7] A. Borici and H.-J. Lüthi, Fast solutions of complementarity formulations in American put pricing, *Journal of Computational Finance* **9**, 1 (2005), 63–81.

[8] R. Chandrasekaran, A special case of the complementary pivot problem, *Opsearch* **7** (1970), 263–268.

[9] B. Chazelle and J. Matoušek, On linear-time deterministic algorithms for optimization problems in fixed dimension, *Journal of Algorithms* **21** (1996), 579–597.

[10] V. Chvátal, *Linear Programming*, W. H. Freeman, New York, NY, 1983.

[11] K. L. Clarkson, Las Vegas algorithms for linear and integer programming, *Journal of the ACM* **42** (1995), 488–499.

[12] A. Condon, The complexity of stochastic games, *Information & Computation* **96**, 2 (1992), 203–224.

[13] A. Condon, On algorithms for simple stochastic games, in: *Advances in Computational Complexity Theory* (J. Cai, ed.), volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, 1993, 51–73.

[14] R. W. Cottle, Monotone solutions of the parametric linear complementarity problem, *Mathematical Programming* **3** (1972), 210–224.

[15] R. W. Cottle and G. B. Dantzig, Complementary pivot theory of mathematical programming, in: *Mathematics of the Decision Sciences* (G. B. Dantzig and A. F. Veinott, eds.), American Mathematical Society, 1968, 115–136.

[16] R. W. Cottle and G. B. Dantzig, A generalization of the linear complementarity problem, *Journal on Combinatorial Theory* **8** (1970), 79–90.

[17] R. W. Cottle, J. Pang, and R. E. Stone, *The Linear Complementarity Problem*, Academic Press, 1992.

[18] R. W. Cottle and J. S. Pang, A least-element theory of solving linear complementarity problems as linear programs, *Mathematics of Operations Research* **3** (1978), 155–170.

[19] R. W. Cottle and J. S. Pang, On solving linear complementarity problems as linear programs, *Mathematical Programming Study* **7** (1978), 88–107.

[20] C. W. Cryer and Y. Lin, An alternating direction implicit algorithm for the solution of linear complementarity problems arising from free boundary problems, *Applied Mathematics and Optimization* **13** (1985), 1–17.

[21] C. Derman, *Finite state Markovian decision processes*, Academic Press, New York, 1972.

[22] M. Develin, LP-orientations of cubes and crosspolytopes, *Advances in Geometry* **4** (2004), 459–468.

[23] P. du Val, The unloading problem for plane curves, *American Journal of Mathematics* **62** (1940), 307–311.

[24] A. A. Ebiefung and M. M. Kostreva, The generalized linear complementarity problem: least element theory and Z-matrices, *Journal of Global Optimization* **11**, 2 (1997), 151–161.

[25] S. Felsner, B. Gärtner, and F. Tschirschnitz, Grid orientations, $(d, d+2)$-polytopes, and arrangements of pseudolines, *Discrete Computational Geometry* **34**, 3 (2005), 411–437.

[26] M. Fiedler and V. Pták, On matrices with non-positive off-diagonal elements and positive principal minors, *Czechoslovak Mathematical Journal* **12** (1962), 382–400.

[27] K. Fukuda and M. Namiki, On extremal behaviors of Murty's least index method, *Mathematical Programming* **64** (1994), 365–370.

[28] B. Gärtner, The Random-Facet simplex algorithm on combinatorial cubes, *Random Structures & Algorithms* **20**, 3.

[29] B. Gärtner, Randomized algorithms: an introduction through unique sink orientations, Lecture notes, 2003.

[30] B. Gärtner, J. Matoušek, L. Rüst, and P. Škovroň, Violator spaces: structure and algorithms, *Discrete Applied Mathematics* To appear.

[31] B. Gärtner, J. Matoušek, L. Rüst, and P. Škovroň, Violator spaces: structure and algorithms, in: *Proceedings of the 14th Annual European Symposium on Algorithms (ESA)*, volume 4168 of *Lecture Notes in Computer Science*, Springer-Verlag, 2006, 387–398.

[32] B. Gärtner, W. D. Morris, Jr., and L. Rüst, Unique sink orientations of grids, *Algorithmica* To appear.

[33] B. Gärtner, W. D. Morris, Jr., and L. Rüst, Unique sink orientations of grids, in: *Proceedings of the 11th Conference on Integer Programming and Combinatorial Optimization (IPCO)*, volume 3509 of *Lecture Notes in Computer Science*, 2005, 210–224.

[34] B. Gärtner and L. Rüst, Simple stochastic games and P-matrix generalized linear complementarity problems, in: *Proceedings of the 15th International Symposium on Fundamentals of Computation Theory (FCT)*, volume 3623 of *Lecture Notes in Computer Science*, Springer-Verlag, 2005, 209–220.

[35] B. Gärtner and E. Welzl, Linear programming - randomization and abstract frameworks, in: *Proceedings of the 13th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, Springer-Verlag, London, UK, 1996, 669–687.

[36] B. Gärtner and E. Welzl, A simple sampling lemma - analysis and applications in geometric optimization, *Discrete and Computational Geometry* **25**, 4 (2001), 569–590.

[37] M. S. Gowda and R. Sznajder, The generalized order linear complementarity problem, *SIAM Journal on Matrix Analysis and Applications* **15**, 3 (1994), 779–795.

[38] G. J. Habetler and C. N. Haddad, Projective algorithms for solving complementarity problems, *International Journal of Mathematics and Mathematical Sciences* **29**, 2 (2002), 99–113.

[39] G. J. Habetler and B. P. Szanc, Existence and uniqueness of solutions for the generalized linear complementarity problem, *Journal of Optimization Theory and Applications* **84**, 1 (1995), 103–116.

[40] N. Halman, Simple stochastic games, parity games, mean payoff games and discounted payoff games are all LP-type problems, *Algorithmica* To appear.

[41] N. Halman, An EGLP formulation for the simple stochastic game problem, or a comment on the paper: *A subexponential randomized algorithm for the simple stochastic game problem* by W. Ludwig, Technical Report RP-SOR-01-02, Department of Statistics and Operations Research, 2001.

[42] N. Halman, *Discrete and Lexicographic Helly Theorems and Their Relations to LP-type problems*, Ph.D. thesis, Tel-Aviv University, 2004.

[43] C. Hildreth, Point estimates of ordinates of concave functions, *Journal of the American Statistical Association* **49** (1954), 598–619.

[44] A. Hoffman and R. Karp, On nonterminating stochastic games, *Management Science* **12**, 5 (1966), 359–370.

[45] M. Jurdziński and R. Savani, A simple P-matrix linear complementarity problem for discounted games, Manuscript, 2007.

[46] A. Karttunen, Go to http://www.research.att.com/∼njas/sequences/ and enter the first numbers of the sequence: 1,3,11,47,231,...

[47] L. G. Khachiyan, Polynomial algorithms in linear programming, *U.S.S.R. Computational Mathematics and Mathematical Physics* **20** (1980), 53–72.

[48] V. Klee and G. J. Minty, How good is the simplex method?, in: *Inequalities III*, Academic Press, 1972, 159–175.

[49] M. Kojima, N. Megiddo, T. Noma, and A. Yoshise, A unified approach to interior point algorithms for linear complementarity problems: a summary, *Operations Research Letters* **10** (1991), 247–254.

[50] C. E. Lemke, Bimatrix equilibrium points and mathematical programming, *Management Science* **11** (1965), 681–689.

[51] C. E. Lemke and J. T. Howson, Equilibrium points of bimatrix games, *SIAM Journal on Applied Mathematics* **12** (1964), 413–423.

[52] O. L. Mangasarian, Linear complementarity problems solvable by a single linear program, *Mathematical Programming* **10** (1976), 263–270.

[53] O. L. Mangasarian, Solution of linear complementarity problems by linear programming, in: *Numerical Analysis* (G. W. Watson, ed.), volume 506 of *Lecture Notes in Mathematics*, Springer-Verlag, Berlin, Heidelberg, New York, 1976, 166–175.

[54] O. L. Mangasarian, Generalized linear complementarity problems as linear programs, *Operations Research Verfahren* **31** (1979), 393–402.

[55] O. L. Mangasarian, Simplified characterizations of linear complementarity problems solvable as linear programs, *Mathematics of Operations Research* **4** (1979), 268–273.

[56] J. Matoušek and B. Gärtner, *Understanding and Using Linear Programming*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[57] J. Matoušek, M. Sharir, and E. Welzl, A subexponential bound for linear programming, *Algorithmica* **16** (1996), 498–516.

[58] N. Megiddo, A note on the complexity of P-Matrix LCP and computing an equilibrium, Technical report, IBM Almaden Research Center, San Jose, 1988.

[59] M. Melekopoglou and A. Condon, The complexity of the policy improvement algorithm for Markov decision processes, *ORSA Journal on Computing* **6**, 2.

[60] S. R. Mohan and S. K. Neogy, The role of representative submatrices in vertical linear complementarity theory, *Linear & Multilinear Algebra* **41**, 2 (1996), 175–187.

[61] S. R. Mohan and S. K. Neogy, Vertical block hidden Z-matrices and the generalized linear complementarity problem, *SIAM Journal on Matrix Analysis and Applications* **18**, 1 (1997), 181–190.

[62] W. D. Morris and J. Lawrence, Geometric properties of hidden Minkowski matrices, *SIAM Journal on Matrix Analysis and Applications* **10**, 2 (1989), 229–232.

[63] W. D. Morris, Jr., Distinguishing cube orientations arising from linear programs, Manuscript, 2002.

[64] W. D. Morris, Jr., Randomized principal pivot algorithms for P-matrix linear complementarity problems, *Mathematical Programming, Series A* **92** (2002), 285–296.

[65] W. D. Morris, Jr. and M. Namiki, Good hidden P-matrix sandwiches, *Linear Algebra and its Applications* To appear.

[66] K. G. Murty, Note on Bard-type scheme for solving the complementarity problem, *Opsearch* **11** (1974), 123–130.

[67] K. G. Murty, *Linear Complementarity, Linear and Nonlinear Programming*, Helderman-Verlag, 1988.

[68] J. Nash, Equilibrium points in n-person games, in: *Proceedings of the National Academy of Sciences*, volume 36, 1950, 48–49.

[69] J. Nash, Non-cooperative games, *Annals of Mathematics* **54** (1951), 286–295.

[70] J. Pang, Hidden Z-matrices with positive principal minors, *Linear Algebra and its Applications* **23** (1979), 201–215.

[71] J. Pang, On discovering hidden Z-matrices, in: *Constructive Approaches to Mathematical Models* (C. V. Coffman and G. J. Fix, eds.), Proceedings of a conference in honor of R. J. Duffin, Academic Press, New York, 1979, 231–241.

[72] J. S. Pang, On cone orderings and the linear complementarity problem, *Linear Algebra and its Applications* **22** (1978), 267–281.

[73] J. S. Pang and R. Chandrasekaran, Linear complementarity problems solvable by a polynomially bounded pivoting algorithm, *Mathematical Programming Study* **25** (1985), 13–27.

[74] T. Parthasarathy and T. E. S. Raghavan, An orderfield property for stochastic games when one player controls transition probabilities, *Journal of Optimization Theory and Applications* **33**, 3 (1981), 375–392.

[75] A. Puri, *Theory of Hybrid Systems and Discrete Event Systems*, Ph.D. thesis, University of California at Berkeley, 1995.

[76] Jörg Rambau, TOPCOM: triangulations of point configurations and oriented matroids, in: *Mathematical Software — ICMS 2002* (A. M. Cohen, X-S. Gao, and N. Takayama, eds.), World Scientific, 2002, 330–340.

[77] J. Rohn, Systems of linear interval equations, *Linear Algebra and its Applications* **126** (1989), 39–78.

[78] L. Rüst, *Unique Sink Orientations of Grids*, Master's thesis, ETH Zurich, Institute of Theoretical Computer Science, 2004.

[79] H. Samelson, R. M. Thrall, and O. Wesler, A partition theorem for Euclidean n-space, in: *Proceedings of the American Mathematical Society 9*, 1958, 805–807.

[80] U. Schäfer, A linear complementarity problem with a P-matrix, *SIAM Rev.* **46**, 2 (2004), 189–201.

[81] I. Schurr, *Unique Sink Orientations of Cubes*, Ph.D. thesis, ETH Zurich, 2004.

[82] I. Schurr and T. Szabó, Jumping doesn't help in abstract cubes, in: *Proceedings of the 11th Conference on Integer Programming and Combinatorial Optimization (IPCO)*, volume 3509 of *Lecture Notes in Computer Science*, Springer-Verlag, 2005, 225–235.

[83] L. S. Shapley, Stochastic games, *Proceedings of the National Academy of Sciences, U.S.A.* **39** (1953), 1095–1100.

[84] M. Sharir and E. Welzl, A combinatorial bound for linear programming and related problems, in: *Proceedings of the 9th Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 577 of *Lecture Notes in Computer Science*, Springer-Verlag, 1992, 569–579.

[85] P. Škovroň, *Generalized Linear Programming*, Master's thesis, Charles University, Prague, 2002.

[86] P. Škovroň, *Abstract Models of Optimization Problems*, Ph.D. thesis, Charles University, Prague, 2007.

[87] A. Stickney and L. Watson, Digraph models of Bard-type algorithms for the linear complementarity problem, *Mathematics of Operations Research* **3** (1978), 322–333.

[88] O. Svensson and S. Vorobyov, Linear complementarity and P-matrices for stochastic games, in: *Proceedings of the 6th International Andrei Ershov Memorial Conference "Perspectives of System Informatics"*, to appear.

[89] T. Szabó and E. Welzl, Unique sink orientations of cubes, in: *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, 2000, 547–555.

[90] B. P. Szanc, *The Generalized Complementarity Problem*, Ph.D. thesis, Rensselaer Polytechnic Institute, Troy, NY, 1989.

[91] S. Tessaro, *Randomized algorithms to locate the sink in low dimensional unique sink orientations of cubes*, Semester thesis, Department of Computer Science, ETH Zurich, 2004.

[92] M. J. Tsatsomeros, Principal pivot transforms: properties and applications, *Linear Algebra and its Applications* **307** (2000), 151–165.

[93] M. J. Tsatsomeros, Generating and detecting matrices with positive principal minors, in: *Focus on Computational Neurobiology*, Nova Science Publishers, Inc., 2004, 115–132.

[94] A. W. Tucker, A combinatorial equivalence of matrices, in: *Combinatorial Analysis* (R. Bellman and M. Hall, eds.), American Mathematical Society, 1960, 129–140.

[95] B. von Stengel, Computing equilibria for two-person games, in: *Handbook of Game Theory*, volume 3, Elsevier Science Publishers (North-Holland), 2002, 1723–1759.

[96] A. C-C. Yao, Probabilistic computations: toward a unified measure of complexity, in: *Proceedings of the 18th Symp. on Foundations of Computer Science*, IEEE, 1977, 222–227.

[97] Y. Ye, *Interior Point Algorithms*, John Wiley and Sons, 1997.

[98] U. Zwick and M. Paterson, The complexity of mean payoff games on graphs, *Theoretical Computer Science* **158**, 1-2 (1996), 343–359.

Curriculum Vitae

Leo Rüst
born March 23, 1980
citizen of Thal (SG), Switzerland

1992-1999
High School
Kantonsschule Zug, Switzerland
Maturität Typus B (Latein)

1999-2004
Technical University
ETH Zurich, Switzerland
Master of Computer Science ETH

2004-2007
Doctorate
ETH Zurich, Switzerland
Doctor of Sciences ETH