ANGEWANDTE MATHEMATIK UND INFORMATIK

UNIVERSITÄT ZU KÖLN

Report No. 97.312

Geometric Ideas for Graph Representation and for Cooperative Game Theory

by

Sándor P. Fekete

1997

Sándor P. Fekete
Center for Parallel Computing
Universität zu Köln
D–50923 Köln
GERMANY

1991 Mathematics Subject Classification: 05C10, 05C75, 68R05, 68R10, 90C27, 90D12

Keywords: Graph representation, visibility, graph drawing, 3-dimensional geometry, complexity, NP-hardness, logic engine, cooperative game, core, cost allocation, tax, traveling salesman, traveling preacher, Held-Karp bound, spanning tree, X3C, nucleolus, nucleon, matching

Contents

Zusammenfassung

Wir berichten in der vorliegenden Arbeit über Untersuchungen, die durch geometrische Ideen motiviert sind.

Im ersten Teil beschäftigen wir uns mit Fragestellungen der Darstellung von Graphen. Diese Thematik hat eine Tradition, die bis in die historischen Anfänge der Graphentheorie zurückreicht: Schon bei Euler [42, 43, 44] gibt es enge Verbindungen zwischen Geometrie und Graphentheorie. Kuratowskis berühmter Satz über die Charakterisierung planarer Graphen [81] ist nicht zuletzt eine Aussage über das Zeichnen planarer Graphen, ein Thema, das auch von Tutte [108] aufgegriffen wurde.

Mit zunehmendem Einsatz von Computern für Aufgaben in der Visualisierung von Daten ist das Interesse an algorithmischen Verfahren zur Darstellung von Graphen („Graph Drawing“) immer größer geworden. Angesichts der Vielfältigkeit darzustellender Informationen ist es nicht nur wichtig, Algorithmen zu untersuchen, sondern auch von Bedeutung, neue Konzepte für die Darstellung von Graphen zu entwickeln. Ein solches Konzept ist die sogenannte „Sichtbarkeitsdarstellung“von Graphen. Dabei werden Knoten durch Objekte im Raum repräsentiert, die miteinander durch eine Kante verbunden sind, wenn sie sich auf wohldefinierte Weise „sehen“ können. Je nach Art der Objekte und Definition der Sichtbarkeit können recht unterschiedliche Graphen dargestellt werden. Aufgrund ihrer einfachen Struktur interessiert man sich besonders für Objekte wie achsenparallele Quadrate, Rechtecke, Kreise, Würfel und Quader. Interessante Versionen der Sichtbarkeit von zwei Objekten sind dadurch gegeben, daß es eine z-achsenparallele oder allgemein achsenparallele Verbindung eines Paares von jeweils inneren Punkten gibt, die keine anderen Objekte berührt.

Es lassen sich eine Reihe von mathematischen Aussagen darüber treffen, wo die Möglichkeiten solcher Darstellungen an ihre Grenzen stoßen. Mit Hilfe von kombinatorischen Methoden konnten wir obere und untere Schranken für die Größe von vollständigen Graphen mit Sichtbarkeitsdarstellungen finden. Außerdem gelang es uns, eine allgemeine Methode zu entwickeln, mit der sich die komplexitäts-

theoretische Schwere einer großen Klasse von Problemen zeigen läßt.

Im zweiten Teil beschäftigen wir uns mit Fragestellungen aus dem Bereich der kooperativen Spieltheorie. In typischen Problemen der kombinatorischen Optimierung geht es darum, Kosten einzusparen; ein bekanntes Beispiel ist das Traveling Salesman Problem (TSP), in dem es um das Auffinden einer möglichst günstigen Rundtour geht. Für viele praktische Probleme ist aber auch die *Verteilung* der entstehenden Kosten auf die beteiligten Parteien von Bedeutung. Welchen Anteil sollen Teilnehmer an einem Transport- oder Kommunikationsnetz bezahlen? Welche Methoden sind bei der Verteilung von Ressourcen anzuwenden? Wie lassen sich Konfliktsituationen so regeln, daß der Nutzen der beteiligten Parteien optimiert wird? Fragen dieser Art werden schon seit geraumer Zeit von der von von Neumann und Morgenstern [109] begründeten Spieltheorie untersucht, insbesondere der *kooperativen Spieltheorie*, die sich mit der konstruktiven Zusammenarbeit einzelner kostenbewußter Parteien beschäftigt: Wenn eine Gruppe von Teilnehmern an einem Netzwerk mehr bezahlen muß, als es sie kosten würde, ein eigenes Netzwerk zu finanzieren, so wird sie dies in einer freien Marktwirtschaft auch tun. Als Menge der für einen Netzbetreiber sinnvollen (oder auch „fairen“) Kostenverteilungen ergibt sich somit das sogenannte *Core* – ein Polytop, dessen Facetten sich aus der exponentiell großen Menge von durch Untergruppen von Teilnehmern induzierten Restriktionen ergeben.

Erst seit relativ kurzer Zeit beschäftigt man sich in der Spieltheorie mit Aspekten algorithmischer Natur, also der Konstruktion systematischer Lösungsmethoden, die derartige Verteilungen in annehmbarer Zeit finden. Da das TSP algorithmisch schwer ist, überrascht es nicht, daß Fragen über sogenannte „TSP-Spiele“ nach dem Stand des derzeitigen Wissens nicht exakt lösbar sind. Zudem kann man zeigen, daß das Core für TSP-Spiele leer sein kann. Dies motiviert die Untersuchung von *approximativen* Kostenverteilungen, bei denen ein (möglichst gering zu haltendes) Defizit auftreten darf, bzw. bei dem der Kostenanteil für einige der Kunden den Preis für eine separate Versorgung in begrenztem Maße übersteigt. Es gelang uns [45], eine elegante geometrisch motivierte Verteilungsmethode zu finden, mit der sich das Defizit auf $\frac{1}{3}$ begrenzen läßt; sollte eine schon länger bekannte Vermutung über die sogenannte „Held-Karp-Schranke“ des Subtour-Polytops korrekt sein, können wir diesen Wert auf maximal $\frac{1}{4}$ begrenzen. Dies ist gleichzeitig der bestmögliche Approximationsfaktor, da wir zeigen konnten, daß es Beispiele gibt, in denen tatsächlich ein Defizit von $\frac{1}{4}$ unvermeidbar ist, wenn man keine der Core-Restriktionen verletzen will.

Den Gedanken der Optimierung eines multiplikativen Faktors zwischen den Kosten, die einer Teilkoalition entstehen, wenn sie sich von der Gesamtheit abspaltet, und den ihr in Rechnung gestellten Kosten kann man konsequent weiterverfolgen. Man gelangt dadurch zu dem von uns in der Arbeit [47] eingeführten Begriff des „Nucleons“ der auch für andere kooperative Spiele von großem Interesse ist. Das Konzept des Nucleons kann als multiplikatives Analogon zum sogenannten „Nucleolus“interpretiert werden, bei dem die Differenz zwischen den für jede Einzelkoalition entstehenden und den ihr tatsächlich berechneten Kosten optimiert wird. Dieses klassische Konzept von Schmeidler [96] hat zwar bislang große Aufmerksamkeit erhalten, reflektiert aber nur unzureichend die in der realen Ökonomie üblichen Gepflogenheiten: Preisnachläße werden eben nicht für jede Kundengruppe in gleicher Weise gewährt, sondern in Abhängigkeit vom Anteil am Umsatz. Es gelang uns, mit einem polynomialen Verfahren zur Berechnung des Nucleons für Matching-Spiele auch die algorithmische Tauglichkeit dieses Konzepts nachzuweisen.

Für viele Zwecke ist es wichtig, über dem Core eines Spieles zu optimieren. Dies ist aber selbst für so einfache Spiele wie das Minimum-Cost-Spanning-Tree-Spiel, für die es immer einfache faire Verteilungen gibt, ein hoffnungsloses Unterfangen: Wir konnten in der Arbeit [46] zeigen, daß es CoNP-schwer ist, zu bestimmen, ob eine vorgegebene Verteilung im Core liegt. Aufgrund der Äquivalenz von Separierung und Optimierung bedeutet dies, daß Optimierung über dem Polyeder der fairen Verteilungen CoNP-schwer ist.

Mit einem neuen Resultat über das sogenannte Traveling-Preacher-Problem [54] gelang es uns schließlich zu zeigen, daß die Ursache für die komplexitätstheoretische Schwierigkeit der Behandlung von kooperativen Spielen in erster Linie in der Existenz eines vorgegebenen Quellknotens zu suchen ist: Das Analogon des TSP games ohne einen solchen Quellknoten läßt sich in polynomialer Zeit lösen, obwohl man es mit exponentiell vielen Restriktionen zu tn hat, von denen jede durch die Lösung eines TSPs beschrieben ist.

Chapter 1

Introduction

1.1 Graph Theory and Geometry

In 1736, Leonhard Euler [42] discovered graph theory, when he realized that the question of traveling a set of bridges in Königsberg can be solved by stripping off the problem's geometry and only considering elementary combinatorial properties. A few years later [43, 44], he drew a connection between geometric objects and graphs by showing that convex three-dimensional polyhedra can be drawn in the plane only if they satisfy $V + F = E + 2$ on the numbers V of vertices, F of faces, E of edges; in fact, this formula must be satisfied by all planar graphs.

Almost two centuries later, Kuratowski [81] gave a very elegant answer to a question aiming in the opposite direction: When can a graph be drawn in the plane? With his precise characterization (a graph can be drawn in the plane without crossing edges, if and only if it does not contain a subdivision of K_5 or $K_{3,3}$), he was the first to consider abstract combinatorial structures as the main subject, whereas geometry was to be a tool for representing these structures. As Tutte [108] showed in 1963, any planar graph can be drawn with straight non-crossing lines, establishing a particularly aesthetic way of drawing. Many others have followed in Kuratowski's foot steps, most notably Robertson and Seymour [93], who showed that a Kuratowski-type theorem holds for embedding graphs on any surface, i. e., there is a finite family of graphs whose subdivisions must not occur as subgraphs.

1.2 Graph Drawing

In recent years, a growing number of scientists in the community of combinatorial optimizers, graph theorists, and computational geometers have dedicated efforts to the problem of drawing graphs. See the volumes [18, 31, 87, 104] for an overview, as well as the upcoming book by Di Battista, Eades, Tamassia, and Tollis [33], the survey articles by the same authors [32], or the article by Brandenburg, Jünger, and Mutzel [19].

The mission is illustrated by the hand-drawn diagram in Figure 1.1, representing a software package. (The package is used for analyzing and forecasting the savings of millions of customers of the German public loan banks – see [7] for some information on this project.) How should this structure be represented geometrically, such that a human can easily extract the important information, comprehend the relevant substructures, and get an intuitive understanding of the relevant connections?

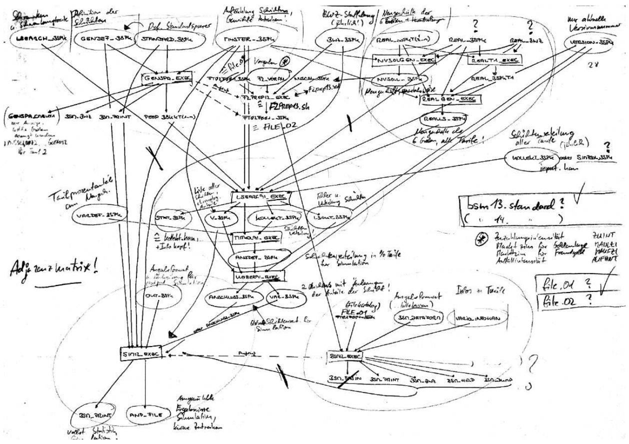


Figure 1.1: Structure of a large software package: hand-drawn version by the developers

Many different types of mathematical questions are motivated by this scenario. One can try to embed a graph by optimizing a layout that is a compromise between “attracting” and “repelling” forces; this optimization is done heuristically with local improvement methods. Since Eades [36] presented this idea, numerous teams of researchers have presented software packages using this approach. Another line of research arises from the background of combinatorial optimization: If it is not possible to draw a graph “perfectly” (e. g., without edge crossings), can one find embeddings with a small number of crossings? Jünger and Mutzel [70, 71, 72, 85] have used a polyhedral approach to deal with this NP-hard problem. Other researchers have examined objective functions like area of the drawing (see [9, 57]), or number of bends per edge (see [10, 37, 56]).

Over recent years, the ever-increasing improvements in speed, memory, and graphic capabilities of modern computers have raised the possibilities of three-dimensional graph drawing. Since many practical instances of graph drawing come from the world of modern telecommunication (e. g., patterns of international phone calls or internet navigation) as well as software design, it is easy to understand that these improving possibilities have to keep pace with increasing demands on the performance. This makes it indispensable to develop new concepts to represent graphs. Clearly, this goal cannot be achieved without understanding some fundamental geometric concepts of representing graphs in three dimensions. In Chapter 2, we describe an approach that follows the idea of combining geometry with graph theoretic structures; in particular, we give an overview over the papers [51, 53, 52]. In Section 2.1, we introduce the concept of *visibility representations*. In Section 2.2, we describe the results of the paper [51], showing that the graph K_{56} does not have a visibility representation by rectangles in three-dimensional space; we also show that K_7 is the largest complete graph that can be represented by unit squares, while every complete graph has a visibility representation by unit discs. Section 2.3 gives an overview of the paper [53] and includes a variety of lower and upper bound of complete graphs that can be represented by fully-dimensional boxes in space. Section 2.4 deals with aspects of complexity. We introduce the technique of the *wobbly logic engine*, which can be used to prove NP-hardness of non-rigid graph representation problems. This method is the subject of the paper [52], where it is used to resolve the complexity of several of these problems.

1.3 Cooperative Games

The field of game theory deals with the economic behavior of individuals, companies, and nations. What are the prices for goods and services in a market economy? What strategies should be followed to choose from a variety options in order to optimize profit? These are questions that were studied in the book by von Neumann and Morgenstern [109]. They suggest that rational entities (the "players") interact in a way that maximizes their individual benefit. Instances of these interactions are called "games"; while *Competitive* Game Theory studies the strategies of conflict, *Cooperative* Game Theory deals with aspects of cooperations between a finite set N of players: How should they divide cost or profit in a fair way?

Formally, let $v : 2^N \to \mathbb{R}$ be a function that describes the cost for any subset of players. This function characterizes the game; typical examples arise from the cost for a connected cable network, or a roundtrip that visits all players in the set. It is reasonable to assume that no subset S is willing to contribute more than the amount of $v(S)$ – the cost of their own network or roundtrip. Since we have a finite number of subsets, the set of cost allocations satisfying all these conditions is a polyhedron – the *core* or set of fair allocations.

Definition 1.3.1 *Let (N, v) be a cooperative game. The* core *of the game is the polyhedron consisting of all vectors $x \in \mathbb{R}^n$ satisfying*

$$\begin{aligned} x(N) := \sum_{i \in N} x_i &= v(N), && (1.1) \\ x(S) := \sum_{i \in S} x_i &\leq v(S) \textit{ for all } S \subset N. && (1.2) \end{aligned}$$

Note that in the case of distributing savings or profit instead of cost, the inequalities have to be reversed. The concept of *stability* was already postulated by von Neumann and Morgenstern [109]. The first to define the core explicitly was Gillies (see [100]).

The geometry of the core is the subject of a number of questions: Is the core of a game non-empty? What is to be done if it is? What allocation should be chosen if there is more than one core element? How can this be achieved algorithmically? These are the types of question that we consider in the following sections.

It should be noted that over the years, game theory has been strong on concepts; these concepts include the Shapley value (Shapley 1953 [97]), bargaining sets (Aumann and Maschler 1964 [6]), the kernel (Davis and Maschler 1965 [27]), and the nucleolus (Schmeidler 1969 [96]). On the other hand, until recently, there has been less emphasis on aspects of computation; this may be in part because the description of a general cooperative game already takes space that is exponential in the number of players – there are exponentially many values of $v(S)$ that need to be listed. However, typical structures where cost needs to be allocated have been considered by combinatorial optimizers in the context of finding a best solution for a particular set of players: Scenarios where the cost for a set of nodes are given by a minimum cost spanning tree, a shortest roundtrip, or an optimal matching are examples of *implicit* descriptions of the value $v(S)$ of a subset. It is evident that combinatorial optimization games can be understood much better with the help of some insights from combinatorial optimization. In recent year, problems of this type have been considered: for traveling salesman games by Tamir [105], Faigle and Kern [48], Kuipers [79], and Potters and Curiel [91]; for routing games (where the order of nodes to be visited by a traveling salesman is induced by the permutation on the full set of nodes) by Curiel, Potters, and Tijs [25], Derks and Kuipers [30], Solymósi, Aarts, and Driessen [102]; for minimum cost spanning tree games by Bird [11], Aarts [1], Aarts and Driessen [2], Claus and Kleitman [21], Granot and Hubermann [60, 61], Kent [75], and certainly quite a few others; for matching games by Shapley and Shubik [98], and Solymósi and Raghavan [101]. Moreover, there has been work on a number of other combinatorial optimization games – see the article by Deng, Ibaraki, and Nagamoch [29] and the references in [75, 79], to name just three recent works.

In Chapter 3, we present the papers [45, 46, 47]. We describe geometrically motivated approaches to deal with empty and non-empty cores.

In Section 3.1, we consider Traveling Salesman games, the subject of paper [45]. We give a geometrically motivated allocation rule for the cost of a roundtrip. This allocation can be computed in polynomial time and allows it to bound the deficit. We can show that this bound is optimal for the worst case, provided that a well-known conjecture about a TSP relaxation is true. Section 3.2 deals with Minimum Cost Spanning Tree (MCST) games. In the paper [46], we show that it is CoNP-hard to decide whether a cost allocation is fair or not, implying that it is computationally hard to optimize a linear function over the core of an MCST game. In Section 3.3, we discuss Traveling Preacher games and illustrate the difference that it makes if there is no fixed central node for the game. We present a new result [54] showing that an analogue to the TSP game without a depot can be treated in polynomial time, even though the related polytope is described by an exponential number of facets, each of which is described by a problem that is computationally hard. In Section 3.4, we describe a number of solution concepts, including the nucleolus and its multiplicative analogue, the *nucleon*, which we introduce in [47]. We also give a polynomial algorithm for computing the nucleon of general matching games.

Chapter 2

Geometric Ideas for Graph Representation

2.1 Visibility Representation of Graphs

Determining a *visibility representation* of a graph, where the vertices of the graph map to disjoint sets or objects in the plane and the edges are expressed as visibility relations between these sets, has received considerable attention. Even before the described surge in graph drawing, motivations came from the large number of applications in areas such as VLSI wire routing, algorithm animation, CASE tools, and circuit board layout.

A visibility representation is defined by specifying a class of objects to represent the vertices and the visibility relation between the objects that is used to describe edges. In two dimensions, common choices for objects are axis-aligned line segments or rectangles. Two visibility relations that have been considered previously are defined as follows:

- Two objects are (*mutually*) *visible* if and only if they can be joined by a line segment that does not intersect any other object. Often, the direction of the line segment is restricted to be axis-parallel.

- Two objects are ϵ-*visible* if and only if they can be joined by a family of parallel line segments such that no segment intersects any other objects, and the union of the segments covers a region of positive area, i.e., positive "width" ϵ. Again, the direction of the line segments is often restricted to be axis-parallel.

For example, in Figure 2.1, line segments 1 and 2 are visible but not ϵ-visible. Line segments 2 and 3 as well as line segments 1 and 3 are ϵ-visible. However, line segments 1 and 3 are not vertically ϵ-visible. Line segments 2 and 4 are not visible by either of the above notions of visibility. Although these two different types of visibility seem closely related, the restrictions imposed by ϵ-visibility often radically change the class of graphs that admit visibility representations. See [103] for a discussion of the different definitions of visibility and the sensitivity of results to the choice of visibility definition.

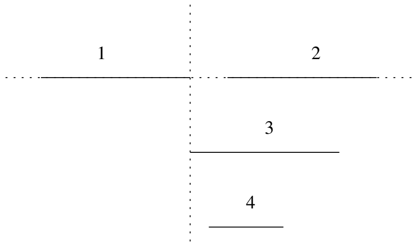


Figure 2.1: An illustration of visibility relations.

In two dimensions, the problem of characterizing graphs that can be represented has been studied quite well. Wismath [110] and Tamassia and Tollis [103] independently showed that every 2-connected planar graph admits a visibility representation where the vertices are represented by closed disjoint horizontal line segments in the plane and where two vertices are adjacent if and only if their corresponding segments are vertically ϵ-visible. (More precisely, they showed that such a representation exists for a graph if and only if the graph is planar and has an embedding with all cut vertices on the exterior face.) For the same model, Kant et al. [74] studied the visibility representations of trees. When the vertices are represented by disjoint axis-aligned rectangles in the plane and visibility is defined as ϵ-visibility in the horizontal and vertical directions, Wismath [110] showed that every planar graph admits a visibility representation. Dean and Hutchinson [28] proved that K_8 is the largest complete graph that admits a representation in this model. For an overview of the various visibility representations studied, see [32].

In 1993, a group of researchers (Prosenjit Bose, Hazel Everett, Sándor Fekete, Joan Hutchinson, Goos Kant, Marc van Kreveld, Beppe Liotta, Anna Lubiw, Henk Meijer, Kathleen Romanik, Tom Shermer, Steve Skiena, Roberto Tamassia, Yanni Tollis, Godfried Toussaint, and Sue Whitesides) gathered at the Bellairs Research Institute of McGill University to investigate new ideas of three-dimensional graph drawing. One of these concepts was visibility representation in three dimensions: Consider an arrangement of closed, disjoint rectangles in $\mathbb{R}^3$ such that the planes determined by the rectangles are perpendicular to the z-axis, and the sides of the rectangles are parallel to the x- or y-axes. Two rectangles R_i and R_j are *z-visible* if and only if between the two rectangles there is a closed cylinder C of positive length and radius such that the ends of C are contained in R_i and R_j, the axis of C is parallel to the z-axis, and the intersection of C with any other rectangle in the arrangement is empty.

Figure 2.2 shows a visibility representation of the graph from Figure 1.1, as seen from the top. This way of drawing the graph is not only better structured, it also allows for conveying other information: the size of the rectangles corresponds to the importance of the files, the edges are directed from bottom to top, input is black, output is white. Other properties of vertices can be given by use of colors, patterns, labels, etc.

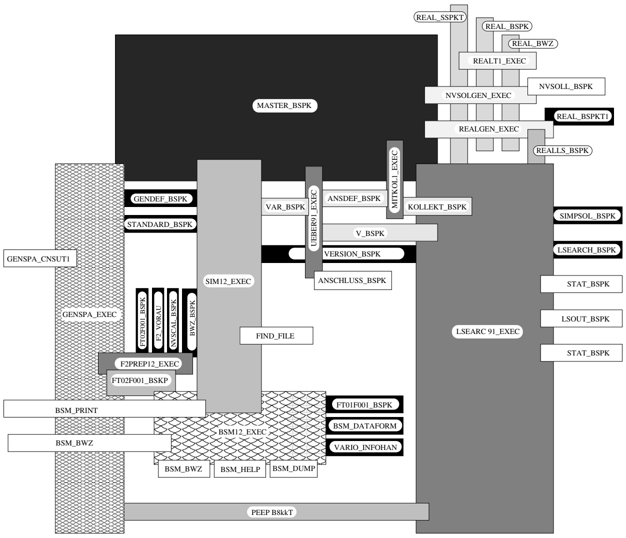


Figure 2.2: A visibility representation of the software package

What graphs can be represented using this concept? As was shown by Bose, Everett, Fekete, Lubiv, Meijer, Romanik, Shermer, and Whitesides [13, 14], the complete graph K_{20} does have a representation, while K_{103} does not have one. All planar graphs can be represented, but the class of representable graphs is not closed under deletion or contraction of edges, i. e., taking minors. This excludes characterization by a finite family of “excluded minors” along the lines of Robertson and Seymour [93].

This research has motivated a number of other conference papers such as [4, 22, 66, 94]. Closely related questions have been examined in [28, 78].

In the following sections, we give an overview over the papers [51, 53, 52]. The first establishes significant new bounds on representable graphs, the second considers "thick" objects instead of "flat" rectangles, while the third one considers the computational complexity of deciding whether a particular graph can be represented.

2.2 New Results on Visibility Representations

The main results of the paper [51] can be summarized as follows:

- K_{56} has no representation by arbitrary rectangles.
- K_7 (and hence K_n for all $n \leq 7$) has a representation by unit squares, but no K_n for $n \geq 8$ has a representation by unit squares.
- Any K_n has a representation by unit discs.

After the presentation of [51] at the "Graph Drawing '95", Rote and Zelle [95] obtained a visibility representation of K_{22} by using simulated annealing techniques on our combinatorial characterization. In the following, we give a short sketch of our results.

2.2.1 An upper bound of 56

We consider sequences of n rectangles lying parallel to the x, y-plane in $\mathbb{R}^3$, and ordered by increasing z-coordinate. We call a sequence *valid* if its associated visibility graph is K_n. Consider the projections of all the rectangles in a valid sequence onto the x, y-plane. Because each two must intersect and the objects are axis-aligned rectangles, application of a Helly-type theorem [26] shows that the intersection of all the projections must be non-empty. Thus we can choose a common point O (henceforth regarded as the origin) belonging to the interior of each of the projections. To simplify the notation, we do not distinguish between a rectangle and its projection onto the x, y-plane; the meaning will be clear from the context.

Each rectangle R in a valid sequence can be described in terms of the perpendicular distances from O to each of its sides. Instead of giving the x, y-coordinates of the vertices of R, we describe R as a 4-tuple (E_r, N_r, W_r, S_r) whose coordinates give, respectively, the distances from $O \in R$ to the east, north, west and south sides of R.

We can assume without loss of generality that no two rectangles of a valid sequence share the same value on any of the four coordinates E, N, W, S. Hence we can assume that each coordinate value of each of the n rectangles is an integer in the range $[1, n]$ without changing the visibility relationships among the rectangles.

Consider two rectangles $A = (E_a, N_a, W_a, S_a)$ and $B = (E_b, N_b, W_b, S_b)$ in a valid sequence. Then the following lemma can be shown:

Lemma 2.2.1 (Fekete, Houle, Whitesides [51]) *In a visibility representation of a complete graph, the rectangles A and B can see each other if and only if one of the following "free corner" conditions F holds* simultaneously for all *the rectangles R between A and B:*

1. *($E_r < \min\{E_a, E_b\}$ OR $N_r < \min\{N_a, N_b\}$);*
2. *($N_r < \min\{N_a, N_b\}$ OR $W_r < \min\{W_a, W_b\}$);*
3. *($W_r < \min\{W_a, W_b\}$ OR $S_r < \min\{S_a, S_b\}$);*
4. *($S_r < \min\{S_a, S_b\}$ OR $E_r < \min\{E_a, E_b\}$).*

This allows it to consider purely combinatorial conditions. In particular, we can use some properties of permutations. In the following, we denote by V_E, V_N, V_W, V_S the sequences of integers obtained by projecting the rectangles onto their E, N, W and S coordinates, respectively, for a given valid sequence of n rectangles $(E_1, N_1, W_1, S_1), \ldots, (E_n, N_n, W_n, S_n)$.

Definition 2.2.2 *A sequence of distinct integers is called* unimaximal *if it has exactly one local maximum.*

Lemma 2.2.3 *For all $m > 1$, in every sequence of $\binom{m}{2} + 1$ distinct integers, there exists at least one unimaximal subsequence of length m. On the other hand, there exists a sequence of $\binom{m}{2}$ distinct integers that has no unimaximal subsequence of length m.*

This result arises from the Erdős-Szekeres Theorem (1935), whose pigeon-hole proof was given by [64]. Lemma 2.2.3 is attributed by F. R. K. Chung [20] to V. Chvátal and J. M. Steele, among others.

Lemma 2.2.4 (Fekete, Houle, Whitesides [51]) *In a representation of K_5 by five rectangles, with no other rectangles present, it is impossible that both sequences V_N and V_S are unimaximal.*

Sketch. With the help of Lemma 2.2.1, it is possible to get a contradiction: If there was a representation as stated, then all rectangles A and B could see each other along a line of sight with y-coordinate 0. By intersecting the arrangement of five rectangles with the x, z-plane, we would get an arrangement of 5 line segments in this plane that all see each other. This contradicts the fact that only planar graphs can be represented by vertical visibility of horizontal line segments in a plane. (See [103] and [110] for results on such representations in the plane.)

∎

Theorem 2.2.5 *No complete graph K_n has a representation by arbitrary rectangles for $n \geq 56$.*

Proof. Suppose we had a representation of K_n with $n \geq 56$. Lemma 2.2.3 implies that V_N has a unimaximal subsequence V'_N of length 11. Consider the associated subsequence V'_S of length 11. It follows again from Lemma 2.2.3 that there is a subsequence V''_S of length 5 that is unimaximal. Remove the rectangles not associated with the subsequence. This destroys no visibility lines, so the five remaining rectangles represent K_5. However, both V''_S and its corresponding subsequence V''_N are unimaximal. This contradicts Lemma 2.2.4.

∎

2.2.2 Unit squares

While it is still a challenge to narrow the gap between the known upper and lower bounds for representation of K_n by rectangles or general squares, we show in the following that the largest possible n for which K_n has a representation by squares of equal size is $n = 7$.

Theorem 2.2.6 *K_7 has a representation by unit squares.*

Proof. See Figure 2.3. The six phases indicate how the seven squares are placed on top of each other. ∎

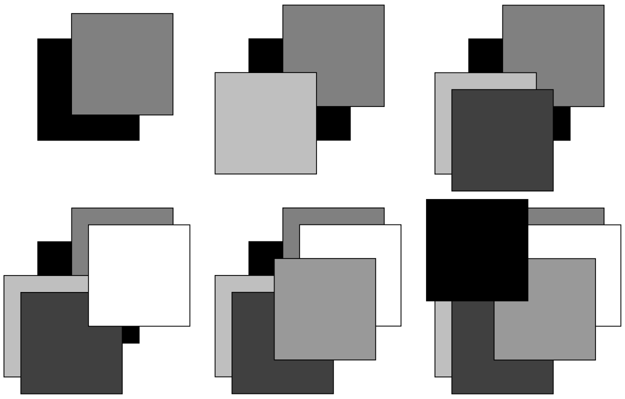


Figure 2.3: Representing K_7 by unit squares

Theorem 2.2.7 *K_8 does not have a representation by unit squares.*

Proof. We did the proof by a computer search that enumerates all maximal valid sequences of squares. The search begins with a single square. It proceeds in a depth-first-search manner, examining all the ways to add a new square above the squares in the valid sequence currently considered. It tests validity by checking the free corner conditions of the previous section. Whenever a valid extension is discovered, an immediate attempt is made to extend it further.

The resulting program is extremely simple, and can easily by verified by hand. Using this depth-first-search strategy, we were able to generate all maximal valid sequences in three minutes running time on a SPARC 1. There were 2064 of these maximal valid sequences, all of which had length 7.

2.2.3 Discs

After the results on rectangles and squares, we examine the situation for discs. We show that any K_n can be represented.

Theorem 2.2.8 *Any K_n can be represented by unit discs.*

Sketch. See Figure 2.4. Let $\frac{1}{2} < r < 1$ and consider the circle c_0 of radius r centered at the origin O. Now pick n points $m_1 = (x_1, y_1, 0), \ldots, m_n = (x_n, y_n, 0)$ equally spaced and in counter-clockwise order on an arc of c_0 of length less than half the circumference of c_0. Without loss of generality, assume that all $y_i < 0$.

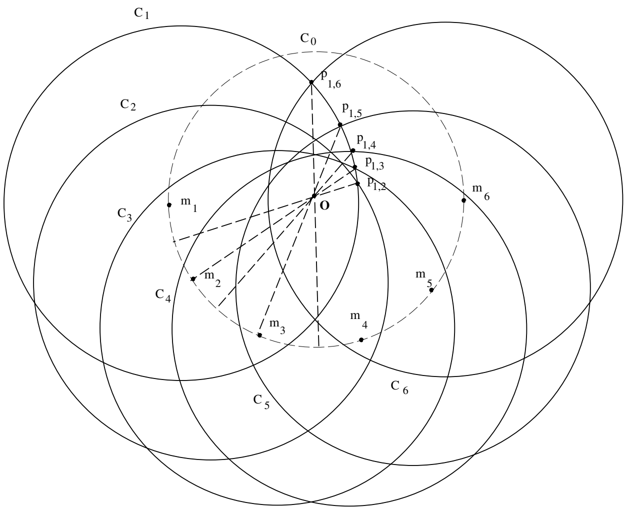


Figure 2.4: Representing K_7 by unit squares

Let $M_i = m_i + (0, 0, i)$. Let C_i be the unit disc parallel to the x, y-plane with center M_i, and let c_i be its projection onto the x, y-plane. We claim that the C_i form a representation for K_n.

It suffices to show that for each i, $1 \leq i < n$, all the discs above C_i can see C_i, i. e., C_j can see C_i whenever $j > i$. This can be done using elementary geometry. ∎

2.3 Representation by Thick Objects

The idea of visibility representations is not limited to lower-dimensional objects in a higher-dimensional space. In fact, visibility representation of graphs by rectangles in two-dimensional space has been studied before – see Wismath [110] for a proof that every planar graph admits a visibility representation. Dean and Hutchinson [28] proved that K_8 is the largest complete graph that admits a visibility representation in this model. This makes it natural to consider representation by full-dimensional boxes, with visibility parallel to the three coordinate directions. A representation of this type is called a *box visibility representation* (BR), whereas the situation for “flat” rectangles as discussed in the previous section is denoted by the term *z-axis parallel visibility representation* (ZPR). As Bose, Josefczyk, Miller, and O’Rourke [16] showed, K_{42} has a BR.

An additional motivation for considering box visibility graphs comes from the area of 3-dimensional packing algorithms, which are of importance for real-life logistics: when using certain local insertion heuristics for packing boxes into a container, the data structure for representing the packing is a box visibility graph. (See [112].) This application makes it interesting to consider representations with a limited number of different objects, corresponding to the situation where parcels can only have few different sizes. This question generalizes and extends the results of [51] about visibility representation by unit squares.

After our work [53] was already accepted for publication, Biedl, Shermer, Whitesides, and Wismath [10] considered BR's in the context of orthogonal drawings of graphs in three dimensions. Using Ramsey theory and our upper bound of 55 on the size of a complete graph with a ZPR, they showed that the complete graph K_{R^*} with the Ramsey number $R^* = R(56, 56, 56)$ does not have a BR. Of course, this number is totally beyond any comprehension. As we show in our paper [53], an upper bound of 183 can be established; in fact, this number can be stated as $3 \times z^* + 18$, where z^* is the size of the largest complete graph with a ZPR. Conversely, we can show a lower bound: All complete graphs with up to $2 \times z^* + 12$ have a BR.

The paper [53] consists of four parts. The first one considers upper and lower bounds for the size of complete graphs with a ZPR, when the number of different shapes is limited. The second part gives lower bounds for BR's, showing that the best upper bounds on ZPR's can be beaten by BR's. The third part proves an upper bound, while the fourth deals with upper bounds when the number of different shapes is limited.

2.3.1 ZPR's with A Limited Number of Shapes

The following Table 2.1 summarizes the results for ZPR's; "min" indicates the best known lower bound, "max" the best known upper bound for the given number of shapes or sizes.

2.3.2 Lower bounds on Box Visibility Graphs

The most notable results are the following:

Theorem 2.3.1 (Fekete, Meijer [53]) *K_8 has a BR by unit cubes.*

Proof. Figure 2.5 shows the projection of 8 unit cubes into the x,y-plane. The cubes have edges of length 1 and the number written inside each projected cube is

Table 2.1: Lower and upper bounds for ZPR's

shapes	min	max	sizes	min	max
1	12	14	1	7	7
2	18	28	2	12	14
3	20	42	3	18	21
4	20	55	4	20	28
5	20	55	5	20	35
6	22	55	6	20	42
..	..	..	7	20	49
			8	20	55
			9	20	55
			10	20	55
			11	22	55
			..	..	..

the z-coordinate of the bottom face of the cube. It can easily be verified that the visibility graph of this arrangement of cubes is complete.

This beats the upper bound of 7 on ZPR's with unit squares which we stated in Theorem 2.2.7. The bound of 55 for general ZPR's can be beaten as well:

Theorem 2.3.2 (Fekete, Meijer [53]) K_{56} *has a BR.*

Proof. See Figure 2.6. The mid sections consist of two BR's for K_{22}, which are obtained by representing the rectangles in the ZPR of K_{22} (see [95] or our paper [53]) by "thin" boxes.

2.3.3 Upper bounds on Box Visibility Graphs

It can be shown that any edge (a, b) of a complete graph K_n with a particular BR can be considered to have one of three "colors" (the boxes A and B representing a and b are either x-, y-, or z-visible) and a "direction" ($A \prec B$ or $B \prec A$ in the partial order induced by the projection on the appropriate coordinate axis.) In particular, we consider chains and antichains in the three partial orders. This can be used to prove that

(a) particular ways of coloring and/or orienting edges cannot occur in representations of complete graphs

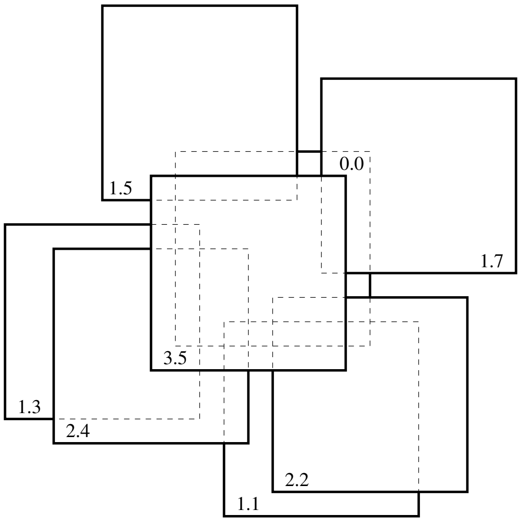


Figure 2.5: BR of K_8 using unit cubes, i. e., one size

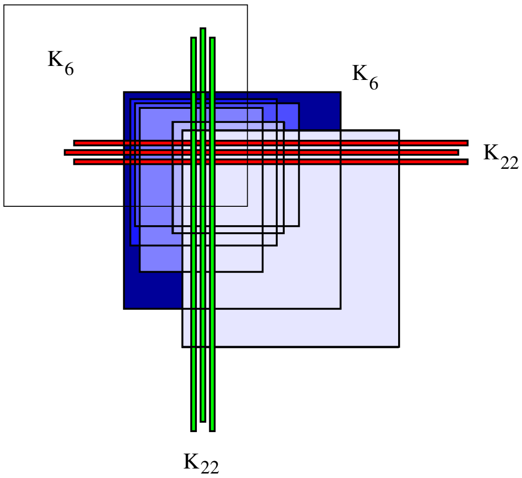


Figure 2.6: BR of K_{56}

(b) excluding certain structures limits the size of complete graphs with a representation.

With some effort, the following two lemmas can be proved:

Lemma 2.3.3 (Fekete, Meijer [53]) *In a box visibility representation of a complete graph, let C be a maximum length x-chain. There cannot be an x-chain D of length $\geq$ 5 such that $C \cap D = \emptyset$.*

Lemma 2.3.4 (Fekete, Meijer [53]) *Suppose there is no chain longer than 4 in a box visibility representation of a complete graph G. Then G can have at most 18 vertices.*

This implies the following theorem:

Theorem 2.3.5 *There is no BR for K_{184}.*

Proof. Let C be a set of boxes whose visibility graph is complete. By Theorem 2.2.5, a largest x-chain C_x in C cannot exceed length 55. By Lemma 2.3.3, there cannot be an x-chain of length 5 that is disjoint from C_x. This means that after removing a maximum size x-chain, a maximum size y-chain, and a maximum size z-chain from C we are left with a BR such that there are no chains longer than 4. As shown in Lemma 2.3.4, this BR contains at most 18 boxes, implying that the size of C is at most $3 \times 55 + 18 = 183$.

∎

By a long sequence of lemmas, the following can be proved for the case of unit boxes:

Theorem 2.3.6 (Fekete, Meijer [53]) *There is no BR for K_{10} using unit cubes.*

This leaves a gap of size 1 between the best known lower bound and the best proven upper bound. The arguments used for proving Theorem 2.3.6 give a large amount of structure for any potential BR of K_9 by unit cubes. We expect that lowering the upper bound to 8 by going through all the cases is possible but very tedious.

Conjecture 2.3.7 *K_9 does not have a BR by unit cubes.*

2.4 Proving Hardness of Graph Representation Problems

When it comes to algorithms, the area of automated graph drawing has largely been shaped by its limitations. Many popular aesthetics for graph representation, such as the desire for planar layouts without edge crossings, are realizable only in the most restricted of application settings. Relaxation of the aesthetic considerations, such as allowing a small number of edge crossings, often leads to difficult (that is, NP-hard) computational problems.

As we mentioned before, the advent of better architectures and software for graphics has made it possible to circumvent some of these limitations by visualizing graphs in three dimensions instead of just two. For example, whereas edge crossings are usually unavoidable in two dimensions, in a 3-dimensional setting they may generally be eliminated altogether. Also, by navigating through a 3-dimensional image, the user can often avoid views of the data which are aesthetically unappealing or confusing.

Although the 3-dimensional setting allows for a richer variety of graph representations, it is also true that the increase in dimensionality brings with it new mathematical and computational challenges. In order to open the way for good heuristics, or to justify the relaxation of some of the objectives, it is often a sensible first step to determine whether computation of the representation is an NP-hard problem.

One of the tools for establishing NP-hardness of graph realization problems is the so-called *logic engine*. It was first used by Bhatt and Cosmadakis [8] to show that it is an NP-complete problem to determine whether a tree of maximum degree 4 can be embedded in a planar grid, where tree vertices must be positioned at grid vertices and tree edges must occupy unit length grid edges. The logic engine was later employed to obtain lower bounds for checking whether a graph can be embedded in the plane as a proximity graph [34, 38, 39].

The technique can be summarized as follows (see the above references for more details; an overview will be contained in the upcoming book [33]). The reduction is from the NP-complete problem "Not-All-Equal-3SAT" (NAE-3SAT), whose instances consist of c 3-element clauses on s literals. "Yes" instances are ones with truth assignments such that each clause contains at least one true literal and at least one false literal (see [58]).

For every instance of the NAE-3SAT problem, a machine can be built such that a non-self-intersecting layout of the machine in the plane exists, if and only if the answer to the corresponding NAE-3SAT instance is "yes". In its most general sense, the machine consists of a rigid frame with vertical left and right sides joined at their

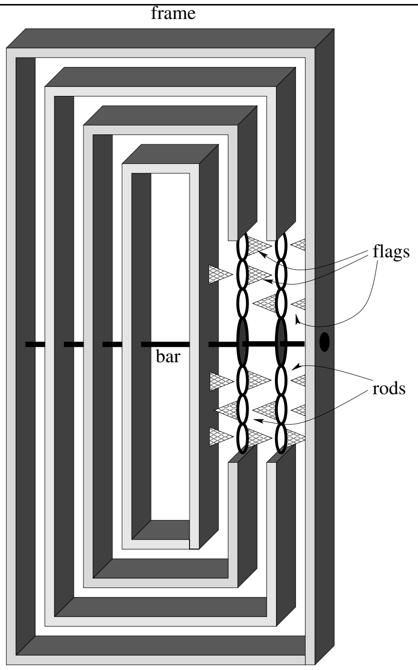


Figure 2.7: A logic engine for the NAE-3SAT instance $(x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_3 \vee \overline{x}_4) \wedge (x_1 \vee x_3 \vee x_4)$. The shown position corresponds to the truth setting $x_1 = 1$, $x_2 = 1$, $x_3 = 1$, $x_4 = 0$.

midpoints by a horizontal bar. Attached to this bar are vertical rods that can rotate independently about the bar. One half of each rod corresponds to some variable X and the other half to its complement X'. Each half-rod has c positions where flags may be placed. The flags may rotate independently about the vertical rods. Clauses are represented by pairs of rows of flag positions. Clause C_i is represented by the row i units above the bar and the row i units below the bar. The contents of each row depends on how the vertical rods are flipped around the bar. The region above the bar is interpreted as the "true" region, the region below is interpreted as the "false" region.

Once this has been established, it is simple to encode a particular instance: if a literal Y = X or X' fails to appear in a given clause C_i, then a flag is positioned on the vertical rod for (X, X'), in the half corresponding to Y and at distance i from the bar. An unfilled position in the i^{th} row above the bar corresponds to the occurrence of a true literal in C_i; similarly, an unfilled position in the i^{th} row below the bar corresponds to the occurrence of a false literal in C_i. Rods and flags are to be turned and flipped so that flags on the left-most and right-most vertical rods point inward, and so that no flags positioned at the same distance i from the bar on two adjacent half-rods face each other. This can be done if and only if the NAE-3SAT instance is a "yes" instance.

To apply the logic engine to a geometric graph representation problem, one first has to find a special graph whose spatial embedding is "rigid"; that is, one in which relative positionings of points in the embedding is fixed. The frame, rods and flags of the logic engine are then constructed using this graph as a building block. The problem with this approach is that for a large number of representations, there are no graphs for which the relative positioning of any points in the embedding is necessarily fixed. For example, no geometric realization of an intersection graph of open squares is rigid, as any of the squares can be perturbed to some degree relative to the others without changing the intersection graph.

The main contribution of our work [52] is a variant of the logic engine which can be applied to graph representation problems in which no rigid embeddings are possible. In place of rigid structures, we show how "wobbly" equivalents can be built such that the variation in position of any element of the engine, relative to any other element, is bounded. More precisely, we consider particular elements, called *springs*, having upper and lower bounds on their size. By building an appropriate structure from several springs, we can force some of them to be stretched close to their upper limits, while some are compressed to their lower limits. As a consequence, the overall variation can be made arbitrarily small with respect to the size of the engine, in such a way that the movements of the wobbly logic engine mimic those of a rigid logic engine. These springs can be designed for many problems, allowing NP-hardness proofs by constructing wobbly logic engines.

In our paper [52], we present this wobbly logic engine technique and apply it to graph intersection problems, and visibility representation problems. In particular, we give an NP-hardness proof for the problem of deciding whether a graph has a nondegenerate z-axis parallel visibility representation (ZPR) by unit squares. Basic building block is a class of graphs called *k-extensors*. These graphs are used as springs for the construction of a wobbly logic engine. We conclude this section with a discussion of other applications of the wobbly logic engine.

2.4.1 k-extensors and springs

An *extensor graph of k links* (k-extensor) is a tripartite graph of the form $G_k = (U_k, V_k, W_k, E_k)$, where

$$\begin{aligned} U_k &= \{u_0, u_1, \ldots, u_k\}, \\ V_k &= \{v_0, v_1, \ldots, v_{k-1}\}, \text{ and} \\ W_k &= \{w_0, w_1, \ldots, w_{k-1}\}. \end{aligned}$$

Edges of E_k are of the form

$$\begin{array}{ll} (v_{i-1}, u_i), (w_{i-1}, u_i) & \text{for } 1 \leq i \leq k, \text{ and} \\ (u_i, v_i), (u_i, w_i) & \text{for } 0 \leq i \leq k-1. \end{array}$$

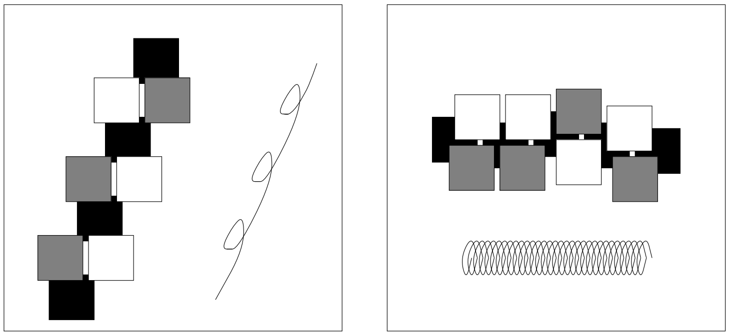
The vertices u_0 and u_k are called the *tabs* of G_k.

Now the following can be shown:

Lemma 2.4.1 (Fekete,Houle,Whitesides [52]) *Let* G *be a graph having the* k*-extensor* $G_k = (U_k, V_k, W_k, E_k)$ *as an induced subgraph, for some* $k \geq 2$*. Furthermore, let the only edges joining vertices of* $G \setminus G_k$ *with those of* G *be incident to the tabs* u_0 *and* u_k*. Let* (x_i, y_i) *be the* x,y*-coordinates of the centers of symmetry of* u_i*, for* $0 \leq i \leq k$*. For any valid non-degenerate layout of* G*:*

- *If* G_k *is vertical, then* $(x_k, y_k) \in (x_0, y_0) \pm \{(-k, k) \times (k, 2k)\}$.
- *If* G_k *is vertical, then* $(x_k, y_k) \in (x_0, y_0) \pm \{(k, 2k) \times (-k, k)\}$.

This makes it possible to think of the layout of a k-extensor as a flexible *spring* that can be compressed and stretched within the limits described in Lemma 2.4.1. See Figure 2.8. These springs can be used to construct a wobbly logic engine for the problem ZPR-NDUS, i. e., deciding whether a given graph has a non-degenerate z-parallel visibility representation by unit squares.



stretched spring compressed spring

Figure 2.8: The layouts "stretched spring" and "compressed spring" for a k-extensor, and their symbols

2.4.2 Overall layout

The idea for the construction is roughly as follows – see Figure 2.9 for an overview. Using "connector rings", we can combine several springs to "blocks" that have very small relative flexibility; these blocks can be used to build frame, rods, and flags of a logic engine. To allow free rotation of the rods and flags in the logic engine, we use special "pivot connector rings". The overall engine has the same basic properties as a rigid logic engine.

Summarizing, we state:

Theorem 2.4.2 (Fekete, Houle, Whitesides [52]) *The problem ZPR-NDUS is NP-hard.*

2.4.3 Other applications

The original logic engine has been used for a variety of "rigid" graph layout problems. One of them is the representation of a graph as the intersection graph of closed objects like unit squares or unit disks. It is not hard to see how degenerate arrangements can be forced that can be used as rigid building blocks for a logic engine.

On the other hand, these degenerate arrangements are necessarily unstable under small perturbations. This makes it interesting to require "fat" intersection of objects, which amounts to considering open objects only. The wobbly logic engine can be used to prove NP-hardness of a number of these problems. In this section, we sketch how we can prove NP-hardness of the problem of deciding whether a given graph is an intersection graph of open unit squares or open unit cubes.

As an immediate corollary of Theorem 2.4.2, we get

Corollary 2.4.3 *The problem* Open Unit Square Intersection Graph *is NP-hard.*

With a number of modifications, it is possible to deal with a higher-dimensional version of this problem.

Theorem 2.4.4 *The problem* Open Unit Cube Intersection Graph *is NP-hard.*

In the same way, we can generalize even further:

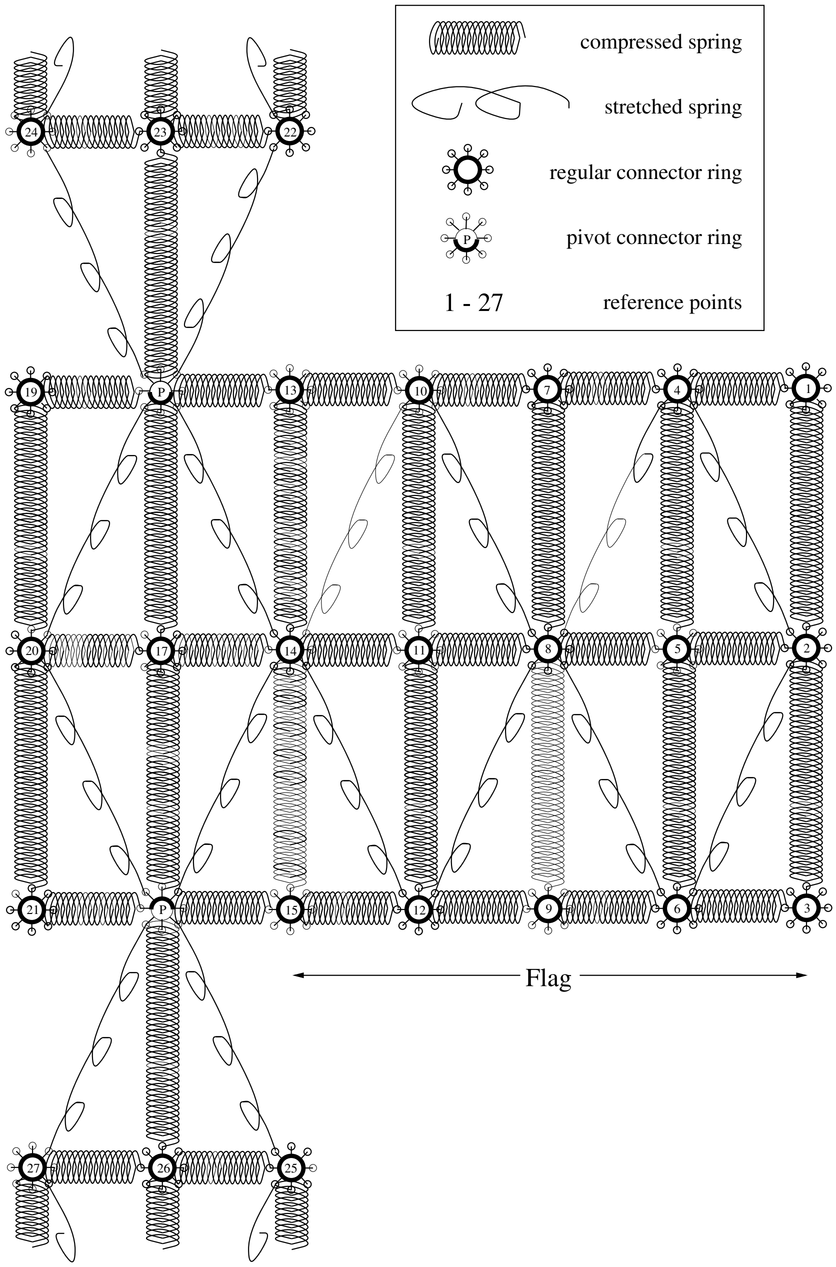


Figure 2.9: A flag in a wobbly logic engine

Theorem 2.4.5 *The problem* Open Unit Hypercube Intersection Graph *is NP-hard.*

The wobbly logic engine technique can be applied to classes of objects other than squares or cubes. In particular, the following can be shown:

Theorem 2.4.6 *The problem* Open Unit Disk Intersection Graph *is NP-hard.*

Chapter 3

Geometric Ideas for Cooperative Games

3.1 Traveling Salesman Games

Traveling Salesman games (TSP games) are our first example of a cooperative cost game.

Definition 3.1.1 *Let* $N = \{1, \ldots, n\}$*,* $N_0 := N \cup \{0\}$ *and consider the complete undirected graph with vertex set* N_0 *and edges weighted with non-negative real numbers* d_{ij}*,* $i, j \in N_0$*. Define for every coalition* $S \subseteq N$*,*

$v(S) =$ *length of an optimal traveling salesman tour through all nodes in* $S \cup \{0\}$.

The special node 0 (the "depot") corresponds to the home base of the traveling salesman. In Section 3.3, we discuss some aspects of the situation where such a depot does not exist.

3.1.1 A smallest Euclidean example with empty core

A majority of TSP instances are geometrically motivated (see [82]). There are some situations where geometry allows for some special possibilities. Most notably, the independent works of Arora [5] and Mitchell [84] showed in 1996 that the Euclidean TSP has a polynomial time approximation scheme, using geometry in connection with dynamic programming. It is well known that the general TSP cannot be approximated arbitrarily well in polynomial time.

Kuipers [79] showed that under quite general assumptions (including the Euclidean case), every TSP-game with up to 5 players has a nonempty core. Figure 3.1 depicts an instance of a 6-person Euclidean TSP-game with empty core.

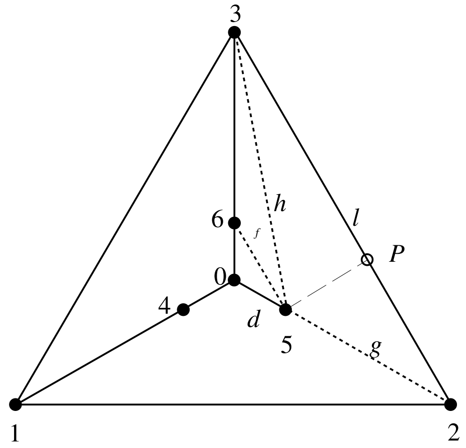


Figure 3.1: A minimum Euclidean example with empty core

Now consider the subsets $S_1 = \{1, 2, 4, 5\}$, $S_2 = \{1, 3, 4, 6\}$, $S_3 = \{2, 3, 5, 6\}$; each player is covered exactly twice by these subsets, so the sum of their allocations must not exceed twice the length of an optimal tour. By an elementary computation it follows that this condition is violated whenever

$$\frac{l}{10\sqrt{3}} \leq d \leq \frac{l}{3\sqrt{3}},$$

implying that the core of the game is empty.

Polyhedrally speaking, this proof of emptiness can be formulated as a fractional cover by subcoalitions that is better than the best integer cover – the whole set. It should be noted that Kern [76] has conjectured that the core is empty, if and only if there are precisely three nonzero coefficients, and they all have value $\frac{1}{2}$.

3.1.2 Geometric cost allocation

If it is not possible to allocate the total cost of a tour without violating one of the core constraints (i. e., overcharging a subset of players), we are forced to run a deficit. More precisely, we say that an allocation is an ε-approximate core allocation, if the relative deficit does not exceed ε.

Dealing with deficits is a common problem when it comes to allocating the cost of transportation. See Figure 3.2 for an example showing the fare zones of Hamburg.

It represents a common approach for charging for public transportation: passengers have to pay for fare zones they cross.

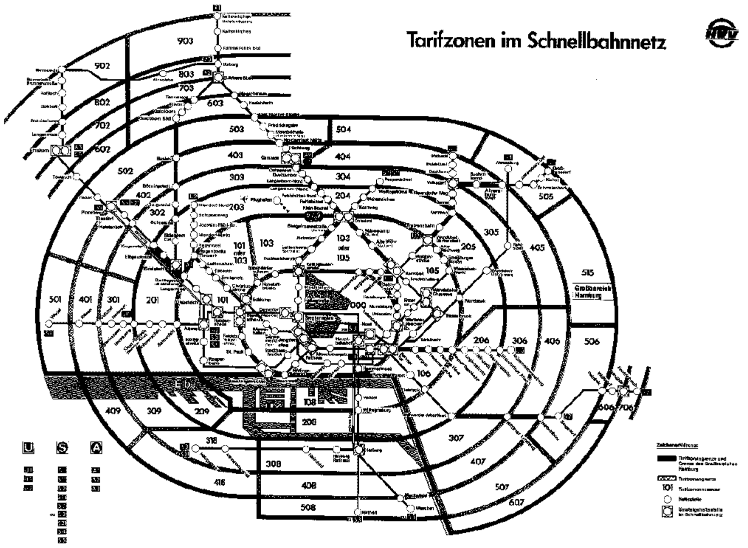


Figure 3.2: The fare zones of Hamburg

The idea of fare zones has been used in the context of combinatorial optimization problems by Jünger and Pulleyblank [73], who showed that for several problems of combinatorial optimization, the dual variables can be drawn geometrically as moats. For our purposes, we define:

Definition 3.1.2 *For a given set of vertices in the plane, a* moat *is a simply closed strip of constant width that separates two nonempty complementary subsets of the vertices. The* inside *of the moat is the region containing the depot, the other region we call the* outside. *A* moat packing *is a collection of moats with pairwise disjoint interior. The* cost *of a moat packing is twice the sum over all widths.*

See Figure 3.3 for a moat packing for an instance of cities in the American Midwest, distances are taken from Nemhauser and Wolsey [86]. Note that the moat packing is tight – no part of the tour is left uncovered, and each moat is crossed exactly twice. It is straightforward to see that this implies optimality of the tour.

Even if there is no tight moat packing, we can try to find a moat packing that covers as much as possible – twice the sum of the moat widths will always be a lower bound of the tour. Indeed, we can allocate the cost of a tour as follows: Distribute twice the width of any moat among the vertices on the outside (in an arbitrary way). It is intuitively clear that if the total cost of the moat packing is distributed this way,

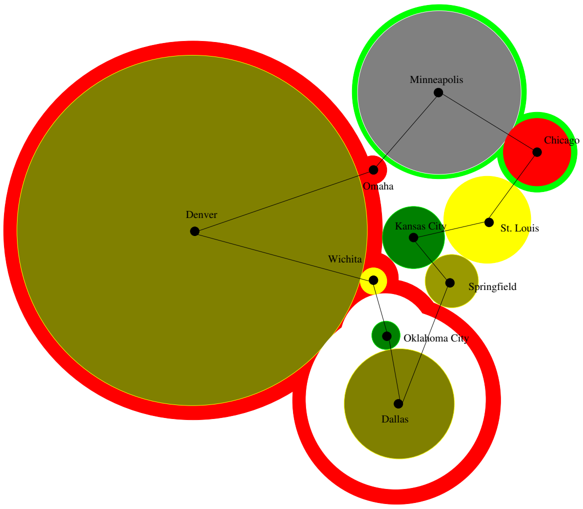


Figure 3.3: A moat packing for an instance of 10 cities in the Midwestern United States

the resulting distribution is such that no coalition pays more than its TSP cost. A formal proof is contained in the paper [45].

The allocated cost for any moat packing is twice the sum of the moat widths. Moat widths are required to be nonnegative and the sum of the widths of moats separating two vertices may not exceed their distance. This motivates the following linear program:

$$\max\left(2\sum_{S,\overline{S}\in\mathcal{M}} w_{\{S\overline{S}\}}\right)$$

(M) subject to the constraints

$$\begin{array}{rcccc} w_{\{S\overline{S}\}} & \geq & 0 & \text{for all} & \{S,\overline{S}\}\in\mathcal{M}, \\ \sum\limits_{(i,j)\in\delta(S)} w_{\{S\overline{S}\}} & \leq & d(i,j) & \text{for all} & i,j\in V. \end{array}$$

Here $\mathcal{M}$ denotes the set of all nontrivial partitions $(S,\overline{S})$ of V (assuming that the depot is contained in S) and $\delta(S)$ is the set of all edges that join a vertex from S to a vertex from $\overline{S}$. Note that there are exponentially many variables. Furthermore, there may be solutions to the linear program which do not correspond to a moat packing, since there may be positive $w_{\{S_1\overline{S_1}\}}$, $w_{\{S_2\overline{S_2}\}}$ with $S_1\cap S_2\neq\emptyset$ and $S_1\not\subseteq S_2$ and $S_2\not\subseteq S_1$, in which case the two corresponding moats are forced to intersect. However, as was shown in [24], for any instance of (M), there is an optimal solution for which the sets with $w_{\{S\overline{S}\}} > 0$ have the special structure of a *nested family*:

Definition 3.1.3 *A family of partitions* $(S_1, \overline{S_1}), \ldots, (S_k, \overline{S_k})$ *is called* nested, *if for any two partitions* $(S_i, \overline{S_i})$ *and* $(S_j, \overline{S_j})$*, we have* $S_i \cap S_j = \emptyset$ *or* $S_i \subseteq S_j$ *or* $S_j \subseteq S_i$.

How can we solve (M) in polynomial time? Consider its dual, i. e., the following linear problem:

$$(D) \quad \begin{array}{c} \min \sum_{(i,j)\in V^2} d(i,j)x_{ij} \\ \text{subject to the constraints} \\ \begin{array}{rclll} x_{ij} & \geq & 0 & \text{for all} & i,j, \\ \sum_{(i,j)\in\delta(S)} x_{ij} & \geq & 2 & \text{for all} & S, \overline{S} \in \mathcal{M}, \end{array} \end{array}$$

Using triangle inequality, we may add some constraints without changing the optimum, i. e., consider the following linear program (HK) instead of (D):

$$(HK) \quad \begin{array}{c} \min \sum_{(i,j)\in V^2} d(i,j)x_{ij} \\ \text{subject to the constraints} \\ \begin{array}{rclll} x_{ij} & \geq & 0 & \text{for all} & i,j, \\ x_{ij} & \leq & 1 & \text{for all} & i,j, \\ \sum_{(i,j)\in\delta(S)} x_{ij} & \geq & 2 & \text{for all} & S, \overline{S} \in \mathcal{M}, \\ \sum_{j\neq i} x_{ij} & = & 2 & \text{for all} & i \in V. \end{array} \end{array}$$

The feasible region for this second linear program is known as the *subtour polytope* S^n, the program itself as *Held-Karp relaxation*: Any of the constraints corresponding to a moat variable is a so-called *subtour elimination constraint*. (See our article [45] for some discussion and references on the subtour polytope.)

By a result of Padberg and Hong [89], the Held-Karp relaxation is solvable in polynomial time by means of the ellipsoid method (see Grötschel, Lovász, and Schrijver [62]). This implies that also a dual solution can be computed efficiently [62].

Assuming triangle inequality, the dual variables $(w_S^*)_{S \subseteq N}$ are non-negative even for singletons and their corresponding subsets can be assumed to form a nested family as described above. Therefore, the vector

$$x_i := 2 \cdot \sum_{\substack{i \in S \\ 0 \notin S}} \frac{w_S^*}{|S|}$$

is a core vector of the game defined by the cost function $c_{(HK)}$.

Using the following theorem, we can give a bound on the relative deficit:

Theorem 3.1.4 (Wolsey[111], Shmoys and Williamson[99]) *If the distances of a Euclidean TSP satisfy the triangle inequality, then the optimum value of* (HK) *is at least* $\frac{2}{3}$ *of the length of a shortest tour.*

It is a well-known open conjecture that the factor of 2/3 can be replaced by 3/4 (see [59] for an explicit statement and some motivation).

It should be noted that Shmoys and Williamson have shown an even stronger version of Theorem 3.1.4:

Theorem 3.1.5 (Shmoys and Williamson [99]) *If the distances satisfy the triangle inequality, then the optimum value of* (HK) *is at least* $\frac{2}{3}$ *of the length of a tour obtained by the method of Christofides.* ▮

This means we can guarantee an allocation of 2/3 of the cost of an *approximate tour* where both the allocation and the tour have been computed in polynomial time.

Summarizing, we state:

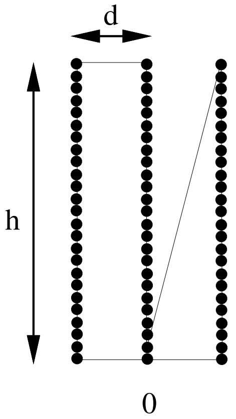
Theorem 3.1.6 (Faigle, Fekete, Hochstättler, Kern [45]) *For TSP games with triangle inequality, there are vectors* x *and* $y \in \mathbb{R}^N$*, which can be computed in polynomial time and satisfies the following conditions:*

- x *is an* ε*-approximate core allocation for* $\varepsilon = \frac{1}{2}$.
- *If the "*$\frac{4}{3}$*-conjecture" on the Held-Karp bound is true,* x *is an* ε*-approximate core allocation for* $\varepsilon = \frac{1}{3}$.
- y *guarantees an* ε*-approximate allocation of the cost of an approximate TSP tour obtained by the Christofides heuristic for* $\varepsilon = \frac{1}{2}$.

It should be noted that “for a large variety of randomly generated instance types, the optimal tour length averages less than 0.8% over the Held-Karp bound, and even for real-world instances in TSPLIB the gap is almost always less than 2%”. (Johnson, McGeogh, and Rothberg [68]). This also reflects the fact that it is possible to get good practical computational results, even though the theoretical polynomial running time involves the ellipsoid method.

3.1.3 $\frac{1}{3}$ is asymptotically best possible

The factor of $\frac{1}{3}$ given in Theorem 3.1.6 cannot be improved: for any $\delta > 0$ there is an instance of the TSP-game such that there is no $(\frac{1}{3} - \delta)$-approximate allocation. This is easily seen by considering the family of instances shown in the following figure.



3.2 Minimum Cost Spanning Tree Games

As we already mentioned in the Introduction, Minimum Cost Spanning Tree (MCST) games are the type of combinatorial optimization games that have received the most attention. This is mostly due to the importance of network problems; see Aarts [1] and Kent [75] for an overview.

However, there is probably an additional mathematical reason: As any combinatorial optimizer knows, the task of determining an MCST in a network is the prototype of an “easy” optimization problem, while the TSP is the standard “hard” problem. This makes MCST games the ideal candidate to explore concepts of cooperative game theory, like the core, whereas describing a single facet of the TSP core requires solving an NP-hard problem.

Formally, we can describe MCST games as follows.

Definition 3.2.1 *Let* $N = \{1, \dots, n\}$, $N_0 := N \cup \{0\}$ *and consider the complete undirected graph with vertex set* N_0 *and edges weighted with non-negative real numbers* d_{ij}, $i, j \in N_0$. *For every* $S \subseteq N$, *let*

$$v(S) = \textit{minimum cost of a tree spanning all vertices in } S \cup \{0\}.$$

The related cooperative game is called Minimum Cost Spanning Tree game *or* MCST-*game.*

As for TSP games, we have a “source” node 0. However, the core for MCST games is never empty – the following proposition shows a very simple way to compute an element of the core of an MCST-game.

Proposition 3.2.2 ([11, 21, 60]) *Let* T *denote a fixed minimum cost spanning tree. Let* $x_i := d_{ij}$, *where* ij *is the unique arc incident with* i *on the (unique) path from* i *to* 0 *in* T. *Then* x *is in a vertex of the core of the game, which we call the* greedy element *of the core.*

As Bird [11] showed, there are indeed many different (extremal) elements of the core. Which point from the polyhedron should be chosen? From an economic point of view, a good choice is to measure the preference for individual players by a linear utility function c, and optimize this function over the core. We will see in the following that this is not possible in polynomial time, provided that P≠NP. In fact, deciding whether a given allocation is in the core is a CoNP-hard problem – implying that it is impossible to convince the players of the fairness of an assignment, unless NP=CoNP.

Theorem 3.2.3 (Faigle, Fekete, Hochstättler, Kern [46]) *The following problem is NP-complete:*

Instance: *A weighted graph* $G = (V, E, d)$ *and* $x \in \mathbb{Q}^V$.

Problem: *Is* x not *a member of the core of the MCST-game defined by* G*?*

The proof is a reduction from EXACT COVER BY 3-SETS (X3C), one of the basic NP-complete problems in the book of Garey and Johnson [58].

Separation and optimization of a linear function over a polyhedron are polynomially equivalent (see [62]); we get the following corollaries.

Corollary 3.2.4 *A polynomial-time algorithm for optimizing over the core of an MCST-game implies NP=P.*

Corollary 3.2.5 *If there is a polynomial method of deciding whether a vector is in the core of an MCST-game, then NP=CoNP.*

This puts additional emphasis on the exploration of alternative choices of core elements. In Section 3.4, we introduce the solution concepts of the nucleolus and the nucleon, which can be interpreted as points that lie in the "center" of the core.

3.3 The Traveling Preacher Problem

For a combinatorial optimizer, it is no surprise that dealing with TSP games is a difficult task. In the previous section, we saw that even an "easy" combinatorial optimization problem like the MCST gives rise to hard questions on cooperative games. We will see in the following that this is not necessarily because of the fact that there are exponentially many restriction, but mostly due to the existence of a depot, creating a Steiner tree type problem.

We will demonstrate this with the following game, arising from TSP games by dropping the depot 0:

Definition 3.3.1 *Consider the complete undirected graph with vertex set N and edges weighted with non-negative real numbers d_{ij}, $i, j \in N$. Define for every coalition $S \subseteq N$, $|S| \geq 3$:*

$v(S)$ = *minimum length of a traveling salesman tour visiting all nodes in S.*

Instead of a salesman with a fixed home vertex, we consider a traveler who is willing to leave his home behind in order to serve a set of followers – consequently, we call this scenario the *Traveling Preacher game*. This motivates the following optimization problem:

$$\max \sum_{i=1}^{n} x_i$$

$$(TPP) \qquad \text{subject to the constraints}$$

$$x(S) \;\leq\; v(S) \quad \text{for all} \quad S \subseteq N, |S| \geq 3$$

Each of the exponentially many constraints involves the solution of an instance of the TSP – seemingly a hopeless task. However, we have the following theorem:

Theorem 3.3.2 (Fekete, Pulleyblank [54]) *The Traveling Preacher Problem can be solved in polynomial time.*

Proof. By the polynomial equivalence of linear optimization and separation over a polyhedron [62], it suffices to prove that separation over the set of feasible solutions can be done in polynomial time.

Therefore, consider a cost allocation $y = (y_1, \ldots, y_n)$ and for any edge $e = (i, j)$, define the following auxiliary edge weights:

$$w(e) := d_{i,j} - \frac{y_i}{2} - \frac{y_j}{2}.$$

In the complete graph with vertex set N and edge weights $w(e)$, the existence of a negative cycle can be decided in polynomial time, see [3]. Suppose we have such a negative cycle, without loss of generality let it be given by $S = \{1, \ldots, s\}$. Then

$$w(s,1) + \sum_{i=1}^{s-1} w(i, i+1) < 0 \Leftrightarrow d_{s,1} + \sum_{i=1}^{s-1} d_{i,i+1} < \sum_{i=1}^{s} y_i.$$

Clearly, the last inequality can hold if and only if there is a subset S with $y(S) > c(S)$. Thus, detection of a negative cycle can be used as a polynomial separation oracle, completing the proof.

∎

Note that it easy to modify the separation oracle to incorporate core restrictions arising from coalitions of two players; including restrictions by individual players makes the game trivial.

3.4 Solution Concepts and Matching Games

In this section, we consider several solution concepts. We introduce the new solution concepts of the *nucleon*, and demonstrate that it can be computed efficiently for the case of matching games. Since matching games deal with allocating savings instead of cost, the inequalities for describing the core etc. are reversed.

3.4.1 The additive ϵ-core and the nucleolus

Given a cooperative savings game $v : 2^N \to \mathbb{R}$, the core is a polyhedron that may empty. In order to find allocations that are still acceptable, Shapley and Shubik ([98]) suggested a way to shrink it uniformly or relax the constraints until there are feasible points:

For $\epsilon \in \mathbb{R}$, consider the modified game (N, v^ϵ), where

$$v^\epsilon(S) = \begin{cases} v(S) & \text{if } S \in \{\emptyset, N\} \\ v(S) - \epsilon & \text{otherwise.} \end{cases}$$

The *additive ϵ-core* of the game (N, v) is defined to be the core of the game (N, v^ϵ). Note that there is always some ϵ yielding a non-empty additive ϵ-core. What is the critical value of ϵ? See Figure 3.4 for a geometric and algorithmic interpretation: We want an allocation x that maximizes the *satisfaction*, i. e., the negative excess

$$e(x, S) = \sum_{i \in S} x_i \; - v(S)$$

uniformly over all proper coalitions S, i. e., we solve the linear program

$$\begin{array}{lrcll} (LP_1) & \max \epsilon & & & \\ & \sum\limits_{i \in N} x_i & = & v(N) & \\ & \sum\limits_{i \in S} x_i & \geq & v(S) + \epsilon & \text{for all } S \notin \{\emptyset, N\} \end{array}$$

Denoting by ϵ_1 the optimal objective function value of (LP_1), it follows that $\epsilon = -\epsilon_1$ is the minimal value admitting a non-empty additive ϵ-core. The set of optimal solutions of (LP_1) is usually called the *prenucleolus* of the game.

Since there may still be more than one element in the prenucleolus (as shown in Figure 3.4, it may be possible to optimize the satisfaction over this set. This refinement leads to the notion of the *nucleolus*, which was first proposed by Schmeidler[96]. Algorithmically, we can describe it as follows: If (LP_1) has a unique solution (ϵ_1, x^*), then x^* is the *nucleolus* of the game (N, v). Otherwise, there is a unique collection $\mathcal{S}_1 \subset 2^N$ of coalitions S $(\neq \emptyset, N)$ for which the inequalities in (LP_1) become tight at $\epsilon = \epsilon_1$. Now, in a second step, we maximize the satisfaction over all remaining coalitions:

$$
\begin{array}{llcll}
(LP_2) & \max \epsilon & & & \\
& \sum\limits_{i\in N} x_i & = & v(N) & \\
& \sum\limits_{i\in S} x_i & = & v(S)+\epsilon_1 & \text{for all } S \in \mathcal{S}_1 \\
& \sum\limits_{i\in S} x_i & \geq & v(S)+\epsilon & \text{otherwise}
\end{array}
$$

Continuing in this way, we obtain a sequence

$$\epsilon_1 < \epsilon_2 < \ldots < \epsilon_k$$

until, finally, the optimal solution of (LP_k) is unique with an allocation x^*, the *nucleolus* of the game.

A more concise (and less algorithmic) description can be given as follows.

With the allocation x we associate the *satisfaction vector* $s(x) \in \mathbb{R}^{2^n-2}$ as the vector of negative excesses arranged in non-decreasing order. The nucleolus is then the unique vector x^* that lexicographically maximizes the satisfaction vectors $s(x)$ relative to the game (N, v).

See Figure 3.4 for a geometric motivation of the nucleolus: it can be understood as a point that is as far away as possible from the infeasible sides of the bounding hyperplanes.

General algorithms for the computation of the nucleolus have been investigated by several researchers (see, e. g., Potters et al. [92]). Relative to special classes of games, these algorithms do generally not guarantee a polynomially bounded running time. On the other hand, Solymosi and Raghavan [1994] could show that the nucleolus of a matching game can be computed in polynomial time in the bipartite case, where the edges of positive weight in the underlying graph do not contain a circuit of odd length. For the case of general convex games, Kuipers [80] has proposed a polynomial algorithm for computing the nucleolus. On the other hand, Faigle, Kern and Kuipers [49] showed that it is NP-hard to compute the nucleolus for MCST games; they use the construction from our NP-hardness proof for Theorem 3.2.3.

3.4.2 The multiplicative ϵ-core and the nucleon

For the case of TSP games, we explored possibilities of bounding the total deficit. This question was motivated by the way how shares are distributed in the economic

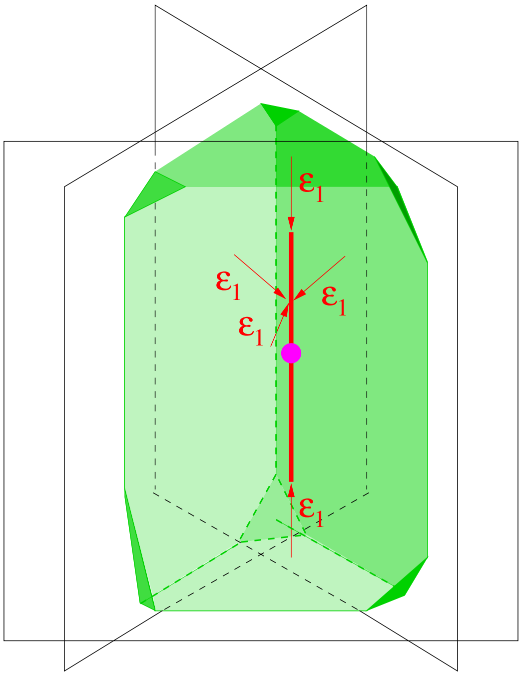


Figure 3.4: Geometric interpretation of the ε-core and the nucleolus

world: Players are charged or rewarded depending on their turnover. This practice is not represented by the nucleolus, where additional charges or rebates are distributed evenly among all players or coalitions, regardless of their amount of business. In the following, we introduce an alternative concept that is more realistic – the *nucleon*.

Faigle and Kern[48] proposed an ϵ-correction relative to the value of a proper coalition and arrive at the modified game (N, v_ϵ), where

$$v_\epsilon(S) = \begin{cases} v(S) & \text{if } S \in \{\emptyset, N\} \\ (1-\epsilon)v(S) & \text{otherwise.} \end{cases}$$

The *multiplicative* ϵ*-core* of (N, v) is the core of the game (N, v_ϵ). We can think of points in the multiplicative ϵ-core as ϵ-approximate allocations.

Similar to the way the nucleolus is defined, we can introduce a multiplicative analogue – the *nucleon*:

Definition 3.4.1 (Faigle, Fekete, Hochstättler, Kern [47]) *Let* (N, c) *be a cooperative savings game and* $x \in \mathbb{R}_+^N$ *an allocation. The* satisfaction ratio *of a coali-*

tion is

$$\alpha(x,S) := \begin{cases} x(S)/v(S) & \textit{if } v(S) > 0 \\ \infty & \textit{if } v(S) = 0. \end{cases}$$

The relative satisfaction vector *is obtained by ordering the* 2^n-2 *satisfaction ratios values* $\alpha(x,S)$ *in a non-decreasing sequence. The* nucleon *of a savings game is the set of allocation vectors that lexicographically maximize the satisfaction ratio vector.*

Just like the nucleolus, we can approach the nucleon by considering a sequence of linear programs to maximize the satisfaction ratios:

Consider the linear program

$$\begin{array}{llllll} & \max & \alpha & & & \\ P(1) & s.t. & x(N) & = & v(N) & \\ & & x(S) - \alpha v(S) & \geq & 0 & \text{for all } S \subset N \end{array}$$

Since $v \geq 0$, this linear program is always feasible. If it has a unique optimum (x^*, α^*) we define $\{x^*\}$ to be the nucleon of the game. Otherwise, let $\alpha^* = \alpha_1$ be the optimum value, so we get $x(S) = \alpha_1 v(S)$ for some coalitions, for all optimal x (which form the "prenucleon"). Define $\mathcal{S}_1$ as the set of these "'tight" subsets of N.

If α_i and $\mathcal{S}_i$ have been computed for $i < j$, then $\mathcal{S}_j$ is given by the tight inequalities in the optimum α_j of the program

$$\begin{array}{llllll} & \max & \alpha & & & \\ P(j) & st. & x(N) & = & v(N) & \\ & & x(S) & = & \alpha_i v(S) & \text{for } i < j \text{ and } S \in \mathcal{S}_i, \\ & & x(S) - \alpha v(S) & \geq & 0 & \text{for all } S \subset N, S \notin \mathcal{S}_i \text{ for } i < j. \end{array}$$

These iterations can be repeated until we get a unique optimal solution or the program becomes unbounded. In the latter case, the nucleon consists of the feasible region of the unbounded program. Note that there can only be finitely many different values of α_i in the relative satisfaction vector; however, if we want to get an efficient algorithmic procedure, we have to argue that this number is polynomial in the number $|N|$ of players.

Intuitively, we can convince ourselves that there can only be a linear number of iterations: Every time we consider a new optimization problem $P(j)$ with a smaller set of feasible solutions, the dimension of the feasible space goes down. Therefore, computing the nucleon can be reduced to solving at most N linear programs, which we will call $Q(1), \ldots, Q(N)$; the $\beta_1, \ldots, \beta_N$ are the corresponding satisfaction ratios α_i.

3.4.3 Matching games

In the following, we consider another type of cooperative game arising from a combinatorial optimization problem: Matching.

Definition 3.4.2 *Let $N = \{1, \ldots, n\}$, consider the complete graph with vertex set N and edges weighted with $w_{ij} \in \mathbb{R}$, $i, j \in N$. For every coalition $S \subseteq N$, define*

$$v(S) = \textit{maximal value of a weighted matching that pairs only vertices in } S \ .$$

A matching of maximal weight never contains a negative edge, so we may assume without loss of generality that the weights w are non-negative. Note that instead of allocating cost, we have to distribute savings.

Matching games were first considered by Shapley and Shubik [98]. Like for TSP games, the core of a matching game may be empty, as illustrated by the situation with three players positioned at the vertices of an isoceles triangle. As we noted in the case of TSP games, nonemptiness of the core is equivalent to the absence of fractional vertices of the corresponding cover polytope – see Faigle and Kern [48] for more details. (Also see the work by Edmonds [40] for a fundamental description of the vertices of the matching polytope.)

For the case of bipartite matching games, the core is always non-empty. Solymósi and Raghavan [101] gave an algorithm for computing the nucleolus in this case. However, it is still unknown how to determine the nucleolus for the general case in polynomial time.

3.4.4 Computing the nucleon of matching games

In the following we will sketch a procedure to compute the nucleon of a matching game which runs in polynomial time.

The basic idea is along the lines described in Subsection 3.4.2; however, each of the linear systems requires exponential space and time when written down explicitly. Similar to the case of optimization over the subtour polytope, a polynomial algorithm is still possible by finding a polynomial separation method and using the ellipsoid method for optimization.

3.4.5 Small coalitions

A key to our approach is to look for small coalitions of one and two players (edges and nodes) which become *tight*: for any feasible x, they satisfy $x(S) = \alpha_S v(S)$ for increasing α. A coalition that is not tight is called *free*.

Let $\mathcal{E}_j$ and $\mathcal{N}_j$ be the set of edges and the set of nodes that become tight at iteration step j. In Proposition 4.1 of [47], we show that a coalition S is tight at step j if and only if S or its complement $N \setminus S$ can be decomposed into tight edges in $\mathcal{E}_l$ for $l \leq j$ and tight vertices in $\mathcal{N}_l$ for $l \leq j$.

This means that a set S can only become tight at step i if it contains either a free edge or a free vertex. Consider such a free edge e and let M be a maximum matching in $S \setminus e$. Then e is allocated an amount of

$$x(e) = \beta_i v(S) - x(M) = \beta_i v(e) + \sum_{f \in M} (\beta_i v(f) - x(f)).$$

With the exception of degenerate cases, this means that S is covered by tight edges except for one edge or one vertex.

More precisely, we get the following:

Let $\mathcal{G}_i$ denote the graph induced by the edges in $\overline{\mathcal{E}_{i-1}} := \bigcup_{j<i} \mathcal{E}_j$. Let c denote the payoffs of edges and vertices which have been fixed in the previous steps. Given $\beta \geq \beta_i$, consider the weight function on $\mathcal{G}_i$

$$w_\beta(f) := \beta v(f) - c(f).$$

For each free edge e let M^e_β denote a matching in $\mathcal{G}_i$ with maximal weight with respect to w_β and set $S^e_\beta := e \cup M^e_\beta$. Similarly, we associate with each free vertex t a unique coalition S^t_β. It is shown in Proposition 4.2 of [47] that the feasible region of the linear program $Q(i)$ from Subsection 3.4.2 is equivalent to the description by the following system:

$$\begin{array}{llcll} Q^*_i: & x(N) & = & v(N) & \\ & x(S) & = & c(S) & \text{for } S \in \mathcal{E}_j \text{ and } j < i, \\ & x(S) & = & c(S) & \text{for } S \in \mathcal{N}_j \text{ and } j < i, \\ & x(S^e_\beta) & \geq & \beta v(S^e_\beta) & \text{for all } S \subset N, e \notin \mathcal{E}_j \text{ for } j < i, \\ & x(S^t_\beta) & \geq & \beta v(S^e_\beta) & \text{for all } S \subset N, t \notin \mathcal{N}_j \text{ for } j < i. \end{array}$$

3.4.6 The algorithm

We have derived a small system which we can solve in polynomial time. However, we still have to deal with the difficulty that the coalitions S^e_β and S^t_β depend on β

and thus on the objective function of the program. This can be dealt with by virtue of Lemma 4.2 of the paper [47]: Since the possible values of β are restricted to an interval, we can use binary search.

In total, we get the following recursive algorithm for computing the nucleolus of a matching game:

1. Compute $\beta_i, \mathcal{E}_i, \mathcal{N}_i$ by solving $Q(i)$ for $i \leq N$.
2. Perform binary search on β to find β_i, such that $Q_i^*(\beta_i) \neq \emptyset$, but $Q_i^*(\beta) = \emptyset$ for all $\beta > \beta_1$.
3. For each β, for all free edges e and vertices t, compute S_β^e, S_β^t by solving weighted matching problems.

Each loop is performed a polynomial number of times. Feasibility of $Q_i^*(\beta)$ can be checked in polynomial time using the ellipsoid method, or an interior point algorithm. For the innermost loop, we have to solve not more than $|E| + |V|$ weighted matching problems. This implies:

Theorem 3.4.3 (Faigle, Fekete, Hochstättler, Kern [47]) *The nucleon P^* of a matching game on the graph $\mathcal{G} = (N, E)$ with edge weights w can be computed in time polynomial in $n = |N|$ and the size $\ll w \gg$ of w.*

Bibliography

[1] H. Aarts. *Minimum Cost Spanning Tree Games and Set Games*. Doctoral thesis, University of Twente, Endschede, 1994.

[2] H. Aarts and T. Driessen. The irreducible core of a minimum spanning tree game. *Zeitschrift für Operations Research*, **38** (1993), pp. 163–174.

[3] R. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows*. Prentice Hall, 1993.

[4] H. Alt, M. Godau, and S. Whitesides. Universal 3-dimensional visibility representations for graphs. *Proc. Graph Drawing 95*, Passau 1996, Lecture Notes in Computer Science # 1027, Springer-Verlag, 1996, pp. 8–19.

[5] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. *Proceedings of the 37th Annual IEEE Symposium on the Foundations of Computer Science (FOCS 1996)*, pp. 2–12.

[6] R. Aumann and M. Maschler. The bargaining set for cooperative games. In: *Advances in Game Theory*, M. Dresher, L. Shapley, A. W. Tucker, eds. Princeton University Press, 1964.

[7] A. Bachem, S. P. Fekete, B. Knab, R. Schrader, I. Vannahme, I. Weber, R. Wegener, K. Weinbrecht, and B. Wichern. Analyse großer Datenmengen und Clusteralgorithmen im Bausparwesen. In *Geld, Finanzwirtschaft, Banken und Versicherungen*. C. Hipp et al. eds., VVW Karlsruhe, 1997.

[8] S. Bhatt and S. Cosmadakis. The complexity of minimizing wire lengths in VLSI layouts. *Information Processing Letters*, **25** (1987), pp. 263–267.

[9] T. Biedl and M. Kaufmann. Area-efficient static and incremental graph drawings. *Algorithms – ESA '97*, Lecture Notes in Computer Science #1284, Springer-Verlag, 1997, pp. 37-52.

[10] T. Biedl, T. Shermer, S. Whitesides, and S. Wismath. Orthogonal 3-D graph drawing. To appear in: *Proc. Graph Drawing '97*, Rome 1997, Springer Lecture Notes in Computer Science, 1998.

[11] C. Bird. On cost allocation for a spanning tree: A game theoretic approach. *Networks*, **6** (1976), pp. 335–350.

[12] G. Birkhoff. Abstract linear dependance in lattices. *Amer. Journal of Mathematics*, **57** (1935), pp. 800–804.

[13] P. Bose, H. Everett, S. Fekete, A. Lubiw, H. Meijer, K. Romanik, T. Shermer, and S. Whitesides. On a visibility representation for graphs in three dimensions. *Proc. Graph Drawing '93*, Paris (Sèvres), 1993, pp. 38–39.

[14] P. Bose, H. Everett, S. Fekete, A. Lubiw, H. Meijer, K. Romanik, T. Shermer, and S. Whitesides. On a visibility representation for graphs in three dimensions, *Snapshots of Computational and Discrete Geometry*, **3** (1994), *eds.* D. Avis and P. Bose, pp. 2–25.

[15] P. Bose, H. Everett, S. P. Fekete, M. E. Houle A. Lubiw, H. Meijer, K. Romanik, G. Rote, T. Shermer, S. Whitesides, and C. Zelle. A visibility representation for graphs in three dimensions. *Technical Report ZPR 97-253*, Center for Parallel Computing, 1997. Available via http://www.zpr.uni-koeln.de.

[16] P. Bose, A. Josefczyk, J. Miller, and J. O'Rourke. K_{42} is a box visibility graph. *Technical Report #034*, Smith College, 1994.

[17] F. J. Brandenburg. Nice drawings of graphs and trees are computationally hard. *Technical Report MIP-8820*, Fakultät für Mathematik und Informatik, Univ. Passau, 1988.

[18] F. J. Brandenburg, ed. *Proc. Graph Drawing 95*, Passau 1996, Lecture Notes in Computer Science #1072, Springer-Verlag 1996.

[19] F.-J. Brandenburg, M. Jünger, and P. Mutzel. Algorithmen zum automatischen Zeichnen von Graphen. *Informatik Spektrum*, **20** (1997), pp. 199–207.

[20] F. R. K. Chung. On unimodal subsequences. *J. Combinatorial Theory*, Series A, **29** (1980), pp. 267–279.

[21] A. Claus and D. Kleitman. Cost allocation for a spanning tree. *Networks*, **3** (1973), pp. 289–304.

[22] F. J. Cobos, J. C. Dana, F. Hurtado, A. Marquez, and F. Mateos. On a visibility representation of graphs. *Proc. Graph Drawing 95*, Passau 1996, Lecture Notes in Computer Science #1072, Springer-Verlag 1996, pp. 152–161.

[23] R. Cohen, P. Eades, T. Lin, and F. Ruskey. Three-dimensional graph drawing. *Proc. Graph Drawing '94*, Princeton NJ, 1994. Lecture Notes in Computer Science #894, Springer-Verlag, 1995, pp. 1–11.

[24] G. Cornuéjols, J. Fonlupt, and D. Naddef. The traveling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming*, **33** (1985), pp. 1–27.

[25] I. J. Curiel, G. Pederzoli, and S. H. Tijs. Sequencing games. *European Journal of Operational Research*, **40** (1989), pp. 344–351.

[26] L. Danzer, B. Grünbaum, and V. Klee. Helly's theorem and its relatives. *Convexity, Proc.* 7^{th} *ACM Symp. Pure Math.*, 1963, pp. 101–181.

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

[28] A. Dean and J. Hutchison. Rectangle visibility representations of bipartite graphs, *Proc. Graph Drawing '94*, Princeton NJ, 1994, Lecture Notes in Computer Science #894, Springer-Verlag, 1995, pp. 159–166.

[29] X. Deng, T. Ibaraki, and H. Nagamochi. Combinatorial optimization games. *Proc.* 8^{th} *Symp. Disc. Algorithms (SODA 1997)*, pp. 720–729.

[30] J. Derks and J. Kuipers. On the core of routing games. To appear in: *International Journal of Game Theory*.

[31] G. Di Battista, ed. *Proc. Graph Drawing 97*, Rome 1997, to appear in Lecture Notes in Computer Science, Springer-Verlag 1998.

[32] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for automatic graph drawing: an annotated bibliography. *Computational Geometry: Theory and Applications*, **4** (1994), pp. 235–282. Also available from wilma.cs.brown.edu by ftp.

[33] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing. Algorithms for geometric representation of graphs*. Prentice Hall, to appear.

[34] G. Di Battista, G. Liotta, and S. Whitesides. The strength of weak proximity. *Proc. Graph Drawing 95*, Passau 1996. Lecture Notes in Computer Science # 1072, Springer-Verlag 1996.

[35] G. Di Battista and R. Tamassia. Algorithms for plane representations of acyclic digraphs. *Theoretical Computer Science*, **61** (1988), pp. 175–198.

[36] P. Eades. A heuristic for drawing graphs. *Congressus numerantium*, **42** (1984), pp. 149–160.

[37] P. Eades, A. Symvonis, and S. Whitesides. Two algorithms for three dimensional graph drawing. *Proc. Graph Drawing 96*, Berkeley 1996, Lecture Notes in Computer Science #1197, Springer-Verlag 1996, pp. 139–154.

[38] P. Eades and S. Whitesides. The realization problem for Euclidean minimum spanning trees is NP-hard. In: *Proc.* 10^{th} *ACM Symp. on Computational Geometry*, 1994, pp. 49–56.

[39] P. Eades and S. Whitesides. The logic engine and the realization problem for nearest neighbor graphs. *Theoretical Computer Science*, **169** (1) (1996), pp. 23–37.

[40] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards (B)*, **71** (1967), pp. 233–240.

[41] H. ElGindy, G. Liotta, A. Lubiw, H. Meijer, and S. Whitesides. Recognizing rectangle of influence drawable graphs. *Proc. Graph Drawing '94*, Princeton NJ, 1994, Lecture Notes in Computer Science LNCS #894, Springer-Verlag, 1995 pp. 352–363.

[42] L. Euler. Solutio problematis ad geometriam situs pertinentis. *Comment. Academiae Sci. I. Petropolitanae*, **8** (1736), pp. 128–140.

[43] L. Euler. Elementa doctrinae solidorum. *Novi Comment. Academiae Sci. Petropolitanae*, **4** (1758), pp. 109–140.

[44] L. Euler. Demonstratio nonullarum insignium proprietarum, quibus solida hedris planis inclusa sunt praedita. *Novi Comment. Academiae Sci. Petropolitanae*, **4** (1758), pp. 140–160.

[45] U. Faigle, S. P. Fekete, W. Hochstättler, and W. Kern. On approximately fair cost allocation in Euclidean TSP games. To appear in: *OR-Spektrum*, 1998. Available via http://www.zpr.uni-koeln.de.

[46] U. Faigle, S. P. Fekete, W. Hochstättler, and W. Kern. On the complexity of testing membership in the core of min-cost spanning tree games. *International Journal of Game Theory*, **26** (1997), pp. 361–366.

[47] U. Faigle, S. P. Fekete, W. Hochstättler, and W. Kern. The nucleon of cooperative games and an algorithm for matching games. To appear in: *Mathematical Programming*, 1998. Available via http://www.zpr.uni-koeln.de.

[48] U. Faigle and W. Kern. On some approximately balanced combinatorial cooperative games. *ZOR – Methods and Models of Operations Research*, **38** (1993), pp. 141–152.

[49] U. Faigle, W. Kern, and J. Kuipers. Computing the nucleolus of min-cost spanning tree games is NP-hard. Manuscript, 1997.

[50] G. Farkas. Theorie der einfachen Ungleichungen. *J. Reine Angew. Math.*, **124** (1902), pp. 1–27.

[51] S. P. Fekete, M. E. Houle, and S. Whitesides. New results on a visibility representation of graphs in 3D, *Proc. Graph Drawing 95*, Passau 1996, Lecture Notes in Computer Science #1072, Springer-Verlag 1996, pp. 234–241.

[52] S. P. Fekete, M. E. Houle, and S. Whitesides. The wobbly logic engine: proving hardness of non-rigid graph representation problems. To appear in: *Proc. Graph Drawing '97*, Rome 1997, Springer Lecture Notes in Computer Science, 1998. Available via http://www.zpr.uni-koeln.de.

[53] S. P. Fekete and H. Meijer. Rectangle and box visibility graphs in 3D. To appear in *International Journal of Computational Geometry and Applications*, 1998. Available via http://www.zpr.uni-koeln.de.

[54] S. P. Fekete and W. R. Pulleyblank. Traveling preacher games. In preparation.

[55] V. Feltkamp, S. Tijs, and S. Muto. Minimum cost spanning extension problems: the proportional rule and the decentralized rule. *Electronic Working Papers in Economics*. Available at http://netec.mcc.ac.uk/ adnetec/WoPEc/kubrce/kubrce9496.html

[56] U. Fößmeier and M. Kaufmann. Drawing high degree graphs with low bend numbers. *Proc. Graph Drawing 95*, Passau 1996, Lecture Notes in Computer Science # 1027, Springer-Verlag, 1996, pp. 254–266.

[57] U. Fößmeier and M. Kaufmann. Algorithms and area bounds for nonplanar orthogonal drawings. To appear in: *Proc. Graph Drawing '97*, Rome 1997, Springer Lecture Notes in Computer Science, 1998.

[58] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, 1979.

[59] M. X. Goemans. Worst-case comparison of valid inequalities for the TSP. *Mathematical Programming*, **69** (1995), pp. 335–349.

[60] D. Granot, and G. Hubermann. Minimum cost spanning tree games. *Mathematical Programming*, **21** (1981), pp. 1–18.

[61] D. Granot, and G. Hubermann. On the core and nucleolus of minimum cost spanning tree games. *Mathematical Programming*, **29** (1984), pp. 323–347.

[62] M. Grötschel, L. Lovász, L., and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin, 1988.

[63] P. Hall. On representatives of subsets, *J. London Math. Soc.*, **10** (1935), pp. 26–30.

[64] J. M. Hammersley. A few seedlings of research. *Proc.* 6^{th} *Berkeley Symp. Math.* Stat. Prob., U. of California Press, 1972, pp. 345–394.

[65] W. Hochstättler. Duality in combinatorial optimization – sometimes it works, sometimes it won't. Habilitationsschrift, Universität zu Köln, 1997. Available via http://www.zpr.uni-koeln.de.

[66] J. Hutchinson, T. Shermer, and A. Vince. On representations of some thickness-two graphs, *Proc. Graph Drawing 95*, Passau 1996, Lecture Notes in Computer Science #1072, Springer-Verlag 1996, pp. 324–332.

[67] P. J. Idicula. *Drawing trees in grids*. Master's thesis, Department of Computer Science, University of Auckland, 1990.

[68] D. S. Johnson, L. A. McGeogh, and E. E. Rothberg. Asymptotic experimental analysis for the Held-Karp Traveling Salesman bound. *Proc.* 7^{th} *Symp. Disc. Algorithms (SODA 1996)*.

[69] M. Jünger, E. Lee, P. Mutzel, and T. Odenthal. A polyhedral approach to the multi-layer crossing number problem. To appear in: *Proc. Graph Drawing '97*, Rome 1997, Springer Lecture Notes in Computer Science, 1998.

[70] M. Jünger and P. Mutzel. Maximum planar subgraphs and nice embeddings: practical layout tools. *Algorithmica*, **16** (1996), pp. 33–59.

[71] M. Jünger and P. Mutzel. Solving the maximum weight planar subgraph problem by branch and cut. *Proc. IPCO 93*, pp. 479–492.

[72] M. Jünger and P. Mutzel. The polyhedral approach to the maximum planar subgraph problem: new chances for related problems. *Proc. Graph Drawing 94*, Rutgers 1994, Lecture Notes in Computer Science #894, Springer-Verlag 1995, pp. 119-130.

[73] M. Jünger and W. R. Pulleyblank. Geometric duality and combinatorial optimization. In: *Jahrbuch Überblicke Mathematik*. Vieweg, New York, 1993, pp. 1–24.

[74] G. Kant, G. Liotta, R. Tamassia, and I. G. Tollis. Area requirements of visibility representations of trees. *Proceedings of the 5th Canadian Conf. on Comp. Geom.*, Waterloo, Ontario, 1993, pp. 192–197.

[75] K. Kent. *Stable cost allocations on minimum spanning tree networks*. Ph. D. thesis. Department of Mathematics and Statistics, SUNY Stony Brook, 1997.

[76] W. Kern. Private communication, 1996.

[77] H. Koike. An application of three-dimensional visualization to object-oriented programming, *Proc. of Advanced Visual Interfaces AVI '92*, Rome, May 1992, v. 36 of World Scientific Series in Computer Science, 1992, pp. 180–192.

[78] E. Kranakis, D. Krizanc, and J. Urrutia. On the number of directions in visibility representations of graphs, *Proc. Graph Drawing '94*, Princeton NJ, 1994, Lecture Notes in Computer Science #894, Springer-Verlag, 1995, pp. 167–176.

[79] J. Kuipers. A note on the 5-person traveling salesman game. *Tech. rep.*, Univ. of Limburg, The Netherlands, 1991.

[80] J. Kuipers. A polynomial algorithm for computing the nucleolus of convex games. *Tech. rep.*, Univ. of Maastricht, The Netherlands, 1996.

[81] K. Kuratowski. Sur le problème des courbes gauches en topologie. *Fund. Math.*, **15** (1930), pp. 271–283.

[82] E. Lawler, J. Lenstra, A. Rinnooy Kan, and D. Shmoys, eds. *The Traveling Salesman Problem*. John Wiley & Sons, New York, 1985.

[83] J. Mackinley, G. Robertson, and S. Card. Cone trees: animated 3d visualizations of hierarchical information, *Proc. of the SIGCHI Conf. on Human Factors in Computing*, 1991, pp. 189–194.

[84] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: Part II – A simple polynomial-time approximation scheme for geometric k-MST, TSP, and related problems. To appear in: *SIAM Journal of Computing*. Available at http://ams.sunysb.edu/~jsbm/jsbm.html.

[85] P. Mutzel. The maximum planar subgraph problem. Dissertation, Institut für Informatik, Universität zu Köln, 1994.

[86] G. L. Nemhauser, L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, New York, 1988, p. 530.

[87] S. North, ed. *Proc. Graph Drawing 96*, Berkeley 1996, Lecture Notes in Computer Science #1197, Springer-Verlag 1996.

[88] G. Owen. On the core of linear production games. *Mathematical Programming*, **9** (1975), pp. 358–370.

[89] M. Padberg and S. Hong. On the symmetric traveling salesman problem: a combinatorial study. *Mathematical Programming Study*, **12** (1980), pp. 78–107.

[90] C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice Hall, 1982.

[91] J. Potters and I. Curiel. Traveling salesman games. *Mathematical Programming*, **53** (1992), pp. 199–211.

[92] J. A. M. Potters, J. H. Reijnierse, and M. Ansing. Computing the nucleolus by solving a prolonged simplex algorithm. *Mathematics of Operation Research*, **21**, pp. 757–768.

[93] N. Robertson and P. D. Seymour. Graph Minors. VIII. A Kuratowski theorem for general surfaces. *Journal of Combinatorial Theory, Series B*, **48** (1990), pp. 255–288.

[94] K. Romanik. Directed VR-representable graphs have unbounded dimension, *Proc. Graph Drawing '94*, Princeton, NJ, 1994, Lecture Notes in Computer Science #894, Springer-Verlag, 1995, pp. 177–181.

[95] G. Rote and C. Zelle. K_{22} has a visibility representation by rectangles. Private communication, 1996.

[96] D. Schmeidler. The nucleolus of a characteristic function game. *SIAM Journal of Applied Mathematics*, **17** (1969), pp. 1163–1170.

[97] L. Shapley. A value for n person games. In: *Contributions to the Theory of Games II*, H, Kuhn and A. W. Tucker, eds. Princeton University Press, 1953.

[98] L. Shapley and M. Shubik. Quasi-cores in a monetary economy with non-convex preferences. *Econometrica*, **34** (1966), pp. 805–827.

[99] D. Shmoys and D. Williamson. Analyzing the Held-Karp TSP bound: a monotonicity property with applications. *Information Processing Letters*, **35** (1990), pp. 281–285.

[100] M. Shubik. Game theory models and methods in political economy. In: *Handbook of Mathematical Economics*, vol. 1. North-Holland, New York, 1981, pp. 285–330.

[101] T. Solymosi and T. Raghavan. An algorithm for finding the nucleolus of assignment games. *International Journal of Game Theory*, **23** (1994), pp. 119–143.

[102] T. Solymosi, H. Aarts, and T. Driessen. On computing the nucleolus of a balanced connected game. To appear in: *Mathematics of Operations Research*.

[103] R. Tamassia and I. G. Tollis. A unified approach to visibility representations of planar graphs, *Discrete Computational Geometry*, **1**, 1986, pp. 321–341.

[104] R. Tamassia, I. Tollis, eds. *Proc. Graph Drawing 94*, Rutgers 1994, Lecture Notes in Computer Science #894, Springer-Verlag 1995.

[105] A. Tamir. On the core of a traveling salesman cost allocation game. *OR Letters*, **9** (1989), pp. 31–34.

[106] A. Tamir. On the core of network synthesis games. *Mathematical Programming*, **50** (1991), pp. 123–135.

[107] S. Tijs and T. Driessen. Extensions of solution concepts by means of multiplicative ϵ-tax games. *Mathematical Social Sciences*, **12** (1986), pp. 9–20.

[108] W. R. Tutte. How to draw a planar graph. *Proc. London Soc. Math.*, **13** (1963), pp. 743–786.

[109] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behaviour*, 3rd ed. Princeton University Press, 1955.

[110] S. K. Wismath. Characterizing bar line-of-sight graphs, *Proc. ACM Symp. on Computational Geometry*, 1985, pp. 147–152.

[111] L. Wolsey. Heuristic analysis, linear programming and branch and bound. *Mathematical Programming Study*, **13** (1980), pp. 121–134.

[112] M. Wottawa. Struktur und algorithmische Behandlung von praxisorientierten dreidimensionalen Packungsproblemen, Doctoral Thesis, Mathematisches Institut, Universität zu Köln, 1996.