Geometric and Combinatorial Problems of Matching and Partitioning in Theoretical Computer Science

Dissertation zur Erlangung des Doktorgrades

vorgelegt am

Fachbereich Mathematik und Informatik
der Freien Universität Berlin

Februar 2015

von

Tillmann Miltzow

Institut für Informatik
Freie Universität Berlin
Takustraße 9
14195 Berlin
t.m@fu-berlin.de

1. Gutachter : Prof. Dr. Günter Rote

2. Gutachter : Prof. Dr. Stefan Felsner

Datum der Disputation: 5.Juni.2015

0.1 Zusammenfassung

In dieser Arbeit betrachten wir vier Probleme der Theoretischen Informatik:

1. Disjunkte Einheitsscheiben in der Ebene und disjunkte Einheitskugeln im Raum lassen sich durch Hyberebenen separieren. Dabei wird versucht die Anzahl der Schnitte von der Ebene mit den Objekten zu minimieren. Obwohl erste Arbeiten dazu in den 80^{er} Jahren des letzten Jahrhunderts entstanden, gab es bis dato noch keinen optimalen deterministischen Algorithmus um solch eine Hyperebene zu finden. Wir stellen einen exakten Algorithmus in der Ebene und einen approximativen Algorithmus in höheren Dimensionen vor. (Dieser Teil entstand gemeinsam mit Michael Hoffman und Vincent Kusters.)

2. Tron ist ein Computerspiel aus den 80^{er} Jahren, das zunächst von Bodlaender und Kloks auf abstrakten Graphen studiert wurde. Wir beantworten offen geblieben Fragen zur algorithmischen Komplexität und untersuchen das maximal und minimale Punktunteverhältnis der beiden Spieler bei rationaler Spielweise. (Beide Spieler erzielen im Laufe des Spieles Punkte.) Wir betrachten diese Fragen in verschiedenen Spielmodi.

3. Pareto Optimale Matchings kommen aus der Ökonomie und Spieletheorie und beschreiben bestimmte "stabile" Situationen ähnlich einem Nash-Equilibrium. Sie spielen auch in algorithmischen Problemen eine Rolle. Wir geben eine obere Schranke für die Anzahl für Pareto Optimale Matchings unter einfachen Nebenbedingungen an. Desweiteren lösen wir eine Reihe verwandter algorithmischer Probleme. (Dieser Teil entstand gemeinsam mit Balázs Keszegh und Andrei Asinowski.)

4. Geometrische Matchings sind kreuzungsfreie Strecken, die eine Menge von Punkten in der Ebene verbinden. Obwohl es sehr einfach ist gefärbten Punktmengen mit nur einem einzigen geometrischen Matching zu finden, gelang es erst jetzt solche Punktmengen allgemein zu charakterisieren. Weitere Fragen zu dieser Klasse von Punktmengen wurden beantwortet. (Dieser Teil entstand gemeinsam mit Andrei Asinowski und Günter Rote.)

Contents

Chapter 1

Introduction

In this chapter, we will describe briefly the results of the forthcoming chapters and we will draw some connections between them. More detailed introductions will be found at the beginning of each chapter.

The work about Tron can be found online [55] and a preliminary version appeared at FUN 2012 [56]. The work about Unique bichromatic matchings can be found online [11] and is accepted for publication in the JOCG [12]. The work on Pareto optimal matchings can be found online [9] and a preliminary version appeared at FUN 2014 [10]. The work about Separating balls by hyperplanes can be found online [38] and a preliminary version appeared at ESA 2014 [39].

The results presented here are joint work with my co-authors. Most lemmas and theorems are a product of collaboration and cannot be credited by only one of the authors. Therefore, I omit the specification of credits.

1.1 Halving Balls in Deterministic Linear Time

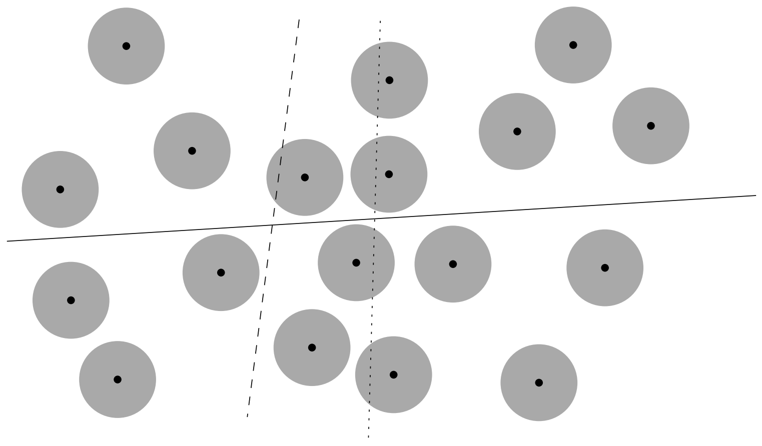


Figure 1.1: A set of 18 disks in $\mathbb{R}^2$ and three separators. The dashed line forms a 6-separator. Both the solid line and the dotted line are halving lines. The solid line is preferable to the other two lines because it separates exactly into half and half and intersects no disks.

This chapter originated during a research visit in Zürich with Michael Hoffmann and Vincent Kusters.

Chapter 2 is about separating balls by hyperplanes. Let $\mathcal{D}$ be a set of n disjoint unit balls in $\mathbb{R}^d$ and let P be the set of their center points. A hyperplane $\mathcal{H}$ is an *m-separator* for $\mathcal{D}$ if each closed halfspace bounded by $\mathcal{H}$ contains at least m points from P. This generalizes the notion of halving hyperplanes, which correspond to $n/2$-separators. The analogous notion for point sets has been well-studied. Separators have various applications, for instance, in divide-and-conquer schemes. In such a scheme any ball that intersects the separating hyperplane may still interact with both sides of the partition. Therefore it is desirable that the separating hyperplane intersects a small number of balls only.

We will present an approximation-scheme that, with the right parameters, constructs an αn-separator for balls in $\mathbb{R}^d$, for any $0 < \alpha < 1/2$, which intersects at most $cn^{(d-1)/d}$ balls, for some constant c that depends on d and α. This upper bound is asymptotically optimal.

Next, we give a linear-time algorithm to construct a halving line in $\mathbb{R}^2$ that intersects $O(n^\delta)$ disks, with δ arbitrarily close to 2/3.

1.2 TRON

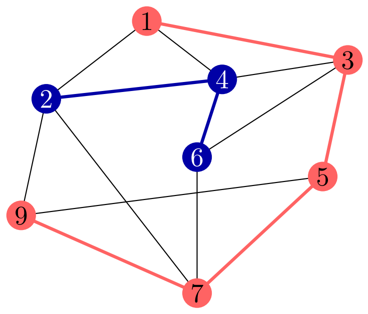


Figure 1.2: ALICE achieves a score of 5 and BOB achieves a score of 3.

Chapter 3 analyzes some aspects of the game, called TRON. TRON is a two player game played on an abstract simple graph. It is motivated by the movie with the same name and was first studied before by Bodlaender [19]. During the game, the players take turns alternately. Each turn consists of visiting a vertex adjacent to the vertex visited last by this player. A vertex that was already visited cannot be revisited ever again. Thus each player forms a path(two paths in total). The game ends when both players cannot move. The player with the longer path wins. We consider eight different game modi: start vertices are given or not, the graph is directed or undirected and whether we consider normal play or misère play. In misère play, the player with the shorter path wins. The first part of the chapter deals with the question, by how much one player can be better than the other. It turns out that for all game modi either player can get all vertices except constantly many. The second part considers the algorithmic complexity. We show PSPACE-completeness for all game modi. In particular Bodlaender and Kloks conjectured PSPACE-completeness for normal play on an undirected graph without given start vertices [20]. We prove this conjecture.

1.3 Pareto Optimal Matchings

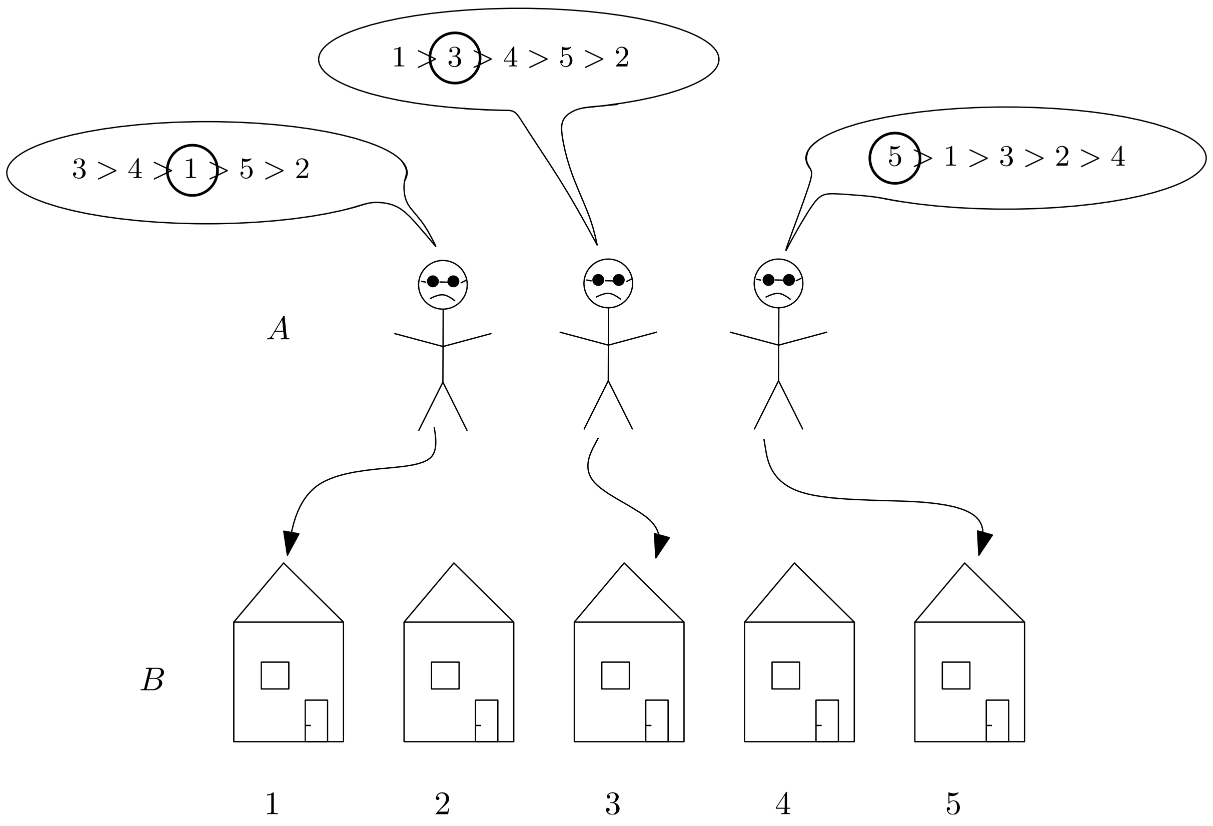


Figure 1.3: Here Set A consists of 3 people and set B of 5 houses. The preference lists of each person is indicated and a matching is indicated by circles and arrows. However it is not Pareto Optimal. Person 1 and 2 form a blocking coalition.

This is joined work with Andrei Asinowski and Balázs Keszegh.

Chapter 4 deals with combinatorial and algorithmic aspects of Pareto Optimal matchings in the house allocation problem.

In an instance of the house allocation problem, two sets A and B are given. The set A is referred to as *applicants* and the set B is referred to as *houses*. We denote by m and n the size of A and B respectively. In the house allocation problem, we assume that every *applicant* $a \in A$ has a preference list over every *house* $b \in B$. We call an injective mapping τ from A to B a matching. A *blocking coalition* of τ is a subset A' of A such that there exists a matching τ' that differs from τ only on elements of A', and every element of A' improves in τ', compared to τ according to its preference list. If there exists no blocking coalition, we call the matching τ an *Pareto optimal matching* (POM). A house $b \in B$ is *reachable* if there exists a Pareto optimal matching using b. The set of all reachable houses is denoted by E^*. We show

$$|E^*| \leq \sum_{i=1,\dots,m} \left\lfloor \frac{m}{i} \right\rfloor = \Theta(m \log m).$$

This is asymptotically tight. A set $E \subseteq B$ is *reachable* (respectively *exactly reachable*) if there exists a Pareto optimal matching τ whose image contains E as a subset (respectively equals E). We give bounds for the number of exactly reachable sets. Further, we give complexity results and algorithms for corresponding algorithmic questions. Finally, we characterize *unavoidable* houses, i.e., houses that are used by all POM's. This yields efficient algorithms to determine all unavoidable elements.

1.4 Unique Bichromatic Matchings

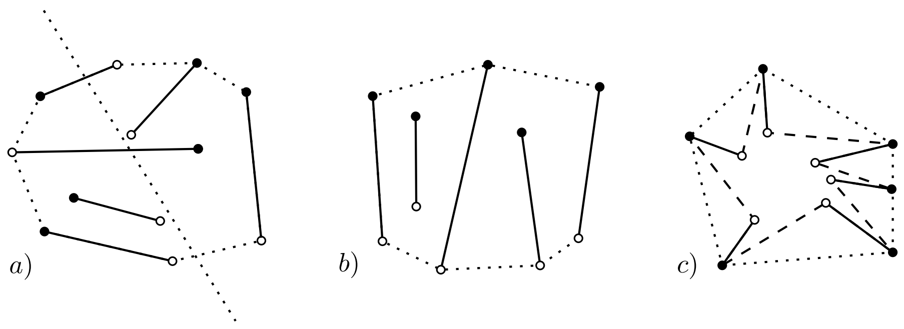


Figure 1.4: (a) An non-unique matching . (b) A linear matching. (c) A circular matching (another matching for the same point set is indicated by dashed lines).

This is joined work with Andrei Asinowski and Günter Rote.

Chapter 5 characterizes bichromatic point sets in the plane with exactly one geometric matching.

Given a set of n red and n blue points in general position in the plane, it is well-known that there is at least one bichromatic perfect matching realized by non-crossing line segments. We characterize such point sets that have *exactly one* matching of this kind. We find several geometric descriptions of such sets, and give an $O(n \log n)$ algorithm that checks whether a given bichromatic set has this property.

Given a perfect bichromatic matching M, a *chromatic-cut* is a line ℓ intersecting at least two segments A and B of M in such a way that the endpoints of A and B on one side of ℓ have different colors, see Figure 1.4 a). We will show that the existence of a chromatic-cut implies the existence of another matching M' and thus M was not unique. Matchings without a chromatic cut fall into two categories. Either they are of linear or circular type. We will characterize both of them and conclude that exactly the matchings of linear type are unique. See Figure 1.4 b) for a matching of linear type and Figure 1.4 c) for a matching of circular type.

Chapter 2

Halving Balls

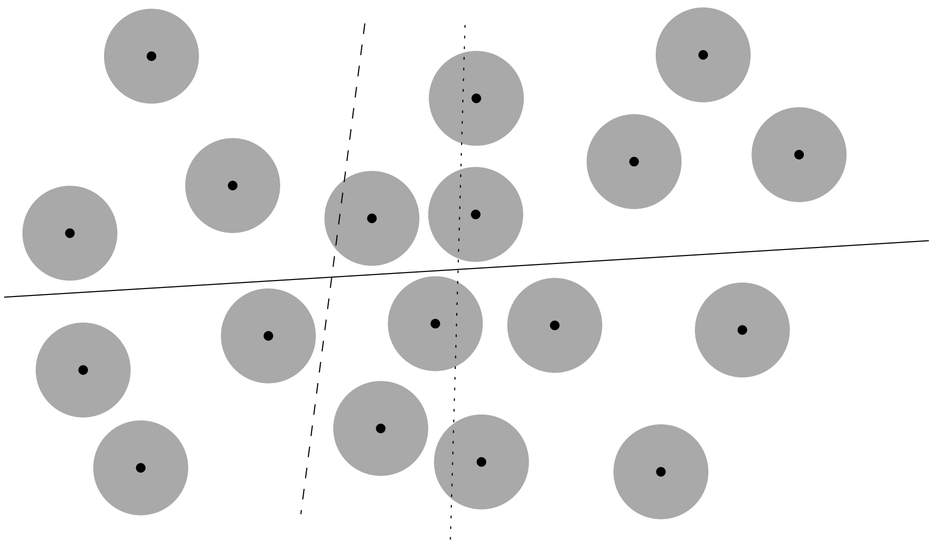


Figure 2.1: A set of 18 disks and three separators. The dashed line forms a 6-separator. Both the solid line and the dotted line are halving lines. The solid line is preferable to the other two lines because it separates perfectly and does not intersect any disk.

2.1 Introduction

Let $\mathcal{D}$ be a set of n pairwise disjoint unit balls in $\mathbb{R}^d$ and P the set of their midpoints. A hyperplane $\mathcal{H}$ is an *m-separator* for $\mathcal{D}$ if each closed halfspace bounded by $\mathcal{H}$ contains at least m points from P. This generalizes the notion of halving hyperplanes, which correspond to $\lfloor n/2 \rfloor$-separators. The analogous notion of separating hyperplanes for point sets has been well studied (see, e.g, [52] for a survey). Separators have various applications, for instance in divide-and-conquer schemes (we discuss some explicit examples below). In such a scheme any ball that is intersected by the separating hyperplane may still interact with both sides of the partition. Therefore it is desirable that the separating hyperplane intersects a small number of balls only.

Alon, Katchalski and Pulleyblank [6] prove that for any set $\mathcal{D}$ in $\mathbb{R}^2$, there exists a direction such that every line with this direction intersects $O(\sqrt{n \log n})$ disks. In particular, this guarantees the existence of a halving line that intersects at most $O(\sqrt{n \log n})$ disks.

Löffler and Mulzer [50] observed that this proof gives a randomized linear-time algorithm to find such a halving line. In this chapter, we present the following three deterministic algorithms, each of which computes an m-separator that intersects $o(n)$ balls for various m.

We develop a generic algorithm in $\mathbb{R}^d$ that can be instantiated with different parameters to obtain Theorem 2.1 and Theorem 2.2. Theorem 2.1 is the most interesting case and Theorem 2.2 is an example for a different choice of parameters.

Theorem 2.1 simply tries, in a certain manner, a constant number of directions for an m-separator. For each direction a linear amount of time is spent.

Theorem 2.1. *Given a set $\mathcal{D}$ of n pairwise disjoint unit balls in $\mathbb{R}^d$ and $\alpha \in (0, 1/2)$, one can construct in $O(n/(1-2\alpha))$ time a hyperplane $\mathcal{H}$ that intersects $O((n/(1-2\alpha))^{(d-1)/d})$ balls from $\mathcal{D}$ such that each closed halfspace bounded by $\mathcal{H}$ contains at least αn centers of balls from $\mathcal{D}$. The constants hidden by the asymptotic notation depend on d only.*

Theorem 2.2 uses a bit more than constantly many directions and thus also a bit more than linear time. On both halfspaces defined by the hyperplane are $n/2 - o(n)$ balls completely contained and thus only a sublinear number of balls are intersected.

Theorem 2.2. *Given a set $\mathcal{D}$ of n pairwise disjoint unit balls in $\mathbb{R}^d$ and a function $f(n) \in \omega(1) \cap o\left(n^{1/(2d-1)}\right)$, one can construct in $O(nf(n))$ time a hyperplane $\mathcal{H}$ such that each closed halfspace bounded by $\mathcal{H}$ contains at least $\frac{n}{2} - O\left(\frac{n}{f(n)}\right) = \frac{n}{2} - o(n)$ balls from $\mathcal{D}$ entirely.*

Note that the statement would become trivial, if we would not require that the balls are completely contained, but it would suffice, if they are partially contained. Note that Theorem 2.2 improves the separation of the midpoints (compared to Theorem 2.1) at the cost of increasing the running time slightly. See Theorem 2.4 for full generality. We don't state it in this introduction, as the full generality seems to be less pleasant to read and it adds little value. The dependence on the dimension d is unusual and we refrain from discussing it in this introduction.

Theorem 2.3 computes a true halving line in the plane. This algorithm uses more insights and sophisticated techniques.

Theorem 2.3. *Let $\varepsilon > 0$. For any set $\mathcal{D}$ of n pairwise disjoint unit disks in $\mathbb{R}^2$ one can construct in $O(n)$ time a line ℓ that intersects $O(n^{2/3+\varepsilon})$ disks from $\mathcal{D}$ such that each closed halfplane bounded by ℓ contains at least $n/2$ centers of disks from $\mathcal{D}$.*

Related work. Bereg, Dumitrescu and Pach [16] (see also [58, Lemma 9.3.2]) strengthen the initial result of Alon, Katchalski and Pulleyblank slightly by proving that there exists a direction such that any line with this direction has at most $O(\sqrt{n \log n})$ disks *within constant distance*. They use this lemma to prove that one can always move a set of n unit disks from a start to a target configuration in $3n/2 + O(\sqrt{n \log n})$ moves. In order to compute these moves they compute in a brute force manner *one* halving line as above with few intersections. Their algorithm runs in $O(n^{3/2}(\log n)^{-1/2})$ time. Theorem 2.3 improves this to $O(n \log n)$ time.

Held and Mitchell [36] introduced a paradigm for modeling data imprecision where the location of a point in the plane is not known exactly. For each point, we are given a unit disk (disjoint from the other disks) that is guaranteed to contain the point. The authors show that after preprocessing the disjoint unit disks in $O(n \log n)$ time, they can construct a triangulation of the actual point set in linear time. Löffler and Mulzer [50] follow the same model to construct the onion layer of an imprecise point set. They observed that the proof by Alon et al. immediately gives a randomized expected linear-time algorithm in the following fashion. Pick an angle $\beta \in [0, \pi]$ uniformly at random and compute a halving line for the disks with slope β. This halving line intersects at most $O(\sqrt{n \log n})$ unit disks with probability at least 1/2. Löffler and Mulzer use this algorithm to compute a (α, β)-*space decomposition tree*: a data structure similar to a binary space partition in which every line is an αk-separator that intersects at most k^β disks. They show that such a $(1/2 + \varepsilon, 1/2 + \varepsilon)$ space decomposition tree can be computed in randomized and deterministic $O(n \log n)$ time, for every $\varepsilon > 0$.

In order to compute this space decomposition in a deterministic way, they present two algorithms. First, they present a simple deterministic linear-time algorithm that guarantees that at least $n/10$ of the disks are completely on each side of some axis-parallel line. This algorithm inspired Theorem 2.1. The second algorithm is a more sophisticated, deterministic $O(n \log n)$ time algorithm to compute a line ℓ such that there are at least $n/2 - cn^{5/6}$ disks completely to each side of ℓ. The algorithm uses an r-partition of the plane [53] to find good candidate lines. Our algorithms improve the running time of those algorithms and thus also simplify the argument that a space decomposition can be found in $O(n \log n)$ time.

Tverberg [67] studies a related question. He proves that for every natural number k there is a number $K(k)$, such that given convex pairwise disjoint sets $C_1, \ldots, C_{K(k)}$, there always exists a line with some set completely on one side and k sets completely on the other side.

Let us also mention that geometric separation theorems can be used to find graph separators for certain graphs with geometric representation [54].

Organization. We develop a generic algorithm to compute a separator in $\mathbb{R}^d$ (where the trade-off between the number of intersected disks and the number of disk centers on each side is determined by a parameter) and prove Theorem 2.1 and Theorem 2.2 in Section 2.2. We prove Theorem 2.3 in Section 2.3. Our algorithm follows the approach used in the linear-time ham-sandwich cut algorithm [49]. It divides the line arrangement dual to the set of disk center points by vertical lines such that each slab (the region bounded by two consecutive vertical lines) contains at most a constant fraction of the vertices of the arrangement. In each iteration, the algorithm chooses a slab and discards a linear fraction of the lines.

2.2 Separating balls in higher dimensions

In this section, we develop a generic algorithm to compute a separator for a given set of pairwise disjoint unit balls in $\mathbb{R}^d$. Using this generic algorithm, we will give two algorithms to compute an approximately halving hyperplane that intersects a sublinear number of balls.

Besides the set $\mathcal{D}$ of n balls in $\mathbb{R}^d$, the generic algorithm has two more parameters. First, a number $b \in \{1, \ldots, n\}$ that quantifies the quality of the approximation: we will show that the hyperplane constructed by the algorithm forms an $(n - b)/2$-separator for $\mathcal{D}$. The main step of the algorithm consists in finding a direction $\vec{v}$ such that we are guaranteed to find a desired separator that is orthogonal to $\vec{v}$. A second parameter $k \in \mathbb{N}$ of the algorithm specifies the number of different directions to generate and test during this step.

As a rule of thumb, generating more directions results in a better separation, because b can be chosen smaller, but the runtime of the algorithm increases proportionally. The algorithm works for certain combinations of these parameters only, as detailed in the following theorem.

Maybe one way to read the following theorem is as follows. The user specifies k (the number of directions to be tested) and thereafter chooses b as small as possible without violating Condition (2.1).

Another perspective is to choose first b and then k. Here k must be chosen large enough such that condition (2.1) is satisfied. However, the proof of the theorem yields that the number of intersections would increase with an increase of k. This is counterintuitive and only an artifact of the proof technique.

Theorem 2.4. *Given a set $\mathcal{D}$ of n pairwise disjoint unit balls in $\mathbb{R}^d$ and parameters $b \in \{1, \ldots, n\}$ and a prime k that satisfy the conditions*

$$dn \leq kb \quad \text{and} \tag{2.1}$$

$$t := \left(\frac{V_d}{2d^{(d-2)/2}}\right)^{1/d} \frac{n^{1/d}}{k^{2-1/d}} > 2, \tag{2.2}$$

where V_d is the volume of the d-dimensional unit ball, one can construct in $O(kn)$ time a hyperplane $\mathcal{H}$ that intersects at most $2b/(t-2)$ balls from $\mathcal{D}$ such that each closed halfspace bounded by $\mathcal{H}$ contains at least $(n-b)/2$ centers of balls from $\mathcal{D}$.

Assuming k to be prime is not a restriction: If k is not prime, then there is a prime $k' \leq k + O(k^a)$, for $a = 4/5$, see [42]. We can compute k' efficiently, for instance, in $O(k)$ time using the Sieve of Eratosthenes.

To see the running time upper bound, first note that, we have only to consider primes in the range $U = \{1, \ldots, \lfloor\sqrt{2k}\rfloor\}$. And we search for primes in the range $W = \{k, \ldots, k + c \cdot k^a - 1\}$, for some fixed c. The computation costs $T(k)$ are upper bounded by

$$T(k) \leq \sum_{p \in U} \frac{|W|}{p} \leq \sum_{i=2,\ldots,\lfloor\sqrt{2k}\rfloor} \frac{c \cdot k^a}{i} = O(k^a \log k) = O(k),$$

where we used the harmonic series in the penultimate step.

In practice it is sufficient to store small tables of quickly growing primes [1].

Perhaps more interesting than Theorem 2.4 in its full generality are the special cases stated as Theorem 2.1 and Theorem 2.2 above. We will now prove Theorem 2.1 and Theorem 2.2 from the general theorem.

Proof of Theorem 2.1. Assume $\alpha \in (0, 1/2)$ is given. We define $b = \lfloor(1-2\alpha)n\rfloor$ and k a prime with $k \in [\frac{2d}{1-2\alpha}, \frac{4d}{1-2\alpha}]$. We will show first that Condition (2.1) and (2.2) hold for these choices. The first condition holds for large enough n, which can be seen as follows.

$$kb \geq \lfloor(1-2\alpha)n\rfloor \frac{2d}{1-2\alpha} \geq \frac{((1-2\alpha)n-1)2d}{1-2\alpha} = 2nd - \frac{2d}{1-2\alpha} \geq nd.$$

[1] `http://stackoverflow.com/questions/4475996/given-prime-number-n-compute-the-next-prime`

The second condition holds for large enough n for given α and d.

$$t = g(d)\ \Theta\left(n^{1/d}(1-2\alpha)^{2-1/d}\right) > 2$$

Here g is some computable function that depends only on d. The running time is $O(kn) = O(n/(1-2\alpha))$.

The number of midpoints on each side is at least

$$\frac{n-b}{2} = \frac{n - \lfloor (1-2\alpha)n \rfloor}{2} \geq \alpha n.$$

The number of intersected balls is bounded by

$$2b/(t-2) = O\left(\frac{n(1-2\alpha)}{n^{1/d}(1-2\alpha)^{2-1/d}}\right) = O\left(\left(\frac{n}{(1-2\alpha)}\right)^{1-1/d}\right). \qquad \square$$

Remark 2.5. *As the dependence of d on the running time is very subtle, we discuss it here in detail. For a moment we consider α as constant, but not the dimension d. It is easy to see that the running time depends quadratically on d: one linear factor stems from the fact that every primitive operation on vectors of length d takes $\Theta(d)$ time; the other linear factor comes from the choice of $k = \Theta(d)$. However, n must be exponential in d to satisfy condition (2.2). This leads to the following statement about the running time $T(n, d)$:*

$$\forall d, n \ :\ n > g(d) \ \Rightarrow\ T(n,d) \leq c \cdot d^2 n,$$

where c is some fixed constant and g is some exponential function. Does this imply

$$T(n,d) = O(d^2 n)?$$

The answer is surprisingly no! Recall by the definition of big O notation the last statement is equivalent to:

$$\exists\, n_0, d_0\ \forall\, d > d_0, n > n_0\ \ : T(n,d) \leq c \cdot d^2 n.$$

The difference is that in the big O notation, the inequality should hold for any n larger than some universal constant n_0. This is in contrast to the previous statement, where the required magnitude of n depends on d as well!

The proof of the Theorem 2.2 differs mainly in some nuances from the previous one.

Proof of Theorem 2.2. Assume the slowly growing function $f : \mathbb{N} \to \mathbb{N}$ is given as in the condition of Theorem 2.2. We define $b = \lceil n/f(n) \rceil$ and k some prime with $k \in [df(n), 2df(n)]$. We will show first that Condition (2.1) and (2.2) hold for these choices. The first condition is seen as follows.

$$kb \geq df(n)\lceil n/f(n) \rceil \geq nd$$

The second condition holds for large enough n for given d.

$$t = g(d)\ \Theta\left(\frac{n^{1/d}}{f(n)^{2-1/d}}\right) > 2$$

The term $g(d)$ depends on d only and thus is constant for constant d. Here we need that $f(n) = o\left(n^{1/(2d-1)}\right)$.

The running time is $O(kn) = O(nf(n))$. The number of midpoints on each side is at most

$$\frac{n-b}{2} = \frac{n - \lceil n/f(n) \rceil}{2} = n/2 - O\left(\frac{n}{f(n)}\right).$$

The number of intersected balls is bounded by

$$2b/(t-2) = O\left(\frac{n}{f(n)} \frac{f(n)^{2-1/d}}{n^{1/d}}\right) = O\left((nf(n))^{1-1/d}\right) = O\left(\frac{n}{f(n)}\right).$$

Here the last equality follows from the fact that $f(n) = o\left(n^{1/(2d-1)}\right)$. It remains to show that the number of balls on each side is at least $n/2 - o(n)$. Note that the number of balls completely contained on either side of the hyperplane is bounded by the number of midpoints on this side minus the number of intersected balls, which equals

$$\left(\frac{n}{2} - O\left(\frac{n}{f(n)}\right)\right) - O\left(\frac{n}{f(n)}\right) = \frac{n}{2} - O\left(\frac{n}{f(n)}\right)$$

as claimed. □

2.2.1 Overview of the approximation algorithm.

Our algorithm for Theorem 2.4 consists of two steps. In the first step, we find a direction $\vec{d}$ in which the balls from $\mathcal{D}$ are "spread out nicely". More precisely, for an arbitrary (oriented) line ℓ consider the set P of points that results from orthogonally projecting all centers of balls from $\mathcal{D}$ onto ℓ. Denote by $p_1, \dots, p_n$ the order of points from P sorted along ℓ. We want to find an $(n-b)/2$-separator orthogonal to ℓ. This means that the separating hyperplane $\mathcal{H}$ must intersect ℓ somewhere in between $p_{\lfloor (n-b)/2 \rfloor}$ and $p_{\lfloor (n+b-1)/2 \rfloor}$.

However, we also need to guarantee that not too many points from P are within distance one of $\mathcal{H}$, which may or may not be possible depending on the choice of $\vec{d}$. Therefore we try several possible directions and select the first one among them that works. In order to evaluate the quality of a direction, we use as a simple criterion the *spread*, defined to be the distance between $p_{(n-b)/2}$ and $p_{(n+b)/2}$. Given a line ℓ with sufficient spread, we can find a suitable $(n-b)/2$-separator orthogonal to ℓ in the second step of our algorithm, as the following lemma demonstrates. Note the safety cushion of width one to interval boundary. This gap ensures that there is no interference with other points of $\mathcal{D}$.

Lemma 2.6. *Given a set P of b (one-dimensional) points in an interval $[\ell, r]$ of length $w = r - \ell > 2$, we can find in $O(b)$ time a point $p \in (\ell + 1, r - 1)$ such that at most $2b/(w-2)$ points from P are within distance one of p.*

Proof. We select $\lceil (w-2)/2 \rceil$ pairwise disjoint open sub-intervals of length two in (ℓ, r). By the pigeonhole principle at least one these intervals contains at most $b/\lceil (w-2)/2 \rceil \leq 2b/(w-2)$ points from P. Select p to be the midpoint of such an interval.

Algorithmically, we can find the interval by just counting for each interval how many points fall into it and storing this information in an array. However, this requires rounding. Here we want to show that it is also possible without rounding.

Another way is to find such an interval using a kind of binary search on the intervals: We maintain a set of points and a range of intervals. At each step consider the median interval I and test for every point whether it lies in I, to the left of I, or to the right of I. Then either I contains at most $2b/(w-2)$ points from P and we are done, or we recurse on the side that contains fewer points per interval, after discarding all points and intervals on the other side. The process stops as soon as the current range of intervals contains at most $2b/(w-2)$ points from P, at which point any of the remaining intervals can be chosen. Given that we maintain the ratio between the number of points and the number of intervals, the process terminates with an interval of the desired type. As the number of points decreases by a constant factor in each iteration, the overall number of comparisons can be bounded by a geometric series and the resulting runtime is linear. □

2.2.2 How to find a good direction.

Our algorithm tries k different directions and stops as soon as it finds a direction with spread at least t (see Theorem 2.4). For a given direction the spread can be computed in $O(n)$ time using linear time rank selection [18]. In the remainder of this section, we will discuss how to select an appropriate set of directions such that one direction is guaranteed to have spread at least t.

For this we need a bound on the number of balls simultaneously within distance $w_1, \ldots, w_d$ of some hyperplanes $\mathcal{H}_1, \ldots, \mathcal{H}_d$. Below we give an easy formula based on a volume argument. This formula in turn motivates our choice of directions, which we will explain thereafter.

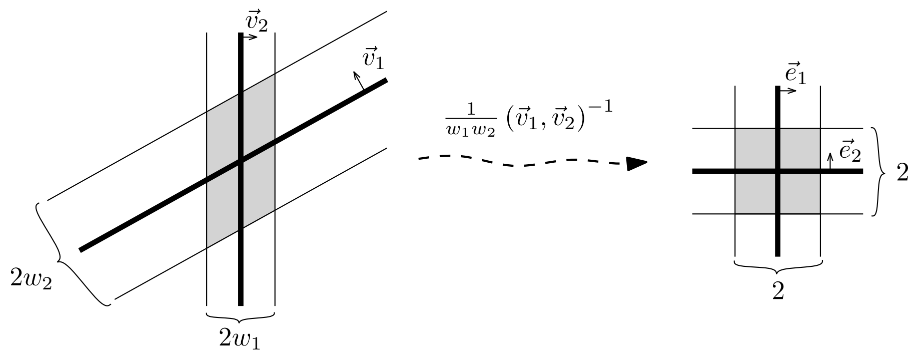


Figure 2.2: Illustration of Lemma 2.7

Lemma 2.7. *Let $\vec{v}_1, \ldots, \vec{v}_d \in S^{d-1} \subset \mathbb{R}^d$ be linearly independent directions and let $\mathcal{H}_1, \ldots, \mathcal{H}_d$ be hyperplanes with corresponding normal directions. Then the maximal number of pairwise disjoint unit balls entirely within distance $w_1, \ldots, w_d$ of $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_d$, respectively, is bounded from above by*

$$\frac{2^d w_1 \ldots w_d}{|\det(\vec{v}_1, \ldots, \vec{v}_d)| V_d},$$

where V_d denotes the volume of the d-dimensional unit ball.

Proof. For each hyperplane $\mathcal{H}_i$ consider the region S_i within distance w_i of $\mathcal{H}_i$. We want to count the number of balls in $S := \bigcap_i S_i$. As each ball has volume V_d and they are pairwise disjoint, it is sufficient to bound the volume of S. The volume of S depends linearly on $w_1, \ldots, w_d$, so we scale them all to one. We can map the linearly independent vectors $(\vec{v}_1, \ldots, \vec{v}_d)$ to the standard basis $(e_1, \ldots, e_d)$ by multiplying with the matrix $(\vec{v}_1, \ldots, \vec{v}_d)^{-1}$. The volume changes by this transformation by a factor of $1/\det(\vec{v}_1, \ldots, \vec{v}_d)$. After this transformation, S' is a cube with side length two. □

The bound in Lemma 2.7 depends on the determinant formed by the d direction vectors, which corresponds to the volume of the $(d-1)$-simplex spanned by them. In order to obtain a good upper bound, we must guarantee that this volume does not become too small. Ensuring this reduces to the *Heilbronn Problem*: Given $k \in \mathbb{N}$ and a compact region $P \subset \mathbb{R}^d$ of unit volume, how can we select k points from P as to maximize the area of the smallest d-simplex formed by these points? Heilbronn posed this question for $d = 2$, the natural generalization to higher dimension was studied by Barequet [15] and Lefmann [47]. We use the following simple explicit construction that goes back to Erdős in the plane and was generalized to higher dimension by Barequet. This construction is the moment curve modulo k. This is also where we use that k is a prime.

Lemma 2.8 ([15, 62])**.** *Given a prime k, let $P = \{p_0, \dots, p_{k-1}\} \subset [0,1]^d$ with*

$$p_i = \frac{1}{k}\left(i, i^2 \bmod k, \dots, i^d \bmod k\right).$$

Then the smallest d-simplex spanned by $d+1$ points from P has volume at least $1/(d!k^d)$.

In order to obtain the desired direction vectors we proceed as follows: Use Lemma 2.8 to generate k points $p_0, \dots, p_{k-1}$ in $[0,1]^{d-1}$. Then lift the points to $S^{d-1} \subset \mathbb{R}^d$ using the map

$$f : (x_1, \dots, x_{d-1}) \mapsto \frac{(x_1 - \frac{1}{2}, \dots, x_{d-1} - \frac{1}{2}, \frac{1}{2})}{\|(x_1 - \frac{1}{2}, \dots, x_{d-1} - \frac{1}{2}, \frac{1}{2})\|}$$

and denote the resulting set of directions by $D = \{\vec{v}_0, \dots, \vec{v}_{k-1}\}$ with $\vec{v}_i = f(p_i)$. Note that all the vectors are on the upper hemisphere.

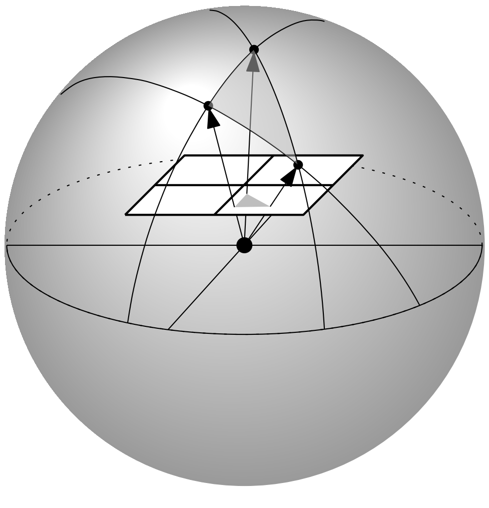


Figure 2.3: Illustration of the projection onto the sphere and the correspondence between the size of the triangles and the volume spanned by the corresponding simplices.

Lemma 2.9. *For any d vectors $\vec{v}_{i_1}, \dots, \vec{v}_{i_d}$ from D we have*
$|\det(\vec{v}_{i_1}, \dots, \vec{v}_{i_d})| \geq 2^{d-1}/((d-1)!d^{d/2}k^{d-1})$.

Proof. Let $p_j = (x_{j,1}, \dots, x_{j,d-1})$, for $j \in \{0, \dots, d\}$. Then

$$|\det(\vec{v}_{i_1}, \dots, \vec{v}_{i_d})| =$$

$$\left|\det\begin{pmatrix} x_{i_1,1}-\frac{1}{2} & \dots & x_{i_d,1}-\frac{1}{2} \\ \vdots & \ddots & \vdots \\ x_{i_1,d-1}-\frac{1}{2} & \dots & x_{i_d,d-1}-\frac{1}{2} \\ \frac{1}{2} & \dots & \frac{1}{2}\end{pmatrix}\right| \prod_{j=1}^{d} \frac{1}{\|(x_{i_j,1}-\frac{1}{2},\dots,x_{i_j,d-1}-\frac{1}{2},\frac{1}{2})\|}$$

$$= \frac{1}{2}\left|\det\begin{pmatrix} x_{i_1,1} & \dots & x_{i_d,1} \\ \vdots & \ddots & \vdots \\ x_{i_1,d-1} & \dots & x_{i_d,d-1} \\ 1 & \dots & 1\end{pmatrix}\right| \prod_{j=1}^{d} \frac{1}{\|(x_{i_j,1}-\frac{1}{2},\dots,x_{i_j,d-1}-\frac{1}{2},\frac{1}{2})\|},$$

where the determinant on the last line describes the volume of the $(d-1)$-simplex spanned by $p_{i_1},\dots,p_{i_d}$. According to Lemma 2.8 this determinant is bounded by $1/((d-1)!k^{d-1})$ from below. Also note that all p_i are in the unit cube and so all coordinates of the vector $(x_{i_j,1}-\frac{1}{2},\dots,x_{i_j,d-1}-\frac{1}{2},\frac{1}{2})$ are between $-1/2$ and $1/2$ and therefore has norm smaller or equal to $\sqrt{d/4}$. It follows that

$$|\det(\vec{v}_{i_1},\dots,\vec{v}_{i_d})| \geq \frac{1}{2(d-1)!k^{d-1}} \prod_{j=1}^{d} \frac{1}{\sqrt{d/4}} = \frac{2^{d-1}}{(d-1)!d^{d/2}k^{d-1}}. \qquad \square$$

We are now ready to prove Theorem 2.4.

Proof. The algorithm goes as follows. Compute directions $\vec{v}_1,\dots,\vec{v}_k$ as in Lemma 2.9. For each $i\in\{1,\dots,k\}$ consider the sequence of center points of the disks in $\mathcal{D}$, sorted according to direction $\vec{v}_i$, and denote by S_i the middle b points in this order (rank $\lfloor(n-b)/2\rfloor$ up to $\lfloor(n+b-2)/2\rfloor$). We can bound

$$kb = \sum_{i=1}^{k}|S_i| \leq (d-1)n + \sum_{i_1<\dots<i_d}|S_{i_1}\cap\ldots\cap S_{i_d}|,$$

noting that a point that is contained in at most $d-1$ sets S_i is counted $d-1$ times on the right hand side, whereas a point that is contained in $a\geq d$ sets on the left side is counted $d-1+\binom{a}{d}\geq a$ times on the right hand side.

Denote by w_i the width of S_i in direction $\vec{v}_i$ (which is the spread of $\vec{v}_i$). We claim that $w_i\geq t$, for some $i\in\{1,\dots,k\}$.

For the purpose of contradiction assume $w_i<t$, for all $i\in\{1,\dots,k\}$. Together with Lemma 2.7 and Lemma 2.9 we get

$$\begin{aligned} kb &= \sum_{i=1}^{k}|S_i| \leq (d-1)n + \sum_{i_1<\dots<i_d}\frac{2^d w_{i_1}\cdots w_{i_d}}{|\det(\vec{v}_{i_1},\dots,\vec{v}_{i_d})|V_d} \\ &< (d-1)n + \sum_{i_1<\dots<i_d}\frac{2^d t^d}{V_d}\frac{(d-1)!d^{d/2}k^{d-1}}{2^{d-1}} \\ &= (d-1)n + \binom{k}{d}\frac{2t^d(d-1)!d^{d/2}k^{d-1}}{V_d} \\ &\leq (d-1)n + \frac{2d^{(d-2)/2}}{V_d}t^d k^{2d-1}. \end{aligned}$$

In combination with Condition (2.1) we get

$$dn \leq kb < (d-1)n + \frac{2d^{(d-2)/2}}{V_d} t^d k^{2d-1}$$

and so

$$t^d > \frac{V_d}{2d^{(d-2)/2}} \frac{n}{k^{2d-1}},$$

in contradiction to the definition of t in Condition (2.2). Therefore, our assumption $w_i < t$, for all $i \in \{1, \ldots, k\}$, was wrong and there is some $w_j \geq t$.

Using Lemma 2.6 on the set S_j projected to a line in direction $\vec{v}_j$ we obtain a hyperplane $\mathcal{H}$ orthogonal to $\vec{v}_j$ that intersects at most $2b/(w_j - 2) \leq 2b/(t-2)$ balls from $\mathcal{D}$. By Lemma 2.6 the hyperplane $\mathcal{H}$ has distance greater than one to any disk in $\mathcal{D}$ whose center is not in S_j, and so $\mathcal{H}$ is the desired separator.

Regarding the runtime bound, as stated above we can compute the spread of any direction in $O(n)$ time, which yields $O(kn)$ time for k directions. The second step of finding $\mathcal{H}$ can be done in $O(b) = O(n)$ time by Lemma 2.6. Therefore the overall runtime is $O(kn)$. □

2.3 An Exact Algorithm in the Plane

In this section we describe a deterministic linear time algorithm to construct a halving line ℓ for a given set $\mathcal{D}$ of n disks in the plane. The line ℓ bisects $\mathcal{D}$ perfectly (at most $\lfloor n/2 \rfloor$ centers lie on either side) and it intersects at most $O(n^\delta)$ disks, where δ may be chosen arbitrarily close to 2/3.

The algorithm follows the prune-and-search paradigm and it is inspired by the prune and search algorithm to find a ham-sandwich cut in deterministic linear time [49].

We will not consider all possible halving lines, but restrict our attention to the halving lines with slope in a certain range. This range becomes smaller and smaller after each iteration. We discard a constant fraction of the disks after each iteration and the disks we discard do not intersect any halving line that we still consider. Each iteration will take linear time, and thus the overall running time is linear as well.

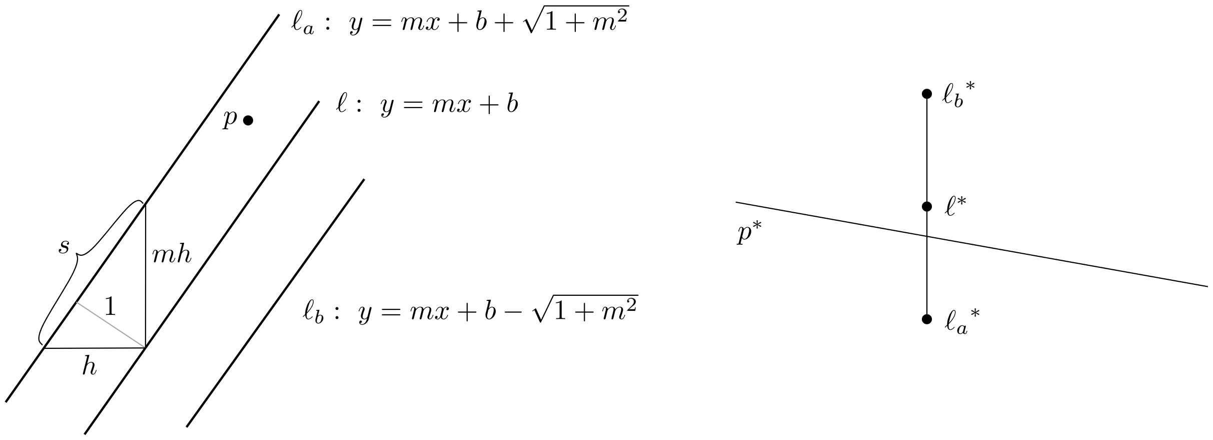
The section is divided into several subsections. We first repeat some definitions regarding point-line duality in Section 2.3.1. In particular, we will investigate how we can determine in the dual whether a given line intersects a unit disk. Further we will investigate what it means in the dual that a disk is intersected by some halving line with slope within some given interval. Section 2.3.2 outlines the basic steps of our prune and search algorithm. In particular we will summarize in the Iteration Lemma 2.15 what our algorithm performs in one iteration. In Section 2.3.3 we will analyze the algorithm under the assumption that the Iteration Lemma 2.15 is true. The remaining sections are devoted to the proof of Lemma 2.15.

2.3.1 Point-line Duality.

As our algorithm works in the dual arrangement, we first briefly review this duality and how it applies to line-disk intersections.

The standard duality transform maps a point $p = (p_x, p_y)$ to the line $p^*\colon y = p_x x - p_y$ and a non-vertical line $g : y = mx + b$ to the point $g^* = (m, -b)$. This transformation is both incidence preserving ($p \in g \iff g^* \in p^*$) and order reversing (p is above $g \iff p^*$ is below g^*). Given a set P of points in the plane, the dual arrangement $\mathcal{A}(P^*)$ is defined by the lines in $P^* = \{p^* \mid p \in P\}$. In order to avoid parallel lines we assume that no two points in P have the same x-coordinate (which can be achieved by a rotation of the plane).

The following lemma characterizes line-disk intersections in the dual plane.



$\ell_a : y = mx + b + \sqrt{1+m^2}$
$\ell : y = mx + b$
p
s
mh
1
h
$\ell_b : y = mx + b - \sqrt{1+m^2}$
${\ell_b}^*$
ℓ^*
p^*
${\ell_a}^*$

Figure 2.4: When does a line ℓ intersect a unit disk centered at p?

Lemma 2.10. *Let $\ell : y = mx + b$ be a non-vertical line and let p denote the center of a unit disk D. Then D intersects ℓ if and only if the line p^* intersects the vertical segment $s = [(m, -b - \sqrt{m^2+1}), (m, -b + \sqrt{m^2+1})]$.*

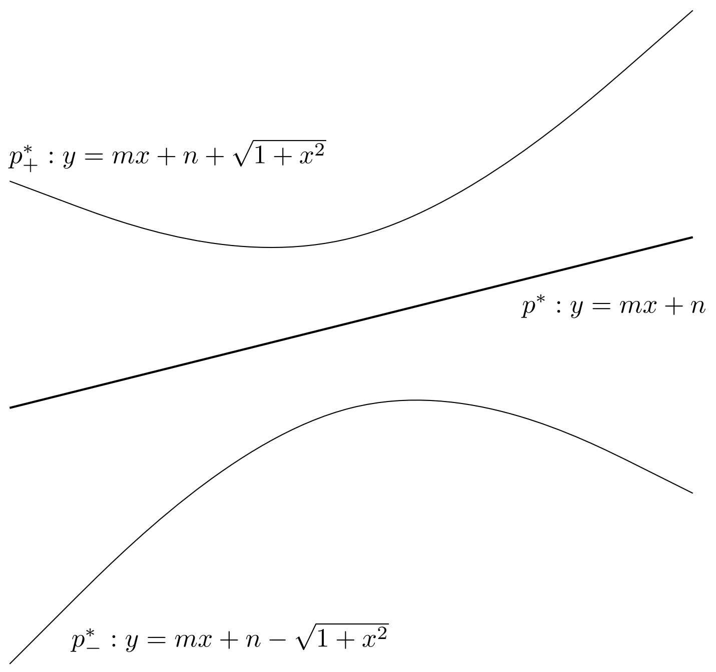
Proof. Consider ℓ and the two lines ℓ_a (above) and ℓ_b (below) at distance 1 from ℓ, see Figure 2.4. Then D intersects ℓ if and only if p is below ℓ_a and above ℓ_b. Equivalently, in the dual, D intersects ℓ if and only if p^* intersects the vertical line segment $\ell_a^*\ell_b^*$ at $x = m$. It remains to calculate the y-coordinates of the endpoints of $\ell_a^*\ell_b^*$.

Consider a right triangle T for which one side determines the horizontal distance h and another side determines the vertical distance mh between ℓ and ℓ_a. Denote the length of the third side of T by s. Then the area of T is $\frac{1}{2}s = \frac{1}{2}mh^2$. By Pythagoras we have $s^2 = h^2(1+m^2)$, which together yields $1 + m^2 = (mh)^2$, and so $mh = \sqrt{1+m^2}$. $\square$

If we view Lemma 2.10 from the perspective of a unit disk D with center p, then the set of lines that intersect D dualizes to the set of points (x, y) whose vertical distance to p^* is at most $\sqrt{1+x^2}$. We call this closed region of points the (dual) 1-*tube* of D, see Figure 2.5. Note that the function $\sqrt{1+x^2}$ is strictly convex and so the 1-tube is bounded by a strictly convex function from above and by a strictly concave function from below.

We may assume that n is odd: If n is even, remove one arbitrary disk and observe that any halving line for the resulting set of disks is also a halving line for the original set.

A halving line ℓ for P corresponds to a point ℓ^* in the dual arrangement that has no more than half of the lines from P^* above it and no more than half of the lines below it. The set of these points is referred to as the *median level* of the arrangement induced by P^*. Since n is odd, for any x-coordinate there is exactly one such point, and so we can regard the median level as a function from $\mathbb{R}$ to $\mathbb{R}$.



$p_+^* : y = mx + n + \sqrt{1+x^2}$
$p^* : y = mx + n$
$p_-^* : y = mx + n - \sqrt{1+x^2}$

Figure 2.5: The 1-tube of the disk centered at $p = (m, -n)$. It is bounded from below by the function $p_-^* = p^* - \sqrt{1+x^2}$ and from above by $p_+^* = p^* + \sqrt{1+x^2}$.

Definition 2.11 (λ-level)**.** *For $\lambda \in \{0, \ldots, n-1\}$ the λ-level of an arrangement $\mathcal{A}(L)$ of a set L of n lines is the set of points on the arrangement with exactly λ lines below it.*

In the following, whenever we have given a set P of points we refer to the λ-level of the dual line arrangement of P just as λ-level and implicitly assume that the underlying line arrangement is clear from the context. In the algorithm it will be important to consider not only the $\lfloor n/2 \rfloor$-level. Halving lines of the original set of disks will correspond to a λ-level of the current set of disks. The value of λ is maintained by the algorithm.

We define a λ-line ℓ of a point set P as a line through at least one point $p \in P$ with exactly λ points above ℓ. The dual of a λ-line is a point on the λ-level. Note that by this definition $\lfloor n/2 \rfloor$-lines are halving lines.

Recall that the algorithm follows the prune-and-search paradigm. In the process we will consider only halving lines with slope in a certain range. Thus in the dual only the λ-level within a range of certain x-coordinates is considered.

Definition 2.12 (Slab)**.** *Given two vertical lines, the closed region bounded by them is called a* slab*. The distance between the two bounding vertical lines is the* width *of the slab.*

For convenience, the slab S with x-coordinates in the interval $[l, r]$ is also denoted by $[l, r]$. The width of S is simply $r - l$. As mentioned before, some disks will be discarded, which do not intersect any of the potential halving lines. However the algorithm runs in the dual and we will discard lines corresponding to midpoints of disks. To describe this in a handy fashion we make the following definition.

Given a line arrangement $\mathcal{A}(L)$ in the dual, some level λ, we denote by $\lambda(x)$ the function that takes as input an x-coordinate and returns the corresponding point on the λ-level. Thus $\lambda(x_0)^*$ is the dual line of the point on the λ-level at x-coordinate x_0. Note that $\lambda(x_0)^*$ is a λ-line in the primal. A line ℓ in the dual *interferes* within slab $S = [l, r]$ and level λ if there is some $x_0 \in [l, r]$ such that the line $\lambda(x_0)^*$ intersects the unit disk with midpoint ℓ^* in the primal, see Figure 2.6. In other words, if we want to find a halving line with slope $m \in [l, r]$ then any line that does not interfere with the $\lfloor n/2 \rfloor$-level within the slab $[l, r]$ can be discarded.

Given a line arrangement $\mathcal{A}(L)$ and some $\lambda \in \mathbb{N}$, we define the 1-tube τ_λ of the λ-level of $\mathcal{A}(L)$ as the region enclosed by the functions $\lambda(x) + \sqrt{1 + x^2}$ and $\lambda(x) - \sqrt{1 + x^2}$, see Figure 2.6.

Lemma 2.13. *For a set of disks $\mathcal{D}$ with midpoints P and a disk D with midpoint $p = (p_x, p_y) \in P$, the following statements are equivalent:*

1. *There is a λ-line ℓ with slope $m \in [l, r]$ that intersects D.*
2. *The line p^* interferes with the λ-level within the slab $S = [l, r]$.*
3. *The line p^* intersects τ_λ within the slab $S = [l, r]$.*

Proof. By the definition of interfering Condition 1 and 2 are equivalent.

[1 $\Rightarrow$ 3] By Lemma 2.10 the first condition is equivalent to the fact that the vertical distance between the line p^* and the point ℓ^* in the dual is smaller or equal to $\sqrt{m^2 + 1}$. Note that every λ-line dualizes to a point on the λ-level and this implies p^* intersects τ_λ. Because the slope m is in $[l, r]$ the 1-tube τ_λ is intersected within the slab $S = [l, r]$.

[3 $\Rightarrow$ 1] Assume p^* intersects τ_λ. Then exists some m in the interval $[l, r]$ such that the vertical distance between $\lambda(m)$ and p^* is smaller or equal to $\sqrt{m^2 + 1}$. Define the line ℓ as $\lambda(m)^*$ and note that the disk with midpoint p is intersected by ℓ according to Lemma 2.10. $\square$

As we have already mentioned we cannot test directly if some line intersects τ_λ, but we can test some necessary criteria. To this end we will explain later how to construct a trapezoid T containing the λ-level within some slab S, see Figure 2.6. For now let us just assume we have given such a trapezoid T. We define the 1-tube $\tau = \tau_T$ of T as the region with vertical distance at most $\sqrt{1 + x^2}$ from T, but excluding T.

Lemma 2.14. *Given a line arrangement $\mathcal{A}(L)$, $\lambda \in \mathbb{N}$ and a trapezoid T in a slab $S = [l, r]$ that contains the λ-level of T. Then a line that does not intersect $T \cup \tau$ does not intersect τ_λ*

Proof. It is sufficient to show that τ_λ is contained in $T \cup \tau$. The lower boundary of τ has at x-coordinate t vertical distance $\sqrt{t^2 + 1}$ from T and thus at least this vertical distance from the λ-level. The same argument works for the upper boundary. $\square$

In order to test if a line intersects $T \cup \tau$, it is sufficient to compute where this line intersects the vertical lines bounding $T \cup \tau$. This can be computed in constant time.

2.3.2 Overview of the Algorithm.

The algorithm works in the dual arrangement. This means that we find a point on the $\lfloor n/2 \rfloor$-level of the dual line arrangement which interferes with at most $O(n^\delta)$ lines. We will show that δ can be chosen arbitrarily close to 2/3.

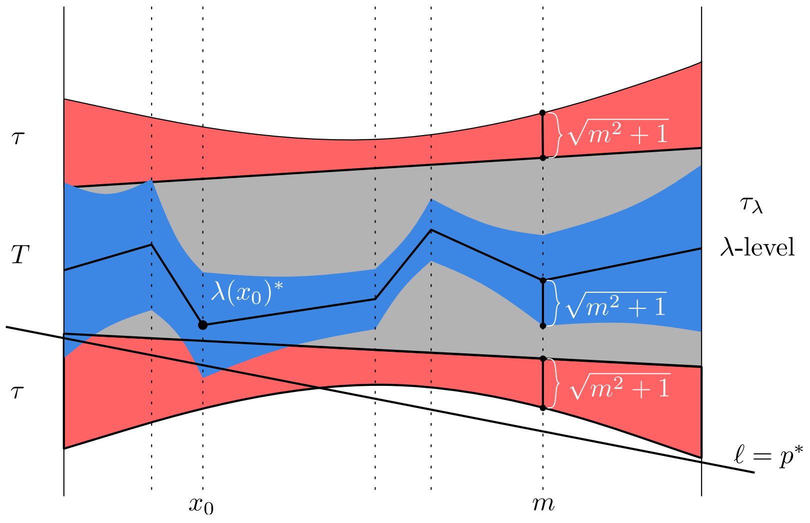


Figure 2.6: The λ-level; in light blue the 1-tube τ_λ; in gray a trapezoid T containing the λ-level, but not necessarily τ_λ; in light red the 1-tube of T. The line $\ell = p^*$ and thus the disk with midpoint p is intersected by the λ-line $\lambda(x_0)^*$. At x-cordinate m the vertical distance between $\lambda(m)$ and the lower boundary of its 1-tube equals $\sqrt{m^2+1}$ and the vertical distance between the trapezoid T and the lower boundary of its 1-tube also equals $\sqrt{m^2+1}$. Therefore, the 1-tube of the λ-level is contained in the 1-tube of the trapezoid T union the trapezoid. ($\tau_\lambda \subseteq \tau \cup T$)

Recall that we will follow the prune-and-search paradigm. The algorithm consists of three phases. It starts with an initialization, then goes into some loop, and finally stops returns an appropriate point.

Crucial for the linear runtime bound is that a constant fraction of all lines from L will be discarded after each iteration. However, by discarding some lines also our level of interest—which is the median level of the original set of lines—changes. Therefore this level also appears as a parameter of the algorithm. We denote this parameter by $\lambda \in \{0, 1, 2, \dots, |L| - 1\}$. Initially $\lambda = \lfloor |L|/2 \rfloor$. We will work in a slab S, which will be narrowed in each iteration. The width of S and the number of vertices in S are two more important parameters of the algorithm. We define a *vertex* as an intersection of two lines. Two global parameters $\gamma, \varepsilon \in (0, \frac{1}{5})$ must be chosen at the start of the algorithm. These parameters are constant for the entire algorithm. The parameter δ above depends on γ and ε in a way that smaller γ and ε imply δ to be closer to $2/3$.

As initialization of the algorithm the slab $S = [0, 1]$ is subdivided into 100 slabs of equal width, and no line is discarded. One of the slabs contains at most

$$v \leq \frac{1}{100(1/2 + \varepsilon)^2} n^2$$

vertices. At the end of Section 2.3.5, we will explain how to find this slab.

The final phase of the algorithm returns any point on the λ-level of the final set of lines.

Before we will describe a single iteration, the following lemma summarizes input and output of one iteration.

Lemma 2.15 (Iteration Lemma)**.** *The algorithm has two global parameters, $\gamma, \varepsilon \in (0, \frac{1}{5})$. A single iteration gets as input:*

- *A set of n lines L in general position.*
- *A slab $S = [l, r] \subseteq [0, 1]$ of width $w = r - l$. The number of vertices of the line arrangement $\mathcal{A}(L)$ inside S is denoted by v. (The number v might not be explicitly known.)*
- *A level-parameter $\lambda \in \{0, \ldots, n-1\}$.*

We require the Width Condition *that is*

$$w \geq \frac{400}{\gamma^2\varepsilon^2}\,\frac{\log n}{n}\,. \tag{WC}$$

The algorithm maintains the Vertex Invariant

$$v \leq \frac{n^2}{100(1/2+\varepsilon)^2}\,. \tag{VI}$$

This means at the end and at the start of each iteration this inequality is satisfied. One iteration of the algorithm runs in linear time and outputs:

- *A slab $\overline{S} \subseteq S \subseteq [0, 1]$ of width $\overline{w} = \frac{(1-2\gamma)}{4}w$*
- *A set of exactly $\overline{n} = n - \lfloor n(\frac{1}{2} - \varepsilon)\rfloor = \lceil(\frac{1}{2} + \varepsilon)n\rceil$ lines $\overline{L} \subseteq L$ such that no line $\ell \in L \setminus \overline{L}$ interferes with the λ-level of L within $\overline{S}$.*
- *A level-parameter $\overline{\lambda} \in \{1, \ldots, n\}$ such that the $\overline{\lambda}$-level in $\overline{L}$ corresponds to the λ-level in L.*

Note that after the initialization the Vertex Invariant (VI) of the previous lemma is satisfied.

The iteration will repeat till the Width Condition (WC) is violated. Then we will go into the final phase. The outline of an iteration step is given below. We put in brackets references to sections and lemmas that give details and proofs of each step.

1. Divide S in four subslabs $S_1, \ldots, S_4$, such that the width of each subslab equals $w/4$.
2. Let v_i be the number of vertices in slab S_i. Compute an approximate number p_i of vertices in each slab, satisfying

   $$v_i \leq p_i \leq v_i + cn^2,$$

   for $c = \varepsilon/100$.

   [Lemma 2.17 in Section 2.3.5]
3. Define $S' = [l', r']$ as the slab with smallest p_i. We will show that for the number of vertices v' within this slab the Vertex Condition holds

   $$v' \leq \frac{n^2}{100}. \tag{2.3}$$

   This condition is tightly connected to the Vertex Invariant (VI).

   [Section 2.3.6]

4. Construct a *trapezoid* $T \subseteq S'$. T contains the λ-level of $\mathcal{A}(L)$ within S' and at most half of the lines from L intersect T. For this step we will use the Vertex-Invariant (VI).

   [Lemma 2.16 in Section 2.3.4]

5. Recall the γ-*core* C_γ is the central $(1-2\gamma)$-slab of S', that is, $\mathrm{C}_\gamma = [l'+\gamma w', r'-\gamma w']$. See Figure 2.7 for an illustration. We continue our search in the next iteration in $\overline{S} = \mathrm{C}_\gamma$. At most εn lines intersect τ but not T within S'. For this step we use the Width Condition (WC). Intuitively this means that the width of the slab is not too small.

   [1-tube $\Rightarrow$ Section 2.3.1, Lemma 2.18 in Section 2.3.7]

6. Discard *exactly* $\lfloor(\frac{1}{2}-\varepsilon)n\rfloor$ lines from L that do not intersect $\overline{S} \cap (\tau \cup T)$.

   [Lemma 2.13 and 2.14 $\Rightarrow$ can be discarded, Lemma 2.16 and 2.18 $\Rightarrow$ at least $(\frac{1}{2}-\varepsilon)n$]

7. We adjust λ accordingly: decrease λ by the number of lines discarded that are below τ.

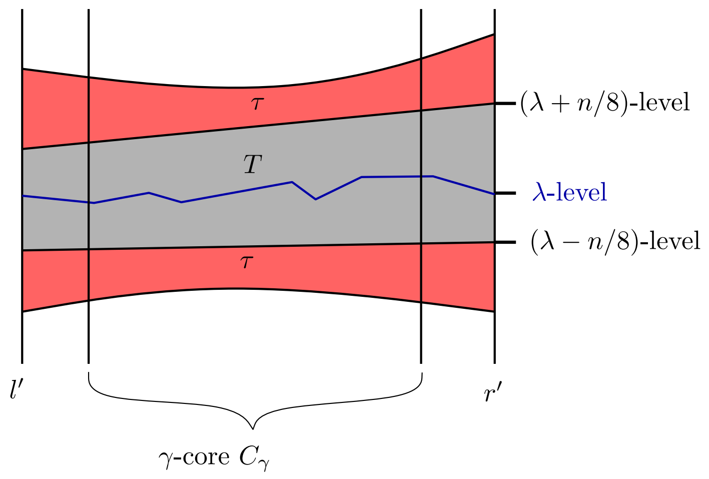


Figure 2.7: The Slab S' together with the trapezoid T in light red and the 1-tube τ in dark green. Note the λ-level is completely contained in T. The drawing is schematic the function $\sqrt{1+x^2}$ looks, except very close to 0, as if it is a linear function. Here, we overemphasize the concave/convex nature.

In Section 2.3.3 we will analyze the algorithm assuming the Iteration Lemma 2.15. The following sections are devoted to give details to the steps above and prove their correctness. In particular this proves the Iteration Lemma 2.15. Section 2.3.4 explains the construction of the trapezoid and shows that the λ-level is always inside the trapezoid. Section 2.3.5 shows how to count approximately vertices inside a slab. Section 2.3.6 shows why the Vertex Invariant (VI) is maintained and the Vertex Condition (2.3) holds. Section 2.3.7 shows that the number of lines intersecting τ, but not T is low. This part is very technical.

In each section we may draw connections to the other sections. We will also explicitly refer to Lemmas proven in previous sections. However, the proofs of each Section can be read independently.

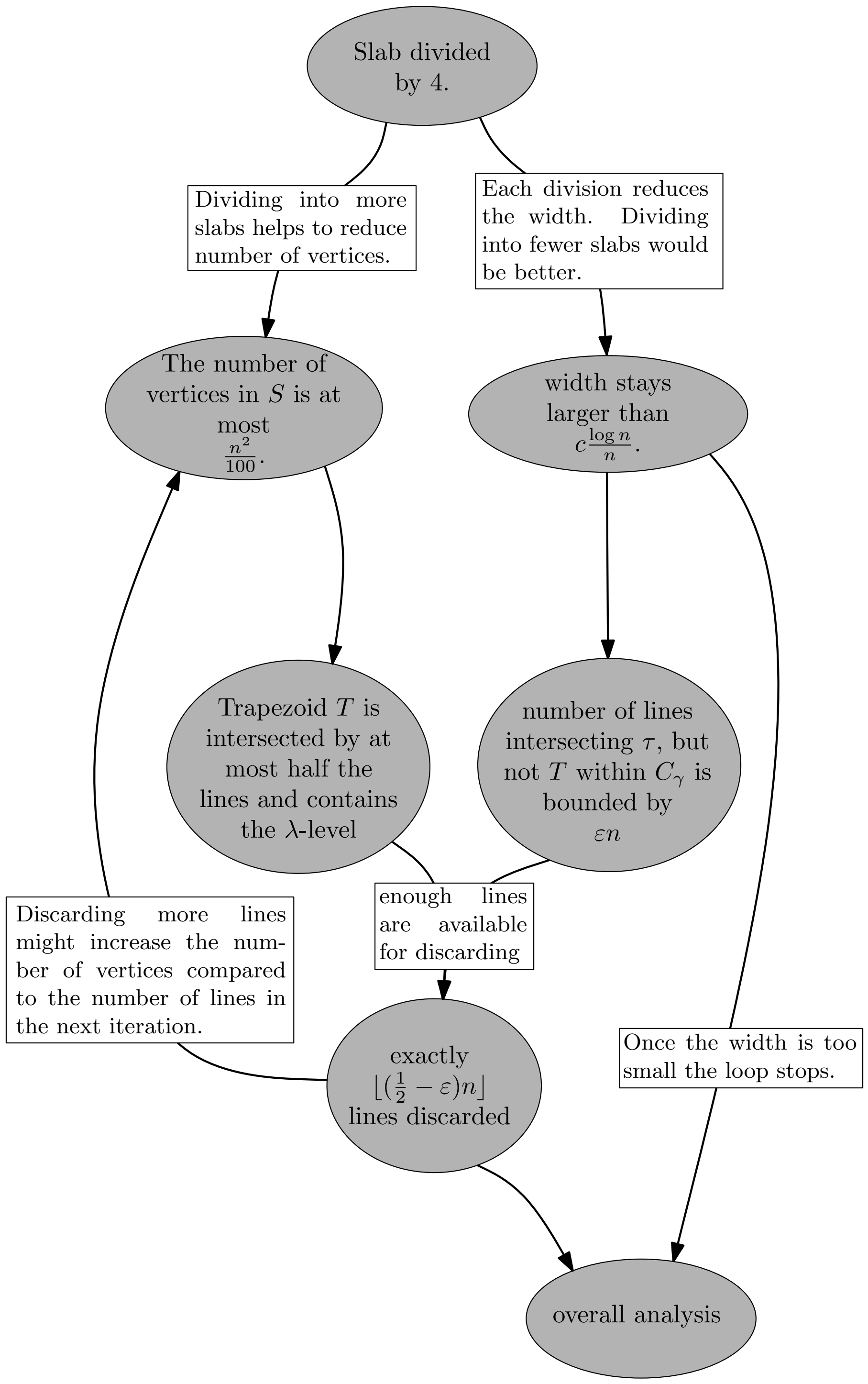

Slab divided by 4.
Dividing into more slabs helps to reduce number of vertices.
Each division reduces the width. Dividing into fewer slabs would be better.
The number of vertices in S is at most $\frac{n^2}{100}$.
width stays larger than $c\frac{\log n}{n}$.
Trapezoid T is intersected by at most half the lines and contains the λ-level
number of lines intersecting τ, but not T within C_γ is bounded by εn
enough lines are available for discarding
Discarding more lines might increase the number of vertices compared to the number of lines in the next iteration.
exactly $\lfloor(\frac{1}{2}-\varepsilon)n\rfloor$ lines discarded
Once the width is too small the loop stops.
overall analysis

Figure 2.8: Design decisions and properties of the algorithm are displayed as bubbles. Connections between them are indicated by arrows and commented in boxes.

In order to keep the notation simple, the meaning of S and n might be different in some sections. It will be explained at the beginning of each section.

We try to sketch connections between the steps of the algorithm, see also Figure 2.8. The problem is that we need to keep the number of vertices in our slab low and simultaneously the width of our slab high. If we were to divide into three slabs, it would not be clear that the number of vertices in S is bounded sufficiently. If we were to divide into five or more slabs the width would be divided by five instead of four in each iteration and thus decrease more rapidly. Once the width is too small we have to stop the iteration and cannot discard any more lines. Recall that we need to bound the number of vertices in order to guarantee that the number of lines intersecting the trapezoid is low. The lower bound on the width is needed to guarantee that the number of lines that do not intersect T but the 1-tube τ is limited. Only lines that intersect neither T nor τ can be discarded. The number of lines discarded per iteration and the reduction of the width gives the overall performance bound of $2/3 + \delta$ for any $\delta > 0$. Also note that we could discard all lines that do not intersect $T \cup \tau$. Those are potentially much more. However, this might increase the number of vertices in S compared to the number of lines and thus fail the Vertex-Invariant (VI).

2.3.3 Analysis of the Algorithm.

In this section we complete the overall analysis of the algorithm, under the assumption that the Iteration Lemma 2.15 is true. We denote by n the initial total number of lines, by n_t the number of lines after the t-th iteration and by w_t the width after the t-th iteration.

To understand a little bit better what is going on, we consider an oversimplified setting first.

For this simplified setting assume the width condition would be

$$w_t \geq \frac{1}{n_t}$$

and also $w = w_0 = 1$. Further, we assume that we always discard half of the lines and that the width will be divided by four in each iteration. The division of the width by four stems from the fact that we divide each slab into four subslabs in each Iteration. Then we have

$$n_t = \frac{n}{2^t} \quad \text{and} \quad w_t = \frac{1}{4^t}.$$

Assume after s iterations our simplified Width Condition fails, that is

$$w_s < \frac{1}{n_s} \;\Rightarrow\; 4^{-s} < \frac{2^s}{n} \;\Rightarrow\; n < 8^s \;\Rightarrow\; s > \frac{\log n}{\log 8}.$$

Thus, the number of remaining lines is

$$n_s = \frac{n}{2^s} = \frac{n}{2^{\frac{\log n}{\log 8}}} = n \cdot n^{\frac{\log(1/2)}{\log 8}} = n \cdot n^{-1/3} = n^{2/3}.$$

We know only the lines remaining after the last iteration can potentially interfere with the λ-level as we discarded only lines that do not interfere with the λ-level.

Now we will do the calculations in a non-oversimplified way.

To disentangle notation, we denote $a_1 = \frac{1}{2} + \varepsilon$, $a_2 = \frac{1}{2} + 2\varepsilon$ and $b = \frac{1-2\gamma}{4}$. Note that when ε and γ is small, a_1 and a_2 is roughly $1/2$ and b is roughly $1/4$. We distinguish between a_1 and a_2 to take care of rounding. We start with $n_0 = n$. Recall that the initialization splits the slab into 100 subslabs and thus we have $w_0 = 1/100$.

We have $n_{t+1} = \lceil a_1 n_t \rceil \geq a_1 n_t$ by definition and we can easily conclude $n_{t+1} \leq a_2 n_t$, as follows:

$$n_{t+1} = \lceil n_t(1/2 + \varepsilon) \rceil \leq n_t(1/2 + \varepsilon) + 1 \leq n_t(1/2 + 2\varepsilon).$$

The last inequality holds as long as $1 \leq n_t \varepsilon$, which is implied by the Width Condition (WC) as follows:

$$n_t \geq n_t w_t \geq \frac{400}{\gamma^2 \varepsilon^2} \log n_t = \frac{400}{\gamma^2 \varepsilon} \log n_t \frac{1}{\varepsilon} \geq \frac{1}{\varepsilon}.$$

Applying these two bounds repeatedly we get

$$a_1^t n \leq n_t \leq a_2^t n.$$

We know about the width after t iterations

$$w_t = b^t / 100.$$

We have to satisfy the Width Condition (WC), that is

$$w_t \geq c \cdot \frac{\log n_t}{n_t}.$$

The constant $c = \frac{100}{\gamma^2 \varepsilon^2}$. Let us consider the case that after s iterations the Width Condition (WC) will be violated the first time. It follows

$$\begin{array}{rcl} w_s & < & c \cdot \dfrac{\log n_s}{n_s} \\ w_s \cdot n_s & < & c \cdot \log n_s < c \cdot \log n \\ b^s/100 \cdot n_s & < & c \cdot \log n \\ b^s \cdot a_1^s n & < & 100 \cdot c \cdot \log n \\ b^s \cdot a_1^s n & < & n^{\varepsilon'} \\ n^{1-\varepsilon'} & < & (1/ba_1)^s \\ (1-\varepsilon') \log n & < & s \cdot \log(1/ba_1) \end{array}$$

$$\Rightarrow \quad s > \frac{(1-\varepsilon') \log n}{\log(1/ba_1)}.$$

The second line follows from the first by rearranging n_s and the observation that $n_s \leq n$. For the third and fourth line we used the bounds on the top and multiplied both sides with 100. For the fifth line, note that $100c \log n \leq n^{\varepsilon'}$ holds for every fixed $\varepsilon' > 0$ and large enough n. The seventh line follows from the previous line by taking logarithms on both sides. Thus we have some lower bound for s. And we know in the last step n_s is bounded from above.

$$\begin{aligned} n_s &\leq a_2^s n \\ &\leq a_2^{\frac{(1-\varepsilon')\log n}{\log(1/ba_1)}} n \\ &\leq n^{1+\frac{(1-\varepsilon')\log a_1}{\log(1/ba_2)}} \end{aligned}$$

The exponent approaches 2/3, as $\varepsilon, \varepsilon'$ and γ goes to zero, since:

$$\log a_1 \to -1, \quad (1-\varepsilon') \to 1, \quad \log 1/ba_2 \to 3.$$

Denote by $R(n)$ the runtime of the algorithm for n disks. Each iteration can be handled in time linear in the number of lines/disks remaining and so

$$R(n) \leq \sum_{t=0}^{t^*} cn_t \leq cn \sum_{t=0}^{t^*} \left(\frac{1}{2}+\varepsilon\right)^t < \frac{2c}{1-2\varepsilon} n = O(n),$$

for some constant c.

2.3.4 Trapezoid Construction

In this section we will define the trapezoid mentioned in Section 2.3.1 and in Step 4. This construction and the following lemma goes back to [49]. We repeat here its proof for self containment. We will be a bit more careful with rounding, but also a bit more generous with the constants. Note that the Vertex Condition (2.3) is exactly so designed as to satisfy the next lemma.

In this section, n denotes the cardinality of some arbitrary set L of lines. Given a slab $S = [l, r]$ we will construct a *trapezoid* $T \subseteq S$. The upper left (right) corner of T is slightly above $(\lambda + \lceil n/8 \rceil)$-level of $\mathcal{A}(L)$ at $x = l$ $(x = r)$. Analogously, the lower corners of T are slightly below $(\lambda - \lceil n/8 \rceil)$-level of $\mathcal{A}(L)$ at $x = l$ $(x = r)$. (If λ is below $n/8$ or above $7n/8$, the trapezoid is unbounded.) In the following whenever we refer to *the trapezoid* then this construction is meant, see also Figure 2.9.

Lemma 2.16 (Trapezoid Construction [49]). *Given n lines L and a slab $S = [l, r]$ that contains at most $\frac{n^2}{100}$ of the vertices of $\mathcal{A}(L)$. Then the trapezoid T defined as above contains the λ-level of $\mathcal{A}(L)$ within S and at most half of the lines from L intersect T.*

Proof. First we will show that at most half the lines intersect T. Note that every line that intersects T does so exactly twice at the boundary of T. The left and right boundary is intersected by exactly $4\lceil n/8 \rceil + 2$ lines.

Consider the top segment σ of T and denote by t the number of lines intersecting σ. We define L_1 as the set of lines crossing σ with slope smaller than the slope of σ and L_2 as the set of lines crossing σ with slope larger than the slope of σ. Whenever a line in L_1 crosses σ the number of lines below σ increases by one and whenever a line in L_2 crosses σ the number of lines below σ decreases by one. As the number of lines below σ at l is the same as the number of lines below σ at r it follows $t/2 := |L_1| = |L_2|$. Every line in L_1 crosses every line in L_2 within S, because one starts below σ and ends above σ and

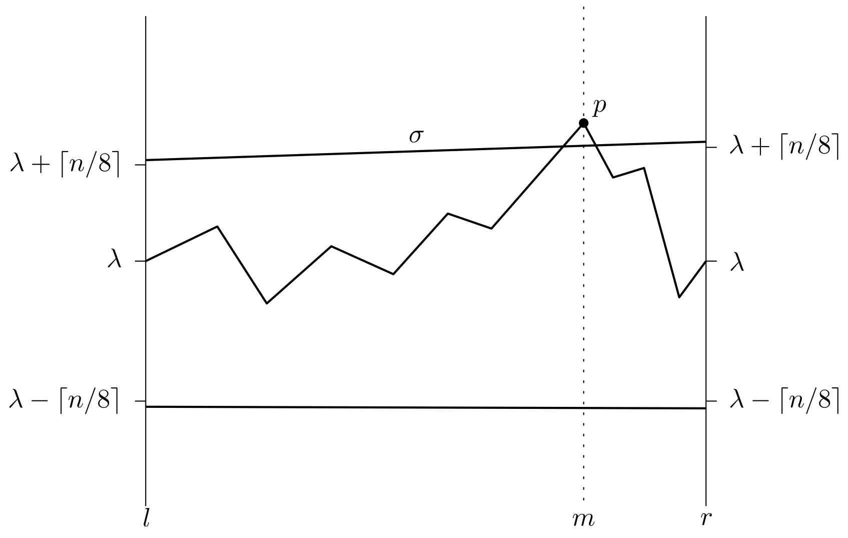

σ
p
$\lambda + \lceil n/8 \rceil$
$\lambda + \lceil n/8 \rceil$
λ
λ
$\lambda - \lceil n/8 \rceil$
$\lambda - \lceil n/8 \rceil$
l
m
r

Figure 2.9: The construction of the trapezoid.

the other starts above σ and ends below σ. Thus we have at least $(t/2)^2$ vertices inside S. Thus

$$(t/2)^2 \le v \le n^2/100.$$

This implies at most $n/5$ lines cross σ. Analogously, at most $n/5$ lines cross the bottom segment of T. Thus in total the number of lines intersecting T is at most

$$\frac{4\lceil n/8 \rceil + 2 + 2n/5}{2} \le n/2,$$

for n large enough.

Now we show that the λ-level is contained in T. Assume for the purpose of contradiction that there exists some point p on the λ-level outside T. Without loss of generality p is above σ. Then there are at most λ lines below σ at this point. However at l and r are $\lambda + \lceil n/8 \rceil$ lines below σ. This implies L_1 and L_2 contain each at least $\lceil n/8 \rceil$ lines — a contradiction. □

2.3.5 Counting Vertices

Our algorithm uses a subroutine to count the vertices of the arrangement within some slab S in the initialization and in Step 2. This is important to construct a trapezoid that contains the λ-level in S as described in Section 2.3.4. In this section we will describe this subroutine and prove its correctness. At the end of this section, we will show how we can perform the initialization step.

How to count the number of vertices in a slab is known using inversion counting [23]. To count inversions in $O(n \log n)$ time is a standard textbook exercise. However, we want to spend only a linear amount of time, in particular for the first iterations of the algorithm. (After $2 \log \log n$ iterations an $O(n \log n)$ algorithm would be also affordable.)

Here we describe in detail an algorithm described briefly in [23]. However, as the algorithm is very simple we cannot exclude that it has been used before by other authors.

Consider Figure 2.10. Given a slab $S = [l, r]$, we would like to know how many vertices of the arrangement are in S. We first label the lines $L = \{\ell_1, \ldots, \ell_n\}$ by ascending slope.

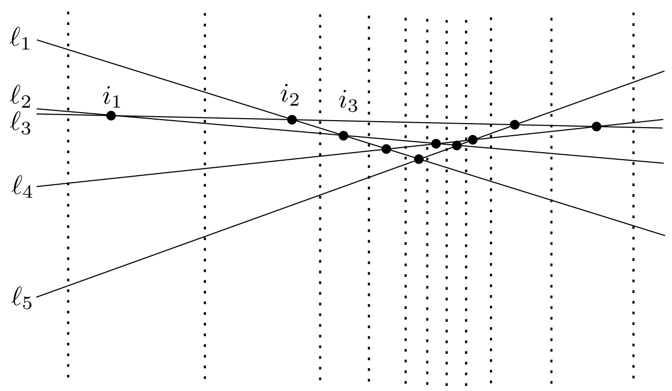


Figure 2.10: The number of vertices between two vertical lines is exactly the number of inversions between the induced permutations.

Now note that each vertical line ℓ induces a permutation of the lines L. For example: a vertical line to the left of all vertices induces the identity permutation. The number of inversions corresponds exactly to the number of vertices to the left of our line ℓ.

Back to our slab $S = [l, r]$. Let ℓ_l, ℓ_r be the two vertical bounding lines of S. Then the number of vertices in S is the difference of the number of inversions induced by the line ℓ_l and ℓ_r.

Lemma 2.17 (Counting Inversions [23])**.** *Fix some constant $c > 0$. Let L be a set of n non-vertical lines and ℓ_l and ℓ_r two vertical lines. Further let v be the number of vertices between these lines.*

We can compute in linear time a number p such that

$$p \leq v \leq p + cn^2.$$

Proof. Let $<_l$ and $<_r$ the order of the lines in L induced by the lines ℓ_l and ℓ_r respectively. (We do not compute this order, as it would require sorting.) We define b groups of size at most $A = \lfloor n/(b-1) \rfloor$, denoted by $G_1, \ldots, G_b$ with the property that $\ell \in G_i$ and $\ell' \in G_j$ implies $\ell <_l \ell'$, if $i < j$. Analogously, we define b groups $F_1, \ldots, F_b$ for the line ℓ_r, with the same properties. The constant b will be chosen later appropriately independent of n.

Note that we can find $b-1$ reference points on ℓ_l and ℓ_r in $O(n \log b)$ time such that we can determine in $O(\log b)$ for a line ℓ to which groups it belongs. We denote by t_{ij} the number of lines in group $G_i \cap F_j$. We can compute all t_{ij} in $O(n \log b + b^2)$ time and store them in a matrix of size $O(b^2)$. The additional b^2 term in the running time stems from the matrix initialization. We define

$$p := \sum_{\substack{i<i' \\ j>j'}} t_{ij} t_{i'j'}.$$

Thus p can be computed in a naive algorithm in $O(b^4)$ time. It is easy to see that p is a lower bound on the number of vertices. We can bound the number of inversions from $<_l$ to $<_r$ that we omitted to count. For any pair of lines that cross in S, and that we omitted to account for in p, these lines must be in the same group G_i or F_i for some i. The number of such pairs is upper bounded by

$$2b\binom{A}{2} \leq 2b\frac{\left(\frac{n}{b-1}\right)^2}{2} \leq \frac{b}{(b-1)^2}n^2.$$

Choosing b large enough, the term $\frac{b}{(b-1)^2}$ becomes smaller than c. The total running time is $O(n \log b + n \log b + b^4)$. This is linear in n. $\square$

The Initialization Recall in the initialization of the algorithm the slab $S = [0, 1]$ is subdivided into 100 slabs of equal width, and no line is discarded. One of the slabs contains at most

$$v \leq \frac{1}{100(1/2 + \varepsilon)^2} n^2$$

vertices. We are now ready to explain how to find this slab. We compute the approximate number of vertices p in each slab according to Lemma 2.17 with $c = 1/200$. To see that this works, note that there exists a slab with $v \leq n^2/200$ by the pigeonhole principle. According to Lemma 2.17 there exists at least one $\bar{p}$ with $\bar{p} \leq n^2/200$. Denote by $\bar{v}$ the number of vertices of the corresponding slab. By Lemma 2.17, it holds $\bar{v} \leq \bar{p} + n^2/200 \leq n^2/100 \leq \frac{1}{100(1/2+\varepsilon)^2} n^2$ as desired.

2.3.6 Bounding the Number of Vertices

In Section 2.3.4 we have seen why the Vertex Condition (2.3) is necessary, that is, the number of intersection in the slab of consideration is below $\frac{1}{100(1/2+\varepsilon)^2} n^2$. In this section we will explain how this is achieved. Simultaneously we will show that the algorithm maintains the Vertex Invariant (VI), that is, the number of vertices in the slab S is at the start of each iteration below $\frac{n^2}{(1/2)100}$. One of the major ideas is to use the bound on the number of vertices of the previous iteration.

Before we indulge ourselves in tedious calculations, consider an over-simplified scenario where we do not care about running times and are able to discard exactly half of the lines. The Vertex Invariant (VI) would simplify to $v \leq \frac{n^2}{25}$ and after splitting our slab into 4 subslabs one of them would contain at most $\overline{v} \leq \frac{n^2}{100}$ vertices. This implies the Vertex Condition (2.3) is satisfied. Once we discard half of the lines the number of vertices might not decrease, but it will not increase either. The number of lines thereafter equals $\overline{n} = n/2$ and we have

$$\overline{v} \leq v \leq \frac{n^2}{100} = \frac{(2\overline{n})^2}{100} = \frac{\overline{n}^2}{25}.$$

This would imply the Vertex Invariant (VI) is indeed satisfied.

We consider now the non-simplified setting. By Vertex-Invariant (VI), we know the number of vertices v of our input slab S satisfies at the start of the iteration

$$v \leq \frac{n^2}{100(1/2 + \varepsilon)^2}.$$

Since v is a natural number, we can even conclude that $v \leq \left\lfloor \frac{n^2}{100(1/2+\varepsilon)^2} \right\rfloor$, but we do not need this stronger inequality. After splitting S into $S_1, \ldots, S_4$, we compute the approximate number of vertices p_1, p_2, p_3, p_4 as in Lemma 2.17, for each Slab. We have $p_1 + p_2 + p_3 + p_4 \leq v$ and thus there exists a slab S_i with $p_i \leq v/4$ and the number of vertices $\overline{v}$ of this slab is bounded by

$$\overline{v} \leq p_i + \frac{\varepsilon n^2}{100} \leq v/4 + \frac{\varepsilon}{100} n^2 \leq \left(\frac{1}{400(1/2 + \varepsilon)^2} + \frac{\varepsilon}{100} \right) n^2.$$

We will show indeed

$$\overline{v} \leq \frac{1}{100} n^2.$$

It is sufficient to show

$$\frac{1}{400(1/2+\varepsilon)^2} + \frac{\varepsilon}{100} \leq \frac{1}{100}.$$

And indeed it holds

$$\begin{aligned}
\frac{1}{400(1/2+\varepsilon)^2} + \frac{\varepsilon}{100} &= \frac{1}{100}\left(\frac{1}{(1+2\varepsilon)^2} + \varepsilon\right) \\
&= \frac{1}{100}\left(\frac{1+\varepsilon+4\varepsilon^2+4\varepsilon^3}{1+4\varepsilon+4\varepsilon^2}\right) \\
&\leq \frac{1}{100}.
\end{aligned}$$

Thus the Vertex Condition (2.3) is satisfied and we can apply Lemma 2.16 in Step 4.

After the iteration $\overline{n} = \lceil (\frac{1}{2}+\varepsilon)n \rceil \geq (\frac{1}{2}+\varepsilon)n$ lines remain and also the Vertex Invariant (VI) is satisfied, that is,

$$\overline{v} \leq \frac{n^2}{100} \leq \frac{\overline{n}^2}{100 \cdot (\frac{1}{2}+\varepsilon)^2}.$$

Here is the point where we need that we discard *exactly* $\lfloor (\frac{1}{2}-\varepsilon)n \rfloor$ lines. If the algorithm would discard more lines it might have the trouble of having too many vertices compared with the number of lines. This is also where our analysis is suboptimal and might be improved. It seems likely that at least some constant fraction of the vertices in S are removed in Step 6, when almost half of the lines are discarded. However, we do not know how to exclude the case that all vertices discarded in this way are outside of S.

2.3.7 Bounding the Intersections with the 1-tube

In this section, we will show that the number of lines intersecting τ within C_γ, but not T within S is bounded from above. We start by clarifying the notation for this section.

Assume we are at the beginning of some iteration and we denote with n the number of the current set of lines L. In Step 5 we have a slab denoted by S with width $w \leq 1$ and $\mathrm{C}_\gamma = [l+\gamma w, r-\gamma w] = [l', r']$ the γ-core of S. The trapezoid of S is denoted by T, it is described in detail in Section 2.3.4. Here we make use of no property of T except its geometric shape: being a trapezoid with two sides on the vertical boundary of the slab S containing it, see Figure 2.13 on page 39.

As in the current iteration at Step 5 our slab has been already split into four slabs, the Width Condition (WC) becomes the *adjusted* Width Condition:

$$w \geq \frac{1}{4} \cdot \frac{400}{\gamma^2\varepsilon^2} \frac{\log n}{n} = \frac{100}{\gamma^2\varepsilon^2} \frac{\log n}{n}. \tag{AWC}$$

Before we proceed note that the adjusted Width Condition implies

$$nw \geq \frac{100}{\gamma^2\varepsilon^2} \log n \geq \frac{100}{\gamma^2\varepsilon^2} \geq 62500$$

as $\gamma, \varepsilon \leq 1/5$. Since $w \leq 1$, we also get

$$n \geq 62500.$$

We will repeatedly use these bounds in this section.

Lemma 2.18. *Given the following:*

- $\gamma, \varepsilon \in (0, 1/5)$*;*
- *a set of* n *lines* L*;*
- *a slab* $S = [l, r]$ *of width* $w \leq 1$ *with trapezoid* T *and* 1*-tube* τ *as described in Section 2.3.1 and* $\mathrm{C}_\gamma = [l + \gamma w, r - \gamma w] = [l', r']$ *the* γ*-core of* S*.*

We assume the adjusted Width Condition (AWC)

$$w \geq \frac{100}{\gamma^2 \varepsilon^2} \, \frac{\log n}{n} \, .$$

By n' *we denote the number of lines intersecting* τ *within* C_γ *but not* T *within* S*. Then it holds*

$$n' \leq \varepsilon n.$$

Before we consider the γ-core, we will study τ and the way lines are intersecting it further. We will use an averaging argument: while τ may be intersected by all lines from L, on average the number of intersecting lines is sublinear. The second crucial observation is that a line ℓ can intersect τ only in certain ways, depicted in Figure 2.13. This will imply that the total number of lines intersecting the γ-core C_γ is small. Also note that our averaging argument is in the primal.

We define with a and b the upper and lower bounding segments of T. We want to bound the number of lines intersecting the 1-tube of those segments. Lemma 2.19 gives a good bound for general segments.

For a segment $s = [(l, b), (r, t)]$ we define a function g_s by setting $g_s(x)$ to be the number of lines that intersect the 1-tube of s at $x \in [l, r]$. We omit the subscript s, if it is clear from the context which segment s is meant. The following lemma provides an upper bound on the average number of such lines.

Lemma 2.19. *Given* n *lines in the plane and a non-vertical segment* $s = [(l, b), (r, t)]$ *of horizontal width* $w = r - l$*, such that the adjusted Width Condition* (AWC) *is satisfied. We have*

$$\int_l^r g_s(x) \, \mathrm{d}x \leq 5\sqrt{nw \ln(nw)} \, .$$

(We assume $0 \leq l < r \leq 1$*.)*

Proof. We follow the approach of Alon et al. [6] but are more specific about some technical details. We define $\overline{x}_i := l + \frac{i}{k} w$ for $i = 0, \ldots, k-1$. The number k will be specified later. Consider the function

$$f(x) := \sum_{i=0}^{k-1} g(\overline{x}_i + x)$$

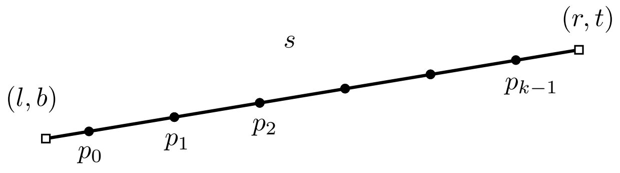


Figure 2.11: Segment s and some equidistant points on it.

over the domain $[0, w/k]$. Clearly, we have

$$\int_0^{w/k} f(x)\,\mathrm{d}x = \int_l^r g(x)\,\mathrm{d}x\,.$$

Next we bound $f(x)$ for some arbitrary but fixed x in the domain of f. We define some set of points $p_i = (x_i, y_i)$ for $i = 0, \ldots, k-1$ on s with

$$x_i \;:=\; \overline{x}_i + x \;=\; l + \frac{i}{k}w + x,$$

see Figure 2.11. Consider the set of lines $P^* = \{p_0^*, \ldots, p_{k-1}^*\}$. Let D_i denote the set of disks from $\mathcal{D}$ that are intersected by the line p_i^*. The value of f is the number of pairs $(d, p_i^*) \in \mathcal{D} \times P^*$ where $d \cap p_i^* \neq \emptyset$. A (generous) upper bound for this quantity is provided by

$$n + \sum_{i<j} |D_i \cap D_j|\,,$$

where the first term counts every disk that intersects only one line and the second term counts every disk that is intersected by at least two lines. (In this way, a disk that is intersected by c lines is counted $1 + \binom{c}{2}$ times.)

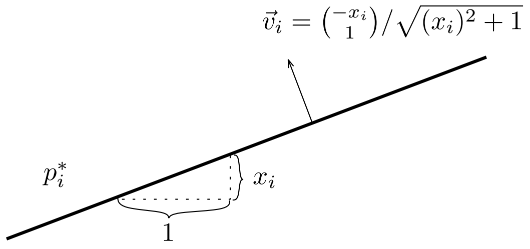


Figure 2.12: Line p_i^* and its normal vector $\vec{v}_i$.

Let $\vec{v}_i = (-x_i, 1)^T/\sqrt{(x_i)^2 + 1}$ be the (unit) normal vector to p_i^*. By Lemma 2.7 (where $d = 2$, $w_1 = w_2 = 2$, $\vec{v}_1 = \vec{v}_i$, and $\vec{v}_2 = \vec{v}_j$) we have (using $x_t \leq 1, \forall t$)

$$|D_i \cap D_j| \leq \frac{16}{\pi\,|\det(\vec{v}_1, \vec{v}_2)|} = \frac{16\sqrt{(x_i)^2+1}\sqrt{(x_j)^2+1}}{\pi\,|x_i - x_j|}$$

$$\leq \frac{32}{\pi\,|x_i - x_j|} = \frac{32k}{\pi w\,|i-j|}$$

and therefore

$$n + \sum_{i<j} |D_i \cap D_j| \leq n + \frac{32k}{\pi w} \sum_{i<j} \frac{1}{j-i}\,.$$

(Note this is the only place we used that $0 \leq l, r \leq 1$.) The sum can be bounded using

$$\sum_{i<j} \frac{1}{j-i} = \sum_{a=1}^{k-1} \frac{k-a}{a} = kH_{k-1} - (k-1) < 1 + k\ln(k)\,,$$

where the last inequality uses the well-known bound $H_n < 1 + \ln(n)$ for the harmonic number. We started out by fixing a particular x, but the derived bound holds for any arbitrary x. Altogether we obtain

$$f(x) < n + \frac{32k}{\pi w}(1 + k\ln k)$$

and so

$$\int_l^r g(x)\,\mathrm{d}x = \int_0^{w/k} f(x)\,\mathrm{d}x < \frac{nw}{k} + \frac{32}{\pi}(1 + k\ln k) < \frac{nw}{k} + 11k\ln k. \tag{INT}$$

The last inequality holds provided k is larger than $\frac{32/\pi}{11-(32/\pi)} \approx 12,59...$ We will show that k is always at least 120.

Before we continue our calculations, let us adress the issue of rounding. Denote the optimal value of k with $z \in \mathbb{R}$, to be specified soon. Unfortunately, we cannot use z directly, as k has to be a natural number. We set $k = \lceil z \rceil$ and

$$z = \sqrt{\frac{nw}{d^2 \ln nw}},$$

with $d = \sqrt{6}$.

We show that the adjusted Width Condition AWC implies $k \geq z \geq 120$ as follows:

$$\begin{aligned}
z &= \sqrt{\frac{nw}{d^2 \ln nw}} && (2.4)\\
&= \sqrt{\frac{\log e}{d^2}\frac{nw}{\log nw}} && (2.5)\\
&= \sqrt{\frac{\log e}{d^2}\frac{100}{\gamma^2\varepsilon^2}} && (2.6)\\
&\geq \sqrt{\frac{1.44}{6}\cdot 62500} && (2.7)\\
&\geq 120. && (2.8)
\end{aligned}$$

Note we used the adjusted Width Condition (AWC) in line (2.6).

With this lower bound on z we can refine the upper bound on (INT) as follows:

$$\begin{aligned}
\frac{nw}{\lceil z \rceil} + 11\lceil z \rceil \ln\lceil z \rceil &< \frac{nw}{z} + 11(z+1)\ln(z+1) && (2.9)\\
&= \frac{nw}{z} + 11z\ln(z+1) + \ln(z+1) && (2.10)\\
&< \frac{nw}{z} + 11z(\ln z + 1/100) + z/10 && (2.11)\\
&= \frac{nw}{z} + 11z\ln z + 11z/100 + z/10 && (2.12)\\
&< \frac{nw}{z} + 12z\ln z. && (2.13)
\end{aligned}$$

To get Equation (2.11) note that the natural logarithm $\ln x$ is strictly concave and its derivative is $1/x$, thus $\ln(x+1) - \ln x < \frac{1}{x} < 1/100$, for $x > 100$.

Using our choice for z in the previous expression (2.13), the first term becomes

$$\frac{nw}{z} = d\sqrt{nw \ln nw}\,.$$

The second factor of the second term becomes

$$\begin{aligned}\ln z &= \ln\left(\sqrt{nw/d^2 \ln nw}\right)\\ &= \frac{1}{2}\ln\left(nw/d^2 \ln nw\right)\\ &< \frac{1}{2}\ln nw\,.\end{aligned}$$

The last inequality holds is satisfied already for $nw > 3$. The second term is upper bounded by

$$\begin{aligned}12\ z\ln z &< 2d^2\sqrt{\frac{nw}{d^2 \ln nw}}\ \frac{1}{2}\ln nw\\ &= d\sqrt{nw \ln nw}\end{aligned}$$

Thus the sum is upper bounded by

$$2d\sqrt{nw \ln nw} < 5\sqrt{nw \ln nw}\,.$$

$(2d = \sqrt{24} < \sqrt{25} = 5.)$ □

We are now ready to finish the proof of Lemma 2.18, that is we have to show $n' \leq \varepsilon n$. Recall n' equals the number of lines intersecting τ within C_γ, but not T in S.

Bounding the integral is not sufficient to bound the number of lines that intersect the 1-tube, because lines that do so for a very short interval only do not contribute much to the integral. In principle the number of lines that intersect the 1-tube at some specific x could be linear.

Observation 2.20. *Every line ℓ that intersects τ within C_γ and does not intersect T intersects the 1-tube of a or b over an interval of length at least γw.*

Every such line intersects τ as g_3 in Figure 2.13. The way g_1 intersects τ is not counted, because g_1 also intersects T. The way g_2 intersects τ is not possible, because of the convex-concave nature of τ. Here is the crucial point that we used convexity of the 1-tube. We also used it to show that counting the number of lines intersecting τ is easy. From this observation follows directly

$$\gamma w n' \leq \int_l^r (g_a(x) + g_b(x)) \leq 2 \cdot 5\sqrt{nw \ln nw} \leq 10\sqrt{nw \ln n}\,.$$

This implies

$$n' \leq \frac{10\sqrt{nw \ln n}}{\gamma w} < \frac{10\sqrt{nw \log n}}{\gamma w},$$

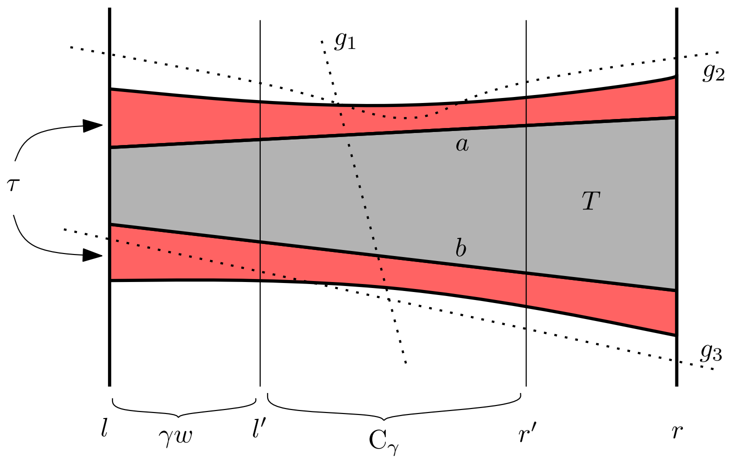


Figure 2.13: Slab S is the region between the two bold vertical lines. Its core C_γ is the region between the two lines between the two bold lines. The corresponding trapezoid T is displayed in gray and its 1-tube τ in light red. The line g_1 intersects both T and τ, whereas g_2 and g_3 intersect τ but not T. Every such line intersects τ at the boundary of the core, like g_3 does. An intersection pattern as depicted for g_2 is impossible for a straight line.

and thus to show $n' \leq \varepsilon n$, it suffices to show

$$\frac{10\sqrt{nw\log n}}{\gamma w} \leq \varepsilon n.$$

However, this is equivalent to

$$\frac{100\log n}{\gamma^2\varepsilon^2 n} \leq w,$$

which is the adjusted Width Condition (AWC) as stated in Lemma 2.18.

2.4 Conclusions

In this chapter we studied the construction of separators for balls in *deterministic* linear time. The aim is to intersect as few balls as possible while (approximately) bisecting the set of midpoints. We presented essentially two ways to compute such seperators with a sublinear number of intersections. The first algorithm is very simple and straight-forward to implement (we gave all constants explicitly), and obtains an arbitrarily good bisection in combination with an asymptotically optimal number of intersections. The strength of the second algorithm is to bisect the center points *exactly*, but it works in the plane only.

Throughout the chapter, we assumed the balls to be disjoint, we used this property only in Lemma 2.7 to bound the number of balls completely contained in some set C, by the volume of C divided by the volume of a ball. In fact, both algorithms work as long as we have some density upper bound on the objects under consideration and some bound on the size of the objects. This upper bound is implicitly given if for instance the objects satisfy some fatness condition and are disjoint or have constant ply.

Acknowledgments. We want to thank Marek Elias, Jiří Matoušek, Edgardo Roldán-Pensado and Zuzana Safernová for interesting discussions on the conjecture for higher dimensions and referring us to related work. Special thanks goes to Günter Rote for a thorough reading and many useful comments.

Chapter 3

Tron

3.1 Introduction

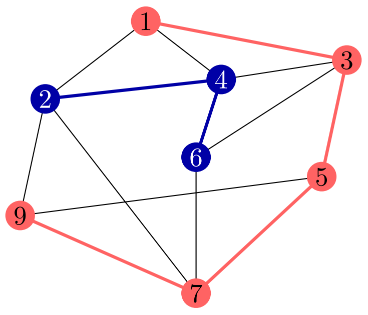


Figure 3.1: Alice receives a score of 5 and Bob receives a score of 3, thus Alice wins.

Definition. Given a graph $G = (V, E)$ two players Alice and Bob take turns alternatingly in the following manner: in the first turn each player chooses a start vertex in G (not both the same): thereafter, the players whose turn it is picks a vertex that is adjacent to his or her last chosen vertex. It is not allowed to pick a vertex that has been chosen by either of the players before at any stage of the game. Thus the vertices get used and cannot be reused. When one player cannot move, the other player continues alone. The game ends when both players cannot move. Each player gets a score equal to the number of vertices picked before they could not move anymore. See Figure 3.1 for an example. Note that each player form a traverses.

We denote the score for Alice and Bob as $\#\mathtt{A}$ and $\#\mathtt{B}$. We say Alice *wins* if $\#\mathtt{A} > \#\mathtt{B}$; Bob *wins* if $\#\mathtt{B} > \#\mathtt{A}$; and otherwise we call a play of a game a *tie*. The outcome of a play is defined as $\#\mathtt{B}/\#\mathtt{A}$. We say that the players play *rational*ly if both try to optimize the worst case outcome, i.e., Alice minimizes and Bob maximizes the worst case outcome.

One could also consider the difference instead of the ratio of $\#\mathtt{A}$ and $\#\mathtt{B}$. We will see that this would lead to less interesting results and strategies.

Here, we differ slightly from [19] where Alice loses if both players receive the same amount of vertices. We introduce this technical nuance because it makes more sense in regard of the extremal question and is not relevant for the complexity question.

The extremal question is by how much can ALICE win in comparison to BOB and vice versa. The complexity question asks for the computational complexity class of determining if ALICE has a winning strategy.

The game must end ultimately when all vertices are eaten up. And of course the game is not *loopy*, as it is impossible to return to a previous state of the game. A game is called loopy if it is possible to return to a previous state of the game. We assume that both players have perfect information at all times. The game is defined as *normal play*, that is the players try to move as long as possible, in the sense made precise above. It is also possible to consider the variant of *misère game*, where the players try to be unable to move as soon as possible. We further distinguish whether start vertices are given and whether the graph is directed or undirected.

A problem L is PSPACE-complete if every decision problem, that can be decided with polynomial amount of space can be reduced to L and L can be decided with polynomial amount of space.

The decision problem TRON is the following: the input is a graph and we want to answer the question whether ALICE has a winning strategy. We consider the decision problem under different game modes as described above.

For convenience, we define the length of a path to be the number of vertices.

Franchise. Tron is a 1982 American science fiction movie. It was directed by Steven Lisberger and produced by Disney Studios. After its release a whole cult originated from the movie: several books, comics, a TV series and computer games emerged as sequels. The movie Tron Legacy released in 2010 produced an income of over \$ 400 million during its theatrical run [69]. After the first movie several implementations of computer games

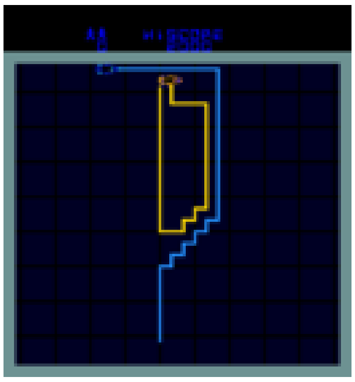


Figure 3.2: Screenshot of TRON played for a short time.

with the same name appeared. Many different game modes exist. Some are completely unrelated to the game we consider. Nevertheless, the name TRON is usually associated to the following game: two light cycles[1] travel within a rectangle; both leave a glowing wall behind them. A player loses when he or she hits one of the walls or the boundary of the rectangle.

Related Work. Much work has been done to analyze games from a mathematical or computer science perspective. A survey can be found by Demaine and Hearn [27].

[1]light cycles are fancy motorcycles.

TRON has been first studied explicitly by Bodlaender [19] and later by Bodlaender and Kloks [20]. PSPACE-completeness was shown for directed graphs under normal play with and without start vertices given by Bodlaender [19]. NP-hardness and coNP-hardness was shown for undirected graphs by Bodlaender and Kloks [20]. There the focus is on studying games where the players form a path during the game play. Among the path-forming games consider the closely related game GEOGRAPHY. On the one hand, it is a textbook example for a PSPACE-complete game on *directed* graphs [66]. On the other hand, it is an example of a game that is polynomially decidable for *undirected* graphs [30]. (this is to decide if ALICE has a winning strategy.) It is applied to show that the ancient game GO is PSPACE-hard [48].

TRON can also be seen as a variation of the traveling salesman problem. Here two players try to visit as many cities as possible, in the presence of a business competitor [29]. Fekete, Fleischer and Fraenkel consider the case that allows to reuse vertices; players can even occupy the same vertex. This game is also PSPACE-complete.

Instead of the algorithmic complexity often games are analyzed more from the perspective by how much a player can win. TRON can be regarded as the game played by two siblings that share some cake (Cut up in a way that the pieces form a graph.) according to the rules above. Knauer, Micek and Ueckerdt answered this question for a related game concerning pizzas (How to eat 4/9 of a pizza [46]). The same question was solved with different methods by Cibulka, Kynčl, Mészáros, Stolař and Valtr [22].

Results We investigate by how much ALICE or BOB can win at most. There exists graphs of arbitrary size such that ALICE can gather all the vertices except a constant number. The same holds for BOB. And the graph can be chosen to be k-connected for any fixed k. For planar graphs, we achieve a weaker, but similar result.

We consider the same question for the misère game and found planar graphs of arbitrary size where BOB kills himself within 2 turns and ALICE has to take all the remaining vertices. Conversely, the same holds for ALICE.

We show PSPACE-completeness for TRON played on *undirected* graphs both when starting positions are given and when they are not given. We also consider misère game with the same constraints. In total all 8 different variations are considered.

Outline Section 3.2 is concerned with elementary results and some general observations used later. Section 3.3 explains all the extremal results mentioned above. At last, Section 3.4 gives detailed proofs of the complexity results.

3.2 Basic Observations

The aim of this section is to show some basic characteristics.

When humans play a few games of TRON on a "random" graph, one observes that one usually ends up in a tie or one finds that one of the players made a mistake during the game, i.e., played suboptimally. A natural first question to ask is if ALICE or BOB can win by more than one at all. The following easy example is counterintuitive because it is disconnected.

Example 3.1 (two disjoint paths)**.** *Let G be a graph which consists of two disjoint paths of length n, refer to Figure 3.3. Without loss of generality:* ALICE *starts in the upper graph with $b-1$ vertices to the left and a vertices to the right and $b \leq a+1$. If* BOB

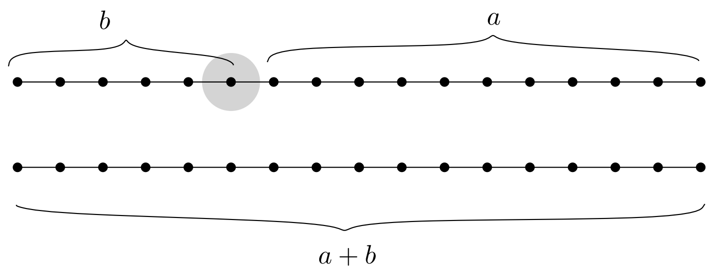


Figure 3.3: Alice has to minimize a/b and $(a+b)/(a+1)$ simultaneously.

starts next to Alice *the outcome will be a/b. If* Bob *starts at the beginning of the other path, the outcome will be $(a+b)/(a+1)$).*

Thus Alice *has to minimize the two possible outcomes simultaneously, which is attained when $(a+b)/(a+1) = a/b$. This is the definition of the golden ratio(up to the constant 1, which can be ignored for large a and b) and hence the outcome, under rational play, equals $\frac{1+\sqrt{5}}{2}$, with $n \to \infty$.* □

Example 3.1 shows that $\#\mathtt{B} - \#\mathtt{A}$ can get arbitrarily large. If we defined the outcome to be the difference of the score, Alice would choose $a \approx n/3$.

We modify the graph above by adding a super vertex v adjacent to every vertex of G. When Alice starts at v, the first vertex on the original graph G will be taken by Bob and Alice will take the second vertex on G. The roles of Alice and Bob have interchanged. The argument holds in general and implies the following lemma.

Lemma 3.2 (Super vertex)**.** *Let G be a graph where* Bob *has a winning strategy and let F be the graph we obtain by adding a super vertex v adjacent to every vertex of G. It follows that* Alice *has now a winning strategy and can achieve an even better ratio.*

Note this observation works not necessarily in the misère game. As Alice has one more vertex, which might make the difference between losing and winning. Also note that some graph properties might get lost.

The other direction holds as well. Let G be a graph where Alice wins and say she starts at vertex v. Delete v from G to attain H. The situation in Alice's first turn in H is the same as Bob's first move in G. Conversely, Bob's first turn in H includes the options Alice had in her second move in G.

Lemma 3.3. *Let G be a graph where* Alice *has a winning strategy and H be the graph we obtain by deleting the vertex where* Alice *starts in an optimal play. Then it follows that* Bob *has a strategy in H to obtain the inverse ratio, except that he misses one vertex.*

The start vertex of Alice in G need not be unique, even when Alice wins: consider the complete graph with an odd number of vertices. As we will later see, it is probably impossible to efficiently compute the optimal start vertex in G. However, Lemma 3.3 is about the existence of H.

In comparison to G the roles of Alice and Bob are exchanged in H. Bob has even more options in his first turn in H than Alice had in her first turn in G. Nevertheless Bob has one vertex less he can occupy.

In the misère game this trick does not help because more options can be a disadvantage in the misère game.

Bodlaender and Kloks [20, Theorem 3.1] showed the first inequality of Lemma 3.4.

Lemma 3.4 (Trees). *Let T be a tree. Then $\#\mathtt{A} \leq \#\mathtt{B} + 1$ and $\#\mathtt{B} \leq 2 \cdot \#\mathtt{A}$.*

Proof. We describe a strategy for Alice and Bob explicitly.

Let v denote the start vertex of Alice with neighbors $w_1, \dots, w_k$ and l_i the length of the longest path from w_i in $T - \{v\}$ for all $i = 1, \dots, k$. The index j is defined by $j \in \operatorname{argmax}\{\ l_i \mid i = 1, \dots, k\}$. If Bob chooses w_j as start vertex then he obtains at least l_j vertices, while Alice receives at most $(l_j + 1)$ vertices. This shows the first inequality.

For the second inequality, let Alice start in the middle of a longest path. Thus she divides the tree into smaller trees $T_1, \dots, T_k$. Bob starts in one of them and receives at most as many vertices as the length of the longest path in T. Alice will enter a different tree, which contains one half of the longest path. Thus she receives at least half of the longest path. □

The first inequality is tight for paths of odd length. The second inequality is most likely not tight.

However, Bob can achieve in some trees a *linear* amount of vertices more than Alice, as shown in the next example. To simplify the calculations we assume that Bob wants to maximize the $\#\mathtt{B} - \#\mathtt{A}$. We will argue $\#\mathtt{B} - \#\mathtt{A} \geq k/3$ and this implies also that the ratio $\#\mathtt{B}/\#\mathtt{A}$ is larger than 1 even under the assumption that the players want to optimize the difference.

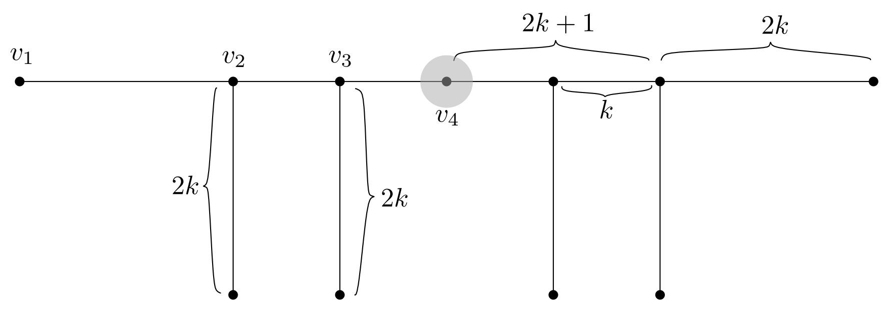


Figure 3.4: Bob can attain more vertices than Alice.

Example 3.5. *The example is depicted in Figure 3.4 schematically. It is symmetric around the marked vertex v_4.*

If Alice *starts more than distance k from v_4 then* Bob *can easily attain at least k more vertices than* Alice*. Thus we assume* Alice *starts x vertices to the right of v_4, with $x \in \{0, 1, \dots, k\}$.*

We describe explicitly a strategy for Bob*. (We will ignore constants.)* Bob *has two different options.*

Option 1*:* Bob *starts on the adjacent vertex to the left of her and receives $x + 4k$ vertices and* Alice *receives $4k - x$ many vertices. In this case $\#\mathtt{B} - \#\mathtt{A} = 2x$.*

Option 2*:* Bob *starts in v_1. In this case* Alice *has two options. Either, she goes towards him and he will get $4k$ vertices, while she gets only $3k + x$ vertices. Or she goes to the right and will receive $4k - x$, while* Bob *gets $5k$ vertices by using the branch at v_3. In both cases $\#\mathtt{B} - \#\mathtt{A} \geq k - x$.*

As Bob *chooses his strategy and* Alice *chooses x the outcome equals*

$$\min_{x \in \{0,1,\dots,k\}} \max \{\ 2x,\ k - x\ \} \approx k/3.$$

3.3 Extremal Question

In this section, we answer the extremal question for TRON, i.e., is there a nontrivial upper and lower bound on the outcome as a function of the number of vertices of G. For the misère game the question is the same, just with the difference that the players try to get as few vertices as possible. Note that it makes no sense to consider variants of the game with given start vertices. Further note that a directed cycle is an example of a directed graph, where BOB receives all but one vertex. Thus in the remainder of this section we only consider undirected graphs without given start vertices.

Subsection 3.3.1 deals with the misère game.

For normal play we investigated several graph classes and the answer depends on the class of graphs we consider. Subsection 3.3.2 deals with general graphs.

The question is again by how much can ALICE win over BOB and vice versa. In particular one would expect any non-constant lower bound how much each player receives. Very surprisingly for every natural number n there exists a graph with n vertices, such that $\#\mathtt{B} = n - c$ for some small constant c.

As corollary, Lemma 3.2 gives us a graph where ALICE can obtain all vertices except a constant amount.

In Subsection 3.3.3 we will construct planar graphs with $\#\mathtt{B}/\#\mathtt{A} = \Theta(\sqrt{n})$ and graphs with $\#\mathtt{A}/\#\mathtt{B} = \Theta(\sqrt{n})$.

In Subsection 3.3.4 k-connected graphs are studied. Surprisingly, we are able to adapt the example fron Subsection 3.3.2 such that it becomes arbitrarily highly connected.

3.3.1 Misère Game

We remind the reader that each player tries to be unable to move in the misère game. Nevertheless, each player has to make a turn if there exists a free adjacent vertex.

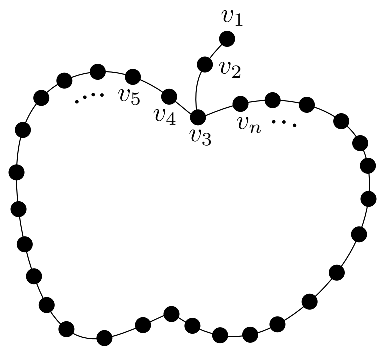


Figure 3.5: ALICE has to collect all but 2 vertices in the graph, called apple.

Theorem 3.6 (extremal misère game). *For every natural number n and each player p there exists a graph on n vertices such that p collects at most 3 vertices and the other player has to collect all the remaining vertices.*

Proof. We first give a proof for BOB. Refer to Figure 3.5. The construction, called *apple*, is a path $P = (v_1, \ldots, v_n)$ with v_3 connected to v_n. We describe explicitly every possible turn of ALICE and BOB's response to them.

Case 1 Alice starts at v_1 the top vertex of the stem of the apple. Consequently, Bob starts on the peel at v_5 and goes towards the stem such that Alice will cut him off and she has to traverse the rest of the graph.

Case 2 Alice starts at v_2 in the middle of the stem. Accordingly, Bob starts at v_1 and Alice has to traverse the rest of the graph.

Case 3 Alice starts somewhere on the peel. Bob starts at v_2 in the middle of the stem and shuts himself out. Thus Alice is alone on the cycle and has to traverse all the remaining vertices.

To see that the theorem also holds for Alice add a super vertex v, which is connected to every other vertex. When Alice starts there the roles of Alice and Bob are interchanged as discussed after Lemma 3.2 □

3.3.2 The normal game on general graphs

We study a simple example first.

Example 3.7 (big circle)**.** *We consider a cycle of length n, and a subtle change of the rules. We assume that* Alice *has to make two moves before* Bob *joins the game. The analysis of this example is short:* Alice *decides for a vertex and a direction and* Bob *can simply start in front of her and take the rest of the cycle. This example will also work, if only every a^{th} vertex is an admissible start vertex for* Bob. □

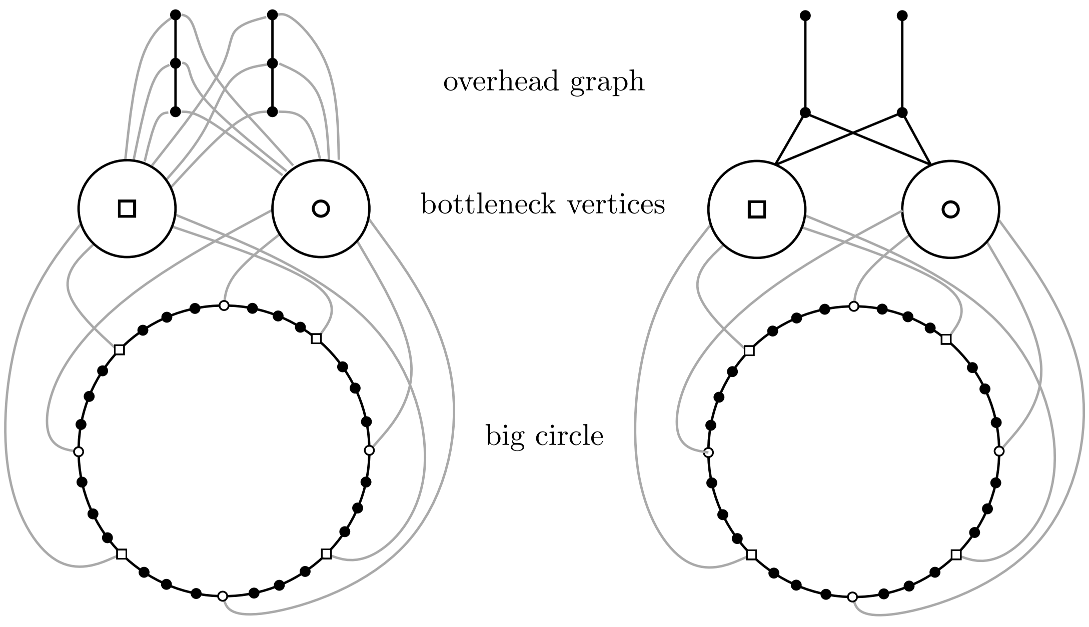


Figure 3.6: The ordinary visage on the left-hand side and the visage from Torsten Ueckerdt on the right-hand side.

Theorem 3.8 (general extremal graphs)**.** *For every natural number n there exists an undirected graph G on n vertices such that* Bob *can traverse all but a constant number of vertices.*

Proof. For n smaller than some constant the statement is trivial.

The construction, called *visage*, consists of three parts: an overhead graph, a big circle, and a bottleneck, as depicted on the left-hand side of Figure 3.6. The overhead graph can be any graph where Bob wins. The two paths in Example 3.1 give us such a graph. It suffices to take paths of length 3 for our purpose. The big circle consists of a large cycle of length $2al$. The constant a can be chosen to be 4. However, we recommend to think of a as a *large enough* constant on a first reading. In contrast, l grows with n.

The last part is a bottleneck between the first two parts and consists of two non adjacent vertices, one is marked with a square and the other with a circle in Figure 3.6. Both bottleneck vertices are adjacent to every vertex of the overhead graph but only to every a^{th} vertex of the cycle. More specifically, we alternatingly mark every a^{th} vertex on the big circle with a square or a circle. The bottleneck vertices are adjacent to the vertices on the big circle marked with the same symbol.

When either player enters the big circle via one bottleneck vertex, it can only leave the big circle via the other. But the next bottleneck vertex is at some distance of a vertices, when he or she is on the big circle. However, the overhead is *adjacent* to both bottleneck vertices.

We give a strategy for Bob for all possible moves of Alice to gather at least $n - 10$ vertices.

Case 1 Alice starts in the overhead graph. In this case, Bob also starts in the overhead graph and wins within the overhead graph, by construction of the overhead graph. Thereafter, Alice has to leave the overhead graph eventually and go to one of the bottleneck vertices. Bob waits one more turn within the overhead graph. If Alice tries to go back to the overhead graph, Bob will go to the other bottleneck vertex and trap her. Thus Alice has to go to the big circle and once there she will have made already two turns when Bob enters the big circle. We already studied this situation explicitly in Example 3.7.

Case 2 Alice starts in one of the bottleneck vertices. Bob will then again start somewhere in the overhead graph. The situation is as in Case 1.

Case 3 Alice starts in the big circle. In this case, Bob will start on the closest bottleneck vertex to Alice and then quickly go to the other bottleneck vertex via the overhead graph. Thus she cannot leave the big circle. Finally, he enters the big circle on the closest vertex in front of her. □

Example 3.9 (Torsten-visage)**.** *Torsten Ueckerdt showed with a very similar construction how to reduce c to 8, as depicted on the right hand side of Figure 3.6. Here not every vertex of the overhead graph is connected to the bottleneck. The crossing number is two: the graph is almost planar. We omit a detailed analysis of all the possible turns of* Alice.

3.3.3 Planar Graphs

We construct a planar visage. Again, it will be more convenient to study a simpler example first.

Example 3.10 (long path)**.** *Consider a path of length n and the change of rules that* Alice *has to make two moves before* Bob *starts. Assume* Alice *goes along the path and has a vertices she could reach. If* Bob *cuts her off, $\#B = a$, $\#A = 2$, and the* *outcome equals $\#B/\#A = \Theta(a)$. Hence* ALICE *could choose a to be small. In this case* BOB *chooses the other side and $\#B = n - a - 2$, $\#A = a + 2$, and the outcome becomes $\#B/\#A = \Theta(n/a)$. This observation implies that* ALICE*'s best choice is $a = \Theta(\sqrt{n})$ and thus the outcome under optimal play is $\Theta(\sqrt{n})$. This example works asymptotically the same way if only every 100th vertex is an admissible start vertex for* BOB. □

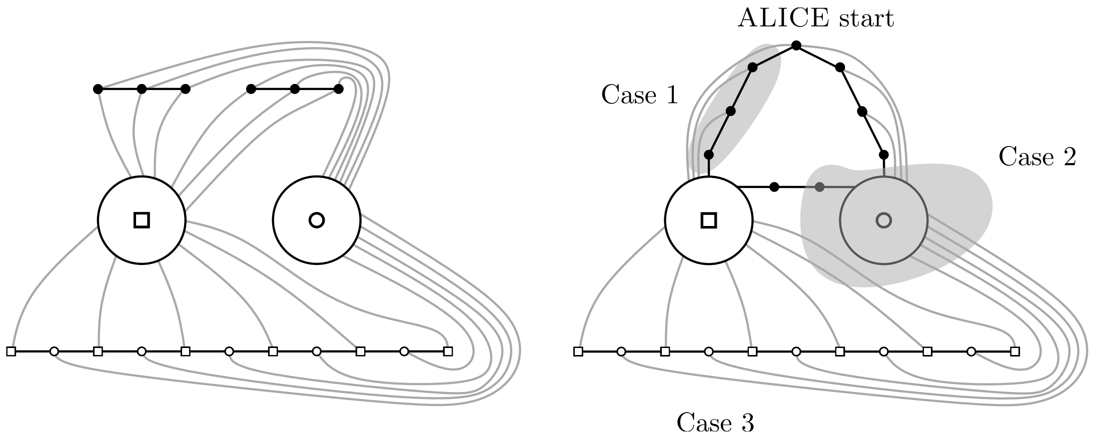


Figure 3.7: left: planar visage for BOB right: planar visage for ALICE

Theorem 3.11 (planar extremal graphs)**.** *For every natural number n there exists an undirected planar graph G on n vertices such that the outcome is $\Theta(\sqrt{n})$.*

Proof. The planar visage in Figure 3.7 differs from the visage on the left of Figure 3.6, only in one point: the big circle is replaced by a long path. We can see from the drawing on the left-hand side of Figure 3.7 that this graph is indeed planar.

BOB's strategy is the same here as in the ordinary visage. Except, BOB enters a long path, not a big circle, two turns after ALICE . □

Unfortunately, we cannot apply Lemma 3.2 to obtain a planar graph where ALICE wins. Even if we add a super vertex that is only connected to wisely chosen vertices, we could not make it work. Instead, we construct a new overhead graph and connect the bottleneck vertices.

Theorem 3.12. *For every natural number n exists an undirected planar graph G on n vertices such that the outcome is $\Theta\left(\frac{1}{\sqrt{n}}\right)$.*

Proof. This example is different from the previous one in two ways. Obviously, the overhead graph has changed, as you can see on the right hand side of Figure 3.7, but more subtly the distance between vertices adjacent to the bottleneck increased from 4 to 7. It is sufficient to show $\#A/\#B = \Theta(\sqrt{n})$, under optimal play.

We give an explicit strategy for ALICE. ALICE's start vertex is marked in Figure 3.7 on the top.

Case 1 BOB starts somewhere on a path from ALICE's start vertex to a bottleneck vertex. ALICE can just go directly to the corresponding bottleneck vertex and trap BOB this way.

Case 2 Bob starts w.l.o.g. on the ○-bottleneck vertex or on the adjacent vertex on the path between the two bottleneck vertices. Alice will slowly go to the □-vertex, i.e., not in one turn.

(a) Bob goes to the □-bottleneck vertex, via the short path connecting them. Alice reaches it earlier and Bob is trapped.

(b) Bob enters the long path. Alice will follow him two turns delayed. The outcome of this scenario is as in Example 3.10.

Case 3 In the only remaining case Bob starts on the long path. Alice will go to the bottleneck vertex which is closer to Bob's position and then to the other bottleneck vertex. (Because this takes longer than in the ordinary visage, the distance between the marked vertices must be increased to seven.) Thus Bob cannot leave the long path and Alice enters it second at least two turns delayed. □

Remark. In both planar visages we presented, it is possible for the winning player to cut off his or her opponent within at most 20 turns. But this might not be the optimal strategy, as shown in Example 3.10.

3.3.4 k-connected Visage

In this subsection, we construct a *k-connected visage*. The essence is twofold. First, we increase the number of bottleneck vertices, and secondly we replace paths of the big circle by double-trees, which will be introduced shortly. Double-trees have a high connectivity, a bottleneck and a Hamilton path.

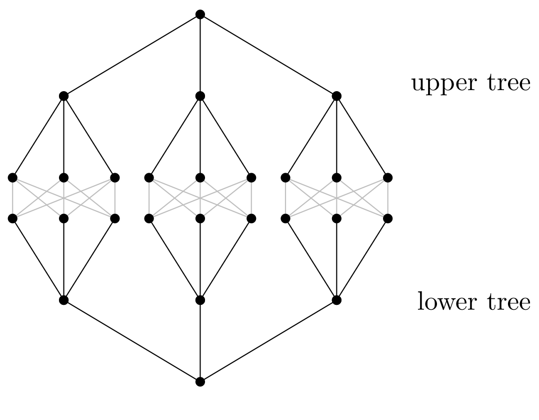


Figure 3.8: A double tree of height $h = 2$ and degree $d = 3$; the edge set E_3 connects these two trees. It is colored gray.

Definition 3.13 (Double-tree). *A* double-tree *of degree d and height h consists of two fully balanced rooted trees with d children for each inner vertex and height h. They are called upper and lower tree.*

The upper and lower trees are connected, via their leaves, by the edge sets E_1, E_2, and E_3, to be defined shortly. Note that we will call vertices, which were leaves in its respective original tree, still leaves, despite the fact that they do not have degree one.

We consider a plane drawing of the double-tree as in Figure 3.9. From left to right, we name the leaves of the upper half of the double-tree $u_1, u_2, \ldots, u_l$ and likewise we name *the leaves from the lower half of the double-tree* $w_1, w_2, \ldots, w_l$, *as depicted in Figure 3.9. We denote the number of the leaves with* l. *Regarding the leaves, indices are read to be modulo* l .

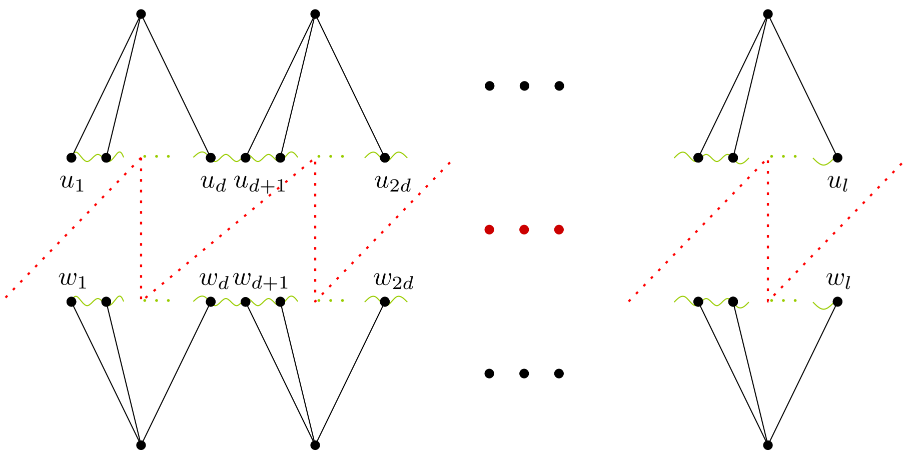


Figure 3.9: All the leaves with their names and parents. The edge set E_1 is drawn green and curvy. One of the cycles partitioning the set of leaves is drawn red and dotted.

We define

$$E_1 = \{(u_i, u_{i+1}) : i = 1, \ldots, l-1\} \cup \{(w_i, w_{i+1}) : i = 1, \ldots, l-1\} .$$

In other words when they come after each other in the order given in the definition, they are adjacent. The edges of E_2 *are indicated with red dotted lines in Figure 3.9 and it holds*

$$E_2 = \{(u_n, w_m) : n = m \textit{ or } n \equiv m + d \mod l\} .$$

At last E_3, *depicted in gray in Figure 3.8, connects leaves with the same corresponding parent of the other tree, more precisely*

$$E_3 = \bigcup_{i=0}^{l/d-1} \{(u_{id+j}, w_{id+k}) : 1 \le j, k \le d\} .$$

Note E_2 *and* E_3 *are not disjoint.* □

The next lemmas show easy properties of the double-tree. We define *concatenation trees* as follows: let $T_1, T_2, \ldots, T_n$ be rooted trees where each inner vertex has at least 2 children. Assume we have a drawing with all the trees crossing-free in a half plane such that all leaves are on the boundary of the half plane. We define leaves as adjacent if and only if the line segment connecting them does not contain any other vertex. Then this set of trees with the described additional adjacencies are called concatenation trees.

Lemma 3.14. *There exists a Hamiltonian path in the concatenation trees.*

Proof. Refer to Figure 3.10 for an illustration of the proof. We construct a special partition of the vertex set of the trees into paths starting and ending in adjacent leaves. In the second step we just connect these paths canonically.

We do the first step by induction on the maximum height of the trees under consideration. If every tree is just a single vertex, we define the partition to be the collection of all one element sets. Let $T_1, T_2, \ldots, T_n$ be some trees as described above and r the root of T_i. We assume that T_i is the highest tree and consists of more than one vertex. Let v_l be the rightmost leaf of the leftmost subtree of T_i and let v_r be the adjacent leaf on the boundary of the hyperplane, see in Figure 3.10. Exactly one path exists from v_l to v_r via the root r within T_i. We add this path to the partition set and delete it from the trees. Inner vertices with deleted parent become new roots. We do this for all trees of maximum height. Now all new trees have smaller height. We can partition the remaining trees by induction.

We connect the right end of each path to the left end of the next path to get a Hamilton path. □

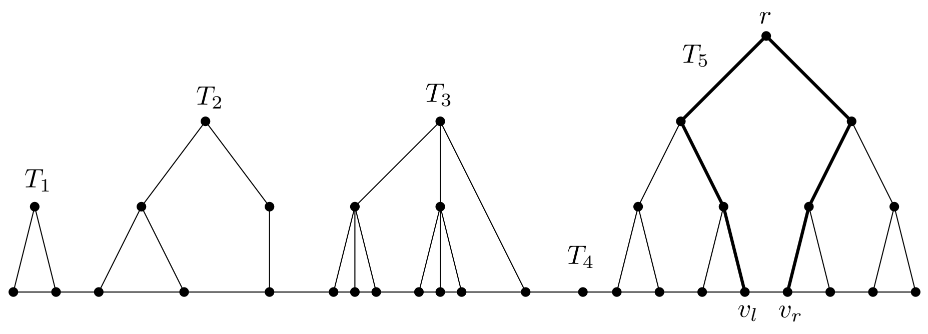


Figure 3.10: The induction step of Lemma 3.14

Lemma 3.15. *There is a Hamilton path from one root of the double-tree to the other root.*

Proof. We start our tour at the top root and go down to u_1 and from there to u_2. Next we use the path constructed in Lemma 3.14 to traverse the rest of the upper tree. After ferrying over from u_l to w_l we traverse the other tree in reverse order to reach the root of the other tree. In summary, we have traversed every vertex without reusing any vertex. □

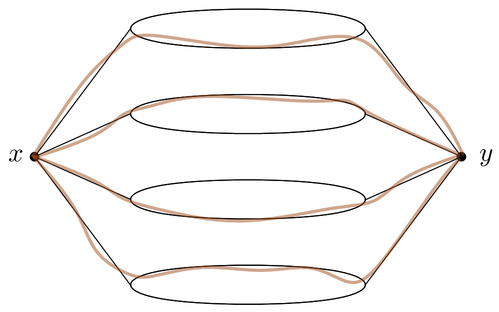


Figure 3.11: How to connect d paths from x to y via d disjoint cycles.

Lemma 3.16. *The double-tree is d-connected*

Proof. For any two nodes we will show that there are d vertex-disjoint paths connecting them. Thus by Menger's Theorem [28, Section 3.3], we know that the graph is d-connected.

The leaves can be partitioned into d cycles. For $i = 1, \ldots, d$, cycle i is described by

$$u_i, w_i, u_{i+d}, w_{i+d}, u_{i+2d}, w_{i+2d}, \ldots, u_{l-d+i}, w_{l-d+i},$$

see the red dotted path in Figure 3.9.

It is easy to see that for every vertex v there exist d vertex disjoint paths to the d respective cycles. For leaves these paths are very short and v lies on one of the cycles, but this is no problem.

To construct d disjoint paths from x to y, we use d vertex-disjoint direct paths from x to the cycles and from y to the cycles. These paths can be connected to each other, via the cycles. See Figure 3.11

The only issue would be that a path from x to a cycle uses y. But this can only happen once and would give us a path from x to y, which is disjoint from the other paths. □

Lemma 3.17 (Many afar leaves)**.** *For all natural numbers m, k and $d \geq 2$ there exists some natural number h_0 such that a double-tree with height $h \geq h_0$ and degree d has at least m leaves which have pairwise distance at least k from each other and from the root.*

Proof. Let $h \geq \lceil \log_d m \rceil + 2k$. Then there exists at least m disjoint height k subtrees, with another subtree in between. Take from each such tree any leave.

Because the height is larger than k it is clear that the distance to the root is at least k. Any two leaves can be either connected via a cycle as indicated in Figure 3.9 or via the tree. For the first case, note that there are at least 2^k leaves in between. For the second case note that any common ancestor has height at least k. □

Theorem 3.18 (k-connected extremal graphs)**.** *For every natural number n there exists an undirected k-*connected *graph G on at least n vertices such that* Bob *can traverse all but a constant number of vertices. (The constant depends on k, but not on n.)*

Proof. We construct a k-connected visage and prove later the required properties.

Refer to Figure 3.12 for the following construction. It consists of three parts, namely, the overhead graph, the bottleneck, and a modified big circle.

The bottleneck vertices are separated into two groups of k vertices each. The vertices of the first group are drawn as a $\square$ and the vertices of the second group are drawn as a $\circ$. Every vertex of the overhead graph is adjacent to every vertex of the bottleneck.

The big circle consists of degree k double trees that are strung together by their roots. The height h of the double trees is chosen so that each double tree has k leaves which all have pairwise distance at least $4k$, see Lemma 3.17. The number of the trees is even grows with n. For each double tree we mark the k afar leaves either all with a $\square$ or all with a $\circ$. We alternate between the two options. The vertices in the big circle are adjacent to the vertices in the overhead graph with the same label.

The overhead graph is composed out of $2k$ disjoint paths of a length longer than the number of vertices in a double tree, which does not depend on n.

We show that the construction is indeed k-connected and Bob has a strategy to traverse every vertex, except constantly many.

To see k-connectedness, we assume $(k-1)$ vertices have been deleted. We show that the graph is still connected. We show that every vertex is connected to one fixed

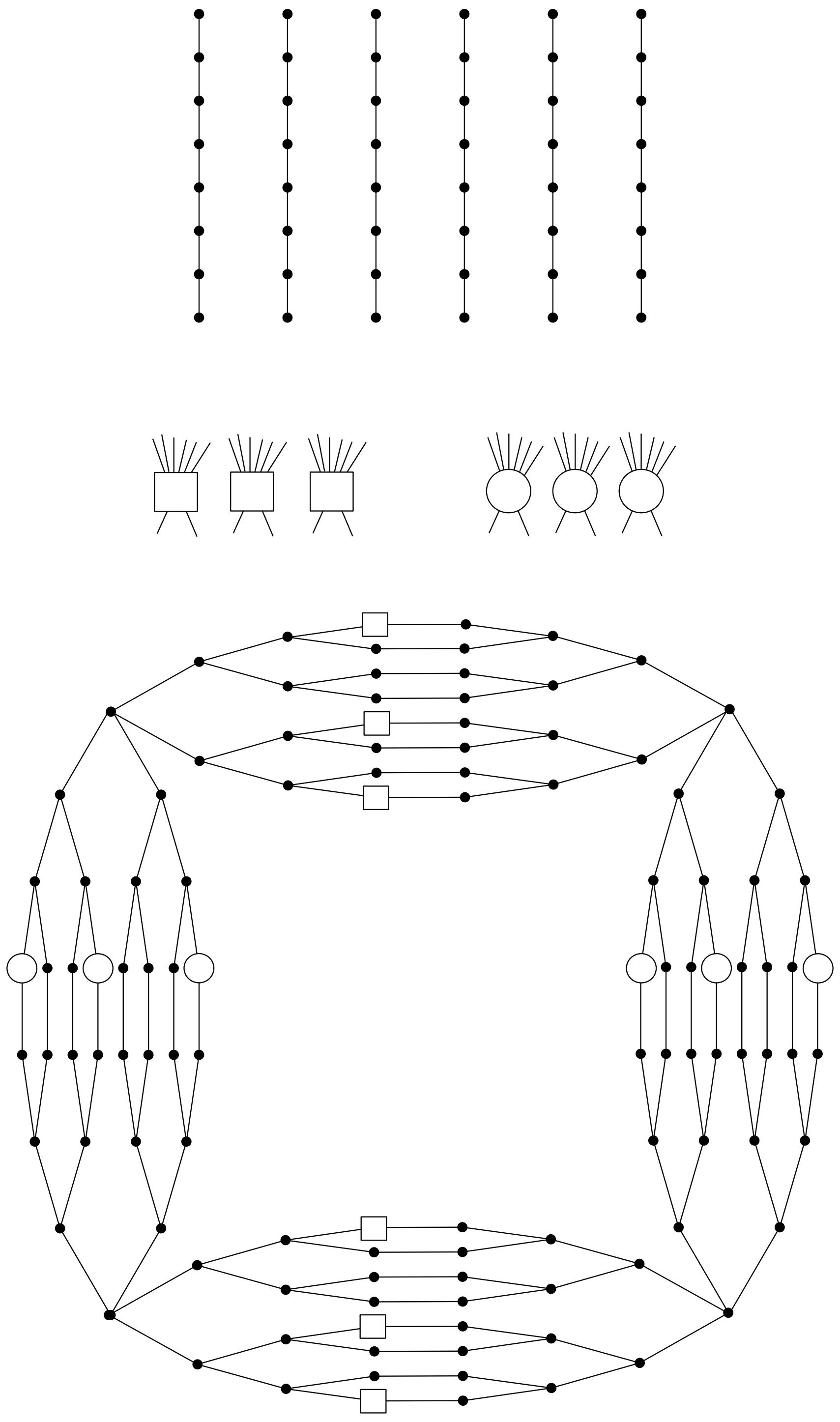


Figure 3.12: A 3-connected visage with 4 double-trees, the leaves depicted as a square are connected to the square bottleneck vertices and likewise leaves depicted as a circle are connected to the circle bottleneck vertices. For clarity of the drawing, we chose the height of the double-trees not large enough and omitted some edges of the double-trees.

bottleneck vertex, say v. Any vertex of the overhead graph is adjacent to v by the construction. Any other bottleneck vertex is connected to v via some vertex of the overhead graph. Lemma 3.16 implies that every double-tree is k-connected. Thus every vertex has still a path to a leaf that is connected to one of the bottleneck vertices, which is itself connected to v.

Bob still wins within the overhead graph. However, Alice could leave the overhead graph go to a bottleneck vertex, return to the overhead and continue to alternate between the two components. Bob can assure that the number of $\square$-bottleneck vertices remains the same as the number of $\circ$-bottleneck vertices and that he will always win within the overhead graph. For this purpose, he will always go to a bottleneck vertex immediately after Alice left one. And thus, return to to the overhead graph after Alice left the bottleneck. See Example 3.1 for a similar situation.

Eventually, Alice is the first to enter the big circle, say w.l.o.g. via a $\square$-bottleneck vertex. No matter how it came to that, at least one $\circ$-bottleneck vertex and one path in the overhead graph is unused, maybe more than one. Bob will first traverse all remaining $\square$-bottleneck vertices and then all remaining $\circ$-bottleneck vertices up to one. Thereafter, Bob waits for Alice to traverse a root of a double-tree. Once Alice did this, it is clear whether she traverses the big circle in clockwise or counterclockwise direction.

At last, Bob goes to the next double-tree via the last remaining bottleneck vertex and then go to the next root, which Alice would reach. He can reach the root faster than her because he starts in the middle and Alice starts on the opposite site of the double-tree. Alice cannot leave this double-tree because all entrances, bottleneck vertices and roots, are taken.

Summing up, Bob can traverse all the vertices of the big circle, except maybe two double trees. The rest of the graph has constant size. This finishes the proof. $\square$

Adding a super-vertex as in Lemma 3.2 gives us instantly a k-connected visage that is good for Alice.

3.4 Complexity Question

In this section we show that Tron is PSPACE-complete, for various game modes. To do this, it turned out to be convenient to consider variations where the graph is directed and/or start positions for Alice and Bob are given. We reduce quantified boolean formula (QBF) to Tron. It is well known that it is PSPACE-complete to decide if a QBF φ is true. A quantified boolean formula has the form $\varphi \equiv \exists x_1 \forall x_2 \exists x_3 \forall x_4 \ldots : C_1 \wedge \ldots \wedge C_k$ with each $C_i \equiv L_{i_1} \vee L_{i_2} \vee L_{i_3}$ and L_{i_j} some Literals [66, Section 8.3]. In Theorem 3.19, we will construct for each φ a directed graph G_φ with given start positions v_1 and v_2 such that Alice has a winning strategy if and only if φ is true. In Theorem 3.20, we will modify this graph, such that it becomes undirected. In Theorem 3.21 we will construct a directed overhead graph to G_φ, which will force Alice and Bob to choose certain starting positions. At last in Theorem 3.22 we will construct an undirected overhead graph. Here we will make use of the constructions of the preceding theorems.

Theorem 3.19 and 3.21 has already been proven by Bodlaender [19]. The proof of the first theorem is similar to the proof that the game Geography is PSPACE-complete [66]. We repeat his proofs, with subtle changes. These differences are necessary for Theorems 3.20, 3.21 and 3.22 to work. It also serves the self-containment of this work.

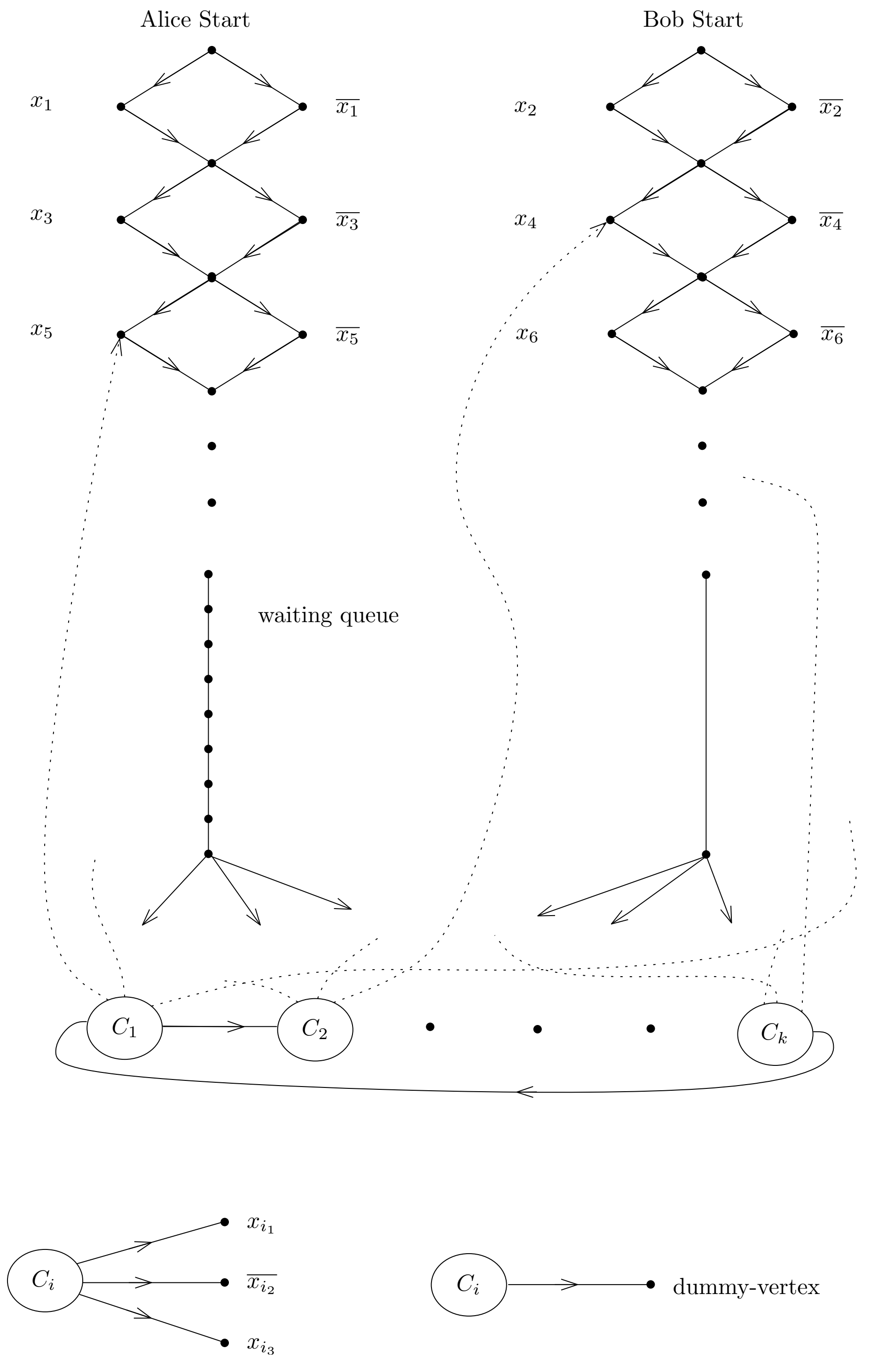


Figure 3.13: The graph G_φ; the edges from the clause vertices to the variable vertices are indicated with dotted lines.

3.4.1 Normal Play

Theorem 3.19 (directed, given start [19])**.** *The problem to decide if* Alice *has a winning strategy in a* directed *graph* with *given start positions is PSPACE-complete.*

Proof. Given a QBF φ with n variables and k clauses, we construct a directed graph G_φ as depicted in Figure 3.13. See also Figure 3.14 for a concrete example. It consists of starting positions for Alice and for Bob from where variable gadgets begin such that Alice and Bob have to decide whether they move left or right; this represents an assignment of the corresponding variable. Thereafter, Alice has to enter a path of length $k - 1$, which we call the waiting queue. Meanwhile Bob enters the clause gadget, which consists of k vertices arranged in a directed cycle each representing exactly one clause. Thus Bob can traverse all but one clause-vertex before Alice enters the clause gadget. When she enters, she has only one clause-vertex to go to, which was chosen by Bob. Now, from each clause vertex we have edges to the corresponding variables and one edge to a dummy-vertex. Thus each player can make at most one more turn. Note if Bob and Alice make a turn, Alice wins; otherwise the game ends in a tie. Thus Bob takes the dummy-vertex, otherwise Alice takes it. Consequently, if φ is true Alice has a strategy to assign the variables in a way that every clause becomes true and she is still able to make one more turn and wins. Otherwise, Bob has a strategy to assign the variables such that at least one clause is false. Thus Alice cannot move anymore from the clause vertex and the game ends in a tie. This shows PSPACE-hardness. As the game ends after a linear number of turns, it is possible to traverse the game tree using linear space. See [66] for a similar argument. $\square$

Our approach is to take the graph G_φ from Theorem 3.19 and convert it to a working construction for Theorem 3.20.

Theorem 3.20 (undirected, given start)**.** *The problem to decide whether* Alice *has a winning strategy in an* undirected *graph* with *given start positions is PSPACE-complete.*

Proof. We replace every directed edge of G_φ by an undirected one. Further, we will carry out 4 modifications and later prove, that the resulting graph G'_φ has the desired properties.

In the following, the exact lengths of the described paths are not very important. But it is important, that certain paths have the exact same length. This makes certain options equally good, or let them differ by exactly one.

Modification 1 (slow-path)**.** *As we want that* Alice *and* Bob *assign each variable in order, we must prevent them from using the edge from a variable-vertex to a clause-vertex. We achieve this via subdividing every such edge to a path of length $2k+n$. See the bottom of Figure 3.15.*

Modification 2 (waiting queue)**.** *It might happen that* Bob *cuts off the waiting queue. We prevent this by replacing the waiting queue by the graph depicted on the top of Figure 3.15. We again refer to this construction as a* waiting queue.

Modification 3 (dummy-vertex)**.** *Another concern is that* Bob *might go towards the dummy-vertex and return. To prevent this, we replace all the edges to the dummy-vertex by the construction in Figure 3.16.*

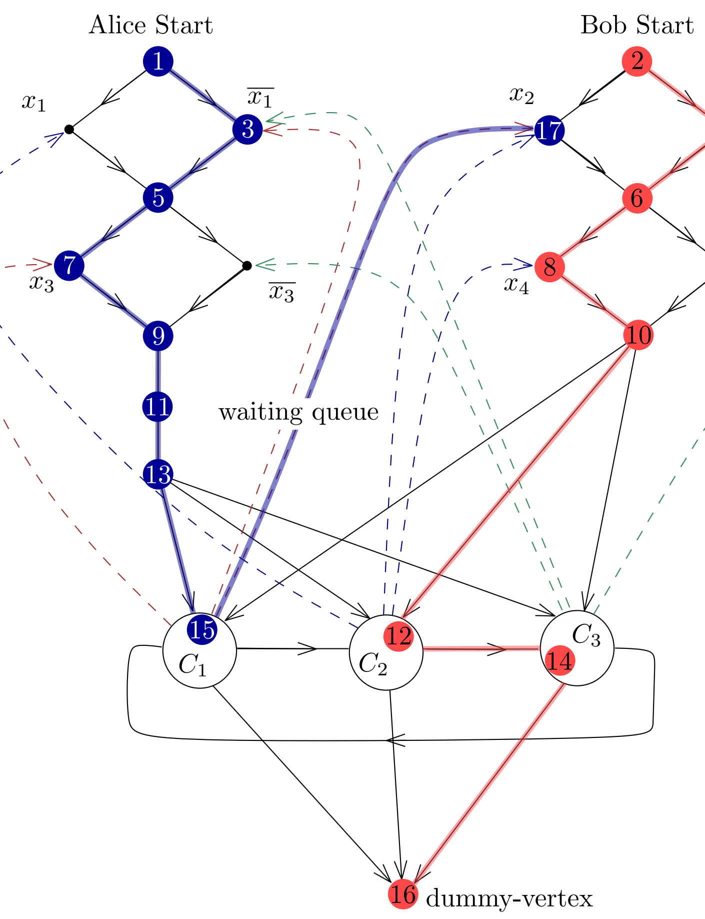


Figure 3.14: G_φ with QBF $\varphi = \exists x_1 \forall x_2 \exists x_3 \forall x_4 : (\overline{x_1} \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$. The indicated paths of ALICE and BOB correspond to an assignment of the variables $x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 0$.

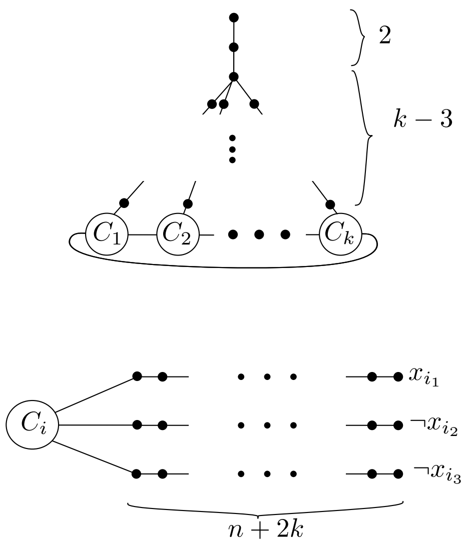


Figure 3.15: the waiting queue on the top and the slow path on the bottom.

Modification 4 (spare path). *It might be an advantage for* Bob *to go to a literal which is contained in two clauses, instead of going to the dummy-vertex because he then might use the other slow path to return to a clause gadget and receive in total* $4k+2n+1$ *vertices after leaving the clause gadget. We attach a path of length* $2n+k$ *to each variable-vertex and the dummy-vertex. This path is denoted in the following as* spare path.

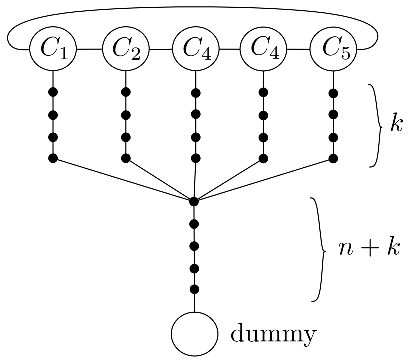


Figure 3.16: Modification of the paths to the dummy-vertex

Let us start to describe the desired scenario as depicted in Figure 3.17 on the top. In this scenario Alice and Bob traverse the variable vertices alternatingly. Then Bob traverses all but one clause vertex, which Alice occupies. At last Bob goes to the dummy vertex and Alice to the a free variable vertex. If such a vertex exists, Alice will win by 1, otherwise Bob will win by $2n+k$, because he can use a spare path (to avoid clutter the spare path is not drawn). We have to show that it is not useful for either participant to divert from this scenario.

First, we show that after Alice and Bob leave their respective start positions they have to assign the variables. There are only two strategies they possibly could follow instead. The first is to use a spare path from Modification 4. This gives at most $2n+k$ vertices. The other player would just go down to the clause vertices and then to the dummy-vertex and proceed to the spare path from the dummy-vertex. Thus using the spare path at this stage leads to a loss.

The other option is to use a slow-path from a variable to the clause-gadget as introduced in Modification 1. It takes quite a while to traverse this path and meanwhile, the other player can just go down to the clause-gadget, traverse all the clause-vertices and then go to the dummy-vertex. Again, it turns out that this strategy is a certain way of losing (assuming the opponent plays optimally). See the first irrational scenario in Figure 3.17.

So far, we have established that Bob reaches the clause gadget, Alice reaches the waiting queue, and they have assigned all the variables alternatingly on their way. We would like if Bob traverses $k-1$ clause vertices and Alice goes to the clause vertex, that Bob left her. See the desired scenario in Figure 3.17. Bob could make one of two plans to prevent this scenario. The first plan is that he might try to go to the dummy vertex and return before Alice has reached a clause-vertex. (It is not necessarily useful, but we want to exclude it anyway.) But the time to return is long enough that Alice will have taken all the clause vertices meanwhile and Bob would receive more vertices if he were to proceed all the way to the dummy vertex and take the spare path. The second plan he might pursue, is to shortcut the waiting queue. Luckily, the queue splits after

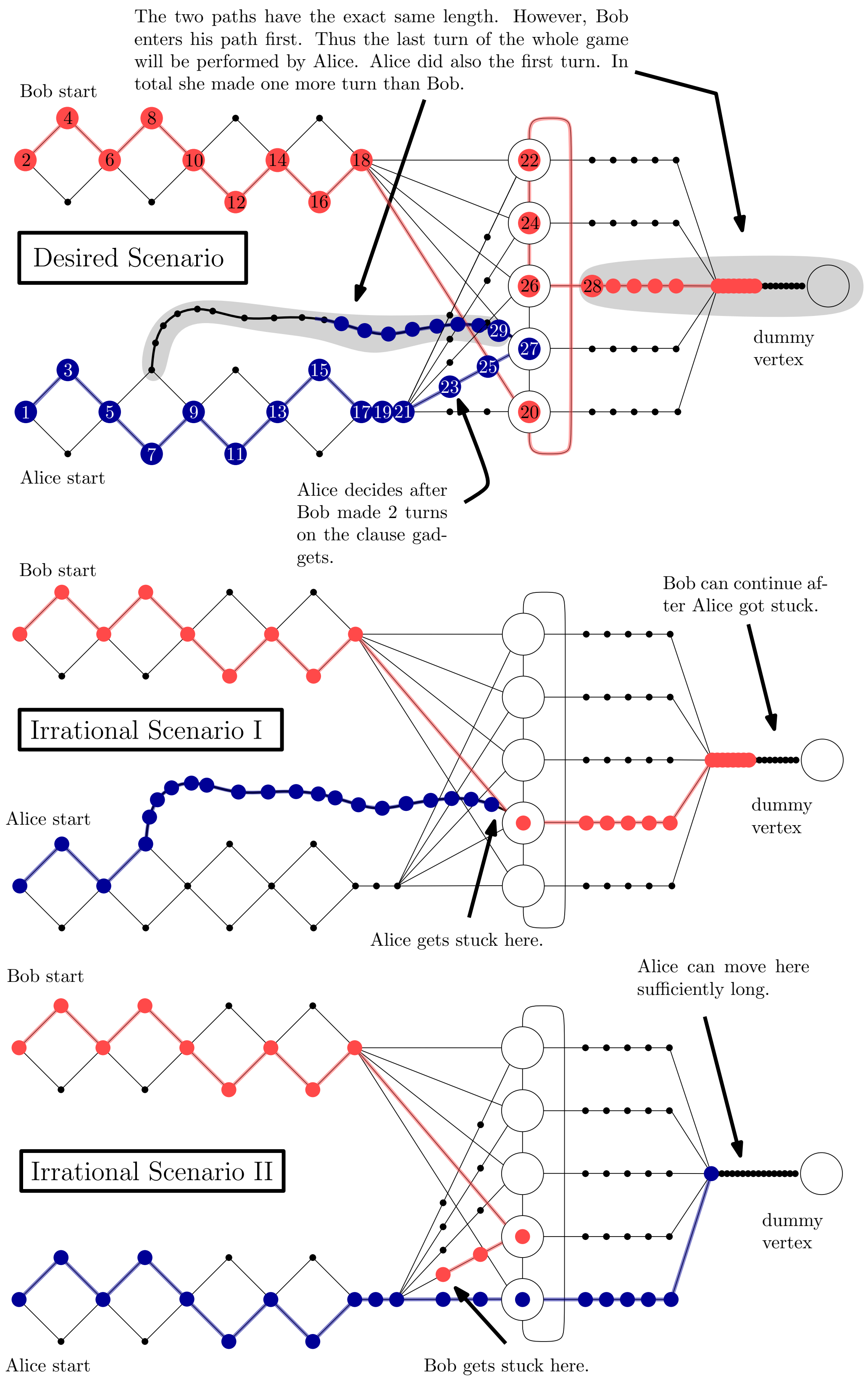

The two paths have the exact same length. However, Bob enters his path first. Thus the last turn of the whole game will be performed by Alice. Alice did also the first turn. In total she made one more turn than Bob.
Bob start
Desired Scenario
dummy vertex
Alice start
Alice decides after Bob made 2 turns on the clause gadgets.
Bob start
Bob can continue after Alice got stuck.
Irrational Scenario I
dummy vertex
Alice start
Alice gets stuck here.
Bob start
Alice can move here sufficiently long.
Irrational Scenario II
dummy vertex
Alice start
Bob gets stuck here.

Figure 3.17: The desired scenario and two undesired ones mentioned in the proof of Theorem 3.20.

two vertices. Thus, when Bob enters the queue before two turns, Alice can avoid him by taking a different branch and the Bob himself gets trapped. If he waits two turns, he must have determined a clause vertex for Alice already. Hence Alice knows which branch to use. This particular branch cannot be reached by Bob by then. See the second 'irrational' scenario in Figure 3.17.

In summary we have established that Bob indeed has to traverse $k-1$ clause-vertices and Alice obviously has to go to the clause-vertex Bob left for her.

At this stage, it is Bob's turn. One of the longest paths that remains goes to the dummy vertex and proceeds via a spare path. So he had better take it, because otherwise Alice will take it and he loses.

Consider Alice's position. If there is a variable vertex she can reach, she also has a path of the same length as Bob does and this would imply that she will win. If not, then she could only go towards a variable-vertex and Bob will win.

Note that the path to the dummy vertex and to a variable vertex has the same length. The reason Alice has a higher score than Bob is that she will make the last turn. Recall that Alice starts. As the two players move alternatingly, Alice occupied one more vertex than Bob.

And again as in Theorem 3.19, Alice has a winning strategy in G'_φ if and only if φ is true. □

If Alice has a winning strategy, she will win not by more than one vertex. Conversely, if Alice does not have a winning strategy, Bob can win, via the dummy vertex and a spare path, by $2n+k+1$ vertices.

We proceed to show how to force Alice and Bob to choose certain start positions in a *directed* graph. We construct a graph $H(G)$ such that Alice wins in $H(G)$ if and only if Alice wins in G when both players start at certain positions v_1 and v_2 in G. It follows that Alice wins in $H(G_\varphi)$ if and only if φ is true. With a similar but different construction, Theorem 3.21 was shown in [19]. In this chapter, we again give a slightly different proof because it is an essential step for our proof of Theorem 3.22.

Theorem 3.21 (directed, without given start [19])**.** *The problem to decide whether* Alice *has a winning strategy in a* directed *graph* without *given start positions is PSPACE-complete.*

Proof. Assume we are given some directed graph G on n vertices, among them $v_1, v_2 \in G$. We construct some directed graph $H(G)$ in polynomial time such that Alice wins in $H(G)$ if and only if Alice wins in G with the predefined start positions v_1, v_2. We have reduced the problem of Tron on a directed graph without start positions to the case of Tron with given start positions.

The general idea of such an overhead graph is simple. We construct two vertices, which are very *powerful*, such that Alice and Bob want to start there, but once there, they are forced to go to the start vertices of the original graph. The idea is also used in Theorem 3.22.

We describe the construction of $H(G)$ as depicted in Figure 3.18 in detail.

We add two vertices u_1 and u_2 with attached directed paths of length $2n = 2\,|V(G)|$ to the start vertices v_1 and v_2 respectively. So far, the longest path starts at u_1 or u_2 and has length l between $l_{low} = 2n$ and $l_{up} = 3n$.

Then we add two directed *auxiliary paths* of length $l_{up}+1$ and vertices s_1 and s_2. We denote the vertices of the auxiliary paths by $x_1, \ldots, x_{(l_{up}+1)}$ and $y_1, \ldots, y_{(l_{up}+1)}$. The

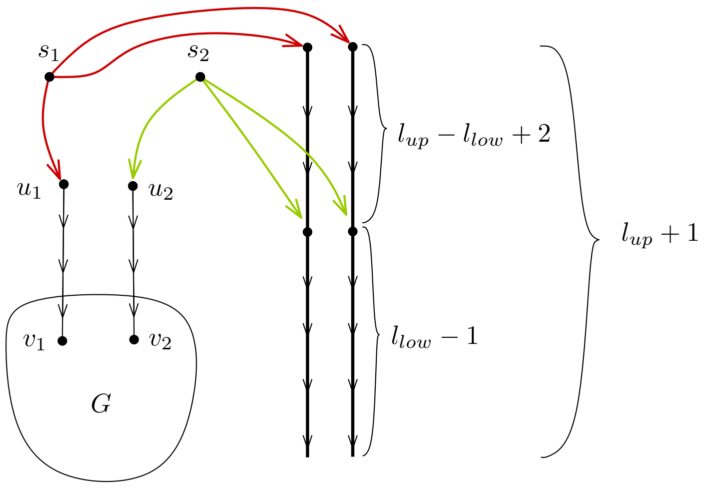


Figure 3.18: H(G)

vertex s_1 is adjacent to u_1, x_1 and y_1. The vertex s_2 is adjacent to $x_{(l_{up}-l_{low}+3)}, y_{(l_{up}-l_{low}+3)}$ and u_2.

We require that the first part of the auxiliary path is shorter than the second part, which is the case as long as $l_{up} - l_{low} + 2 < l_{low} - 1$. Once one of the players is in an auxiliary path or G, there is no way out of the respective component simply because there is no outgoing edge.

Claim 1: If Alice starts at s_1 and Bob at s_2 then Alice should go to u_1 and Bob should go to u_2.

If Alice does not go to u_1, Bob goes into the same auxiliary path as her and receive more vertices than Alice and thus she loses. If Bob does not go to u_2, he receives fewer vertices in an auxiliary path than Alice in G.

Claim 2: It is best for Alice to start at s_1.

Case 1 If Alice starts in G then Bob starts at the top of an auxiliary path.

Case 2 Alice starts in an auxiliary path. As the path is directed, Bob starts in front of her.

Case 3 If Alice starts in s_2 then Bob starts in s_1. In this situation Bob can get $l_{up} + 2$ vertices, via an auxiliary path, in total and Alice at most $l_{up} + 1$ vertices, by going to u_2.

These first 3 cases show Claim 2.

Claim 3: If Alice starts in s_1, Bob is always better off starting at s_2.

Case 4 Bob starts in an auxiliary path. Alice will go to the other auxiliary path and win by at least one.

Case 5 Bob starts in G or a path from u_i to G. Alice will then just go to an auxiliary path.

Claim 1, 2 and 3 finish the proof. □

We show hardness if the graph is *undirected* and the starting positions are *not* given. We will do that by using the graph G'_φ, constructed in Theorem 3.20, and an undirected version of $H(G)$, which we will denote by $H'(G)$. Unfortunately, this will not work immediately. We will, therefore, construct an overhead graph $F(H'(G))$ on top of $H'(G)$.

Theorem 3.22 (undirected, without given start)**.** *The problem to decide whether* ALICE *has a winning strategy in an* undirected *graph* without *given start positions is PSPACE-complete.*

Proof. The general idea of this construction is the same as in Theorem 3.21, but because we build up from the construction from Theorem 3.21, everything gets more involved. However, each argument is elementary.

Let $H'(G)$ be the graph $H(G)$ with all directed edges replaced by undirected ones. Further, the auxiliary paths have to be changed slightly because $l_{low} = 4n$ and $l_{up} = 5n$. In other words, the length l of the longest path in the graph G together with the two paths from u_1 and u_2 has between $l_{low} = 4n$ and $l_{up} = 5n$ vertices.

Claim *The following properties hold for $H'(G)$:*

p1 If ALICE *starts at s_1 and* BOB *starts at s_2, then* ALICE *has to go to u_1 and* BOB *to u_2.*

p2 If ALICE *starts at s_2 and* BOB *at s_1,* BOB *will win.*

p3 If we assume s_1 and s_2 are forbidden to use, except when started at then the longest path starts at s_1. (longest path in the sense as if only one player would exist.)

p4 Any path from s_1 to s_2 can be extended using an auxiliary path.

p5 The shortest path from s_1 to s_2 has length at least 5.

Proof claim. Properties p1 and p2 hold for directed graphs according to the proof of Theorem 3.21 and hold by the same arguments for the undirected case.

Property p3 holds by the definition of the auxiliary paths. In particular, the longest path consist of s_1 and an auxiliary path. The reader can check easily that all other paths are shorter.

Property p4 holds because any path from s_1 to s_2 uses at most one auxiliary path. Thus the path can be extended to an auxiliary path that has not been used yet.

Because s_1 and s_2 have no common neighbors nor are adjacent, Property p5 follows. This proves the claim. □

We construct an overhead graph of $H'(G)$, namely $F(H'(G))$, as depicted in Figure 3.19. It consists of two copies of $H'(G)$, which we denote by H^a and H^b. In addition, two vertices t_1 and t_2 are part of the construction. We indicate with an upper index ^{a} or ^{b} whether a vertex belongs to H^a or H^b, for example s_1^a is the s_1 vertex in H^a.

The edge set consists of all the edges in H^a, H^b, and the edges

$$(t_1, s_1^a), (t_1, s_1^b), (t_1, s_2^a), (t_1, s_2^b), (t_2, s_2^a),$$

$$(t_2, s_2^b), (s_1^a, s_2^a), (s_1^b, s_2^b), (s_1^a, s_1^b), (s_2^a, s_2^b).$$

We call t_2, s_2^a and s_2^b *dot-vertices* and t_1, s_1^a and s_1^b *box-vertices.*

The proof splits into two situations. At first, we will assume that Alice has a winning strategy in G, if both players start at v_1 and v_2. Case 1 to Case 4 are devoted to this situation and cover all possible moves of Bob. We will show an explicit winning strategy for Alice.

In the second situation we assume Alicedoes not win in G and will show that Bob can achieve at least a tie in $F(H'(G))$. Case 5 to Case 9 consider all possible moves of Alice and an appropriate respond of Bob.

Now, we start with the first situation. Alice starts at t_1. W.l.o.g., we will only consider movement to H^a instead of H^b when the situation is symmetric.

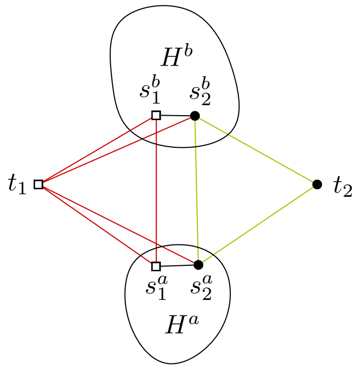


Figure 3.19: The construction of $F(H'(G))$ with the underlying graph H^a and H^b.

Case 1 Bob starts in H^a but not in s_1^a or s_2^a. Bob has distance at least 2 to either s_1^a or s_2^a, by p5. Both can be reached by t_1. So Alice can imprison Bob by going to the closer vertex and then to the other vertex. Bob cannot escape, because of p5. (If both vertices have the same distance Alice can choose.) After that Alice can go to H^b and wins there by p3.

Case 2 Bob starts at s_1^a. Alice takes s_1^b. Then Alice copies every move of Bob and thus wins, since the only move she cannot copy is to t_2. But p4 shows us that this is not a wise move for Bob.

Case 3 Bob starts at s_2^a. Alice goes to s_1^a and s_1^b in this order. We consider all possible positions of Bob.

(a) If Bob is in H^a, he will lose by p3.

(b) If he goes to s_2^b in his next turn, and then goes to t_2 he cannot move anymore and will lose.

(c) If he goes to s_2^b in his next turn, and then makes some other turn in H^b, he is the first to make a turn in H^b and Alice is the second to make a turn in H^b. Thus he loses by p2.

(d) He moved to s_2^b via t_2 and will lose by the assumption.

Case 4 Bob starts at t_2. Alice will go to s_1^a, s_1^b and then enter H^b.

(a) Bob enters H^b one turn before Alice from s_2^b. As in the Case 3 (c) Bob makes the first turn in H^b and the roles are exchanged and we apply p2.

(b) He enters H^b one turn after ALICE and lose by p1 and the assumption.

(c) He enters H^a and loses by p3.

Now, we investigate the second situation and show that BOB can achieve at least a tie. The proof goes by exhaustive case distinction.

Case 5 ALICE starts at t_1. BOB goes to t_2.

(a) If ALICE goes to s_1^a, then BOB will go to s_2^a. If, thereafter, ALICE enters H^a, he will as well. The only other possible turn,for ALICE, is to s_1^b. Then he goes to s_2^b. In any case BOB and ALICE enter the same copy of $H'(G)$ and BOB gets at least a tie, by assumption together with p1.

(b) If ALICE chooses s_2^a as her second move, BOB goes to s_2^b and imitate all of ALICE's moves and thus gets a tie.

Case 6 ALICE starts in H^a but not s_1^a or s_2^a. BOB cuts her off and enters the other copy of $H'(G)$ via s_1^b, see p3, p5 and Case 1.

Case 7 ALICE starts at s_1^a. Then BOB will start at t_1.

(a) ALICE goes to s_1^b. BOB will go to s_2^b. Now ALICE has to enter H^b and BOB acquires at least a tie by assumption and p1.

(b) ALICE goes this time to s_2^a. BOB will then go to s_2^b. Thus ALICE has to enter H^a and BOB can enter H^b via s_1^b and thus wins by p3. (Unless ALICE goes to t_2 and is immediately stuck.)

Case 8 ALICE starts on s_2^a. Then BOB will start at t_1.

(a) ALICE goes to s_1^a, BOB can go to s_1^b and imitate her moves as in Case 6.

(b) ALICE goes to s_2^b, BOB takes s_1^b then ALICE can make a last move to t_2 or enter H^b. In the second case, BOB goes to H^a via s_1^a and wins by p4.

(c) ALICE goes to t_2. Then BOB goes to s_2^b and wins.

Case 9 ALICE starts at t_2. BOB goes to t_1 and follows her in the sense that if she goes to s_2^a, he will go to s_1^a. Thus either ALICE enters H^a via s_2^a and BOB will enter H^a via s_1^a and thus wins by p2, or the same happens with H^b one turn later.

□

3.4.2 Misère Game

In this section the misère game is considered. As stated in the introduction in the misère game the players want to have a lower score than their opponent. Again four variants are considered, namely, when start positions are given or not and whether the graph is directed or undirected.

We use the expressions that a player *dies* or *kills* himself or herself in some vertex and so on. This is a short-hand phrase to say that a player has no possible move from this vertex or in this situation.

Note players are not allowed to kill themselves by going to an already occupied vertex. Both players must go to an free vertex if available in the misère game.

Theorem 3.23 (directed, without given start, misère)**.** *The problem to decide whether* Alice *has a winning strategy in a* directed *graph* with or without *given start positions is PSPACE-complete.*

Proof. We reduce from the decision problem of Tron on a directed graph under *normal play* as in Theorem 3.21.

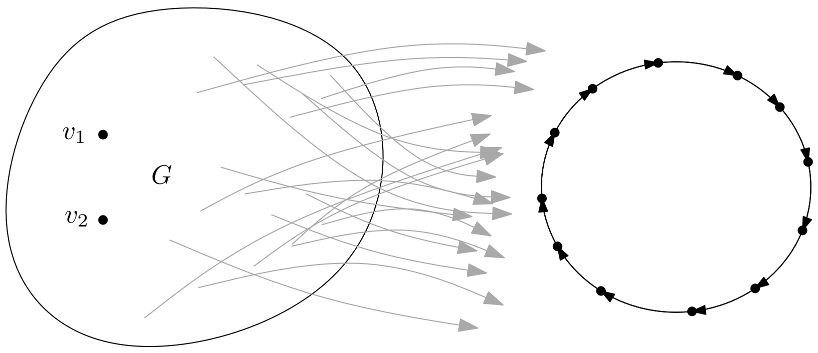


Figure 3.20: G denotes the graph for which we want to decide if Alice or Bob has a winning strategy under normal play.

For the following refer to Figure 3.20. Let G be a directed graph for which we want to decide if Alice has a winning strategy under normal play. We construct a new graph $H(G)$ with the following property:

$$\begin{aligned}&\text{Alice has a winning strategy in } G \text{ under normal play}\\ \Leftrightarrow\ &\text{Alice has a winning strategy in } H(G) \text{ under misère play.}\end{aligned}$$

We describe the construction of $H(G)$ and the desired property will become apparent. The graph $H(G)$ consists of G itself, a directed cycle of length k and all edges (v, w), where v is a vertex of G and w a vertex of the cycle. The construction can be clearly made in polynomial time.

What ever player p visits a vertex of the cycle first will lose because the other player q will go to the vertex before p on the cycle. Therefore, q will finish and p still has to traverse $k - 2$ vertices.

This implies both players want to start in G and whoever loses the normal play in G will lose the misère game in $H(G)$.

The proof works when start positions are given and when they are not.

□

Theorem 3.24 (undirected, with given start, misère)**.** *The problem to decide whether* ALICE *has a winning strategy in an* undirected *graph G* with *given start positions is PSPACE-complete.*

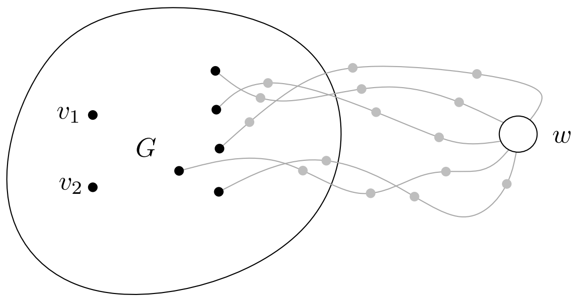


Figure 3.21: G denotes the graph for which we want to decide if ALICE or BOB has winning strategy under normal play.

Proof. We reduce from the decision problem of TRON on an undirected graph under *normal play* as in Theorem 3.20.

For the following refer to Figure 3.21. Let G be an undirected graph for which we want to decide if ALICE has a winning strategy under normal play. We construct a new graph $H(G)$ with the following property:

ALICE has a winning strategy in G under normal play
$\Leftrightarrow$ ALICE has a winning strategy in $H(G)$ under misère play.

First, we describe the construction of $H(G)$, and then the desired property will be explained.

The graph $H(G)$ consists of G itself, a special vertex w, and a path from every vertex of G to w of length 10.

At some point one of the players, say ALICE will enter a path to w. Thereafter, BOB has a winning strategy by also going to w via some other path immediately one turn after ALICE. Clearly, ALICE will arrive at w one turn before BOB. This means BOB dies and ALICE has to choose one of the remaining vertices to return to G and thus move at least another 3 vertices.

If ALICE has a winning strategy in G with the start vertices v_1 and v_2 in G under normal play, BOB has to enter a path first. Otherwise ALICE has to enter a path first. This finishes the proof. □

Theorem 3.25 (undirected, without given start, misère)**.** *The problem to decide whether* ALICE *has a winning strategy in an* undirected *graph* without *given start positions is PSPACE-complete.*

Proof. We reduce from the decision problem of TRON on an undirected graph with given start positions under *misère play* as in Theorem 3.24. We constrain the input graphs to those graphs where no player can finish the game within 10 turns. This is a property of the graphs of Theorem 3.24 that no player can kill herself/himself within 10 turns. This is because the length of the paths to w is 10, see the proof of Theorem 3.24. On the other

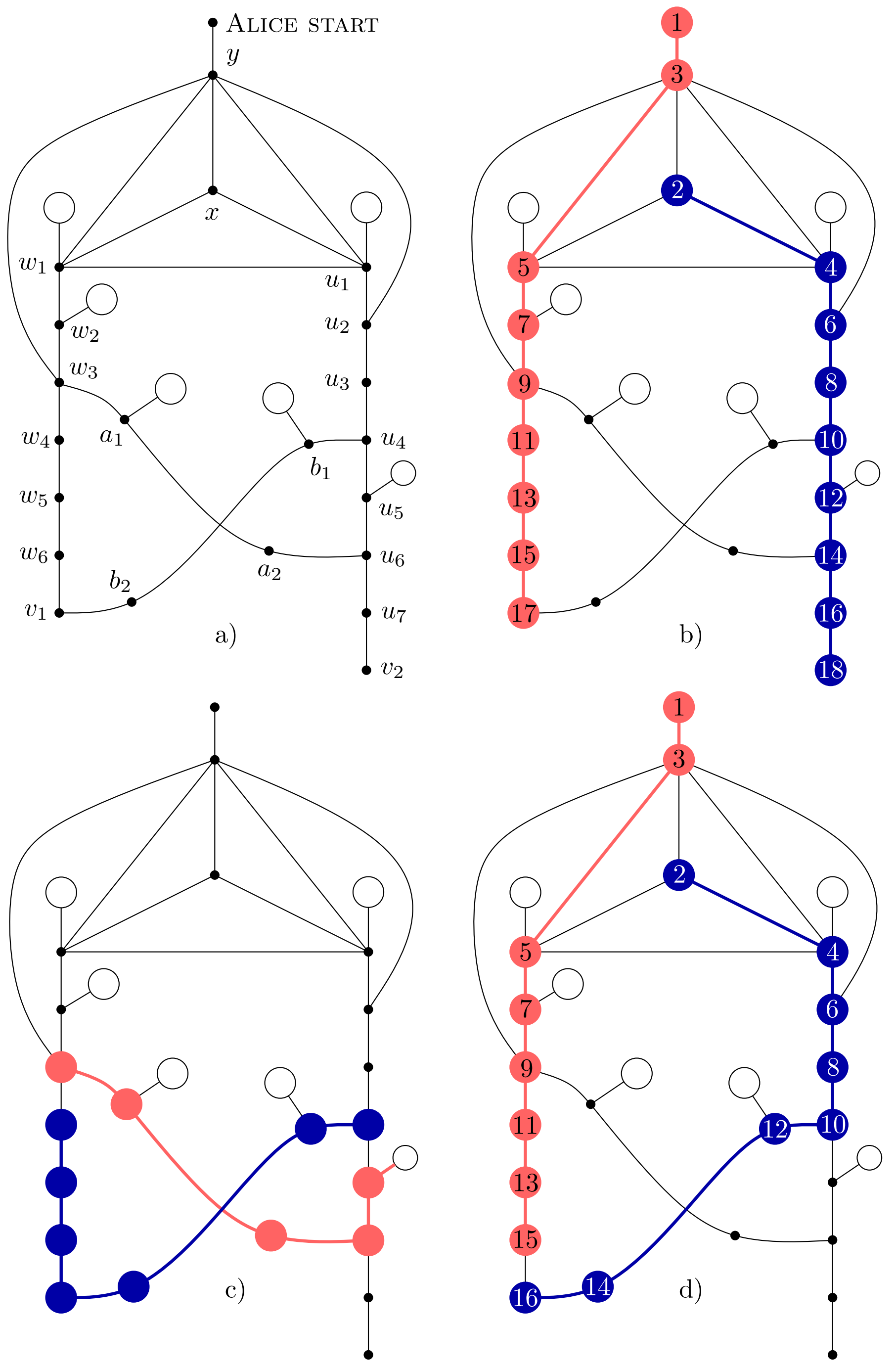


Figure 3.22: a) The graph $H(G)$ and its labels. b) We will show that BOB goes to v_2 and ALICE goes to v_1 these are the start vertices of G. We will exclude all other possible moves of ALICE and BOB . c) Consider the two *bridge paths* a_1a_2 and b_1b_2 from the left to the right and vice versa. It is clear that those paths cannot be traversed in the wrong direction because of the the cycle of length $2n$ attached to a_1 and b_1. Also note that if both players use the bridge paths the player coming from the right will be able to kill itself in w_4. The other player cannot kill itself in u_5. Thus if the player on the left is at w_3 before the player on the right is at u_4 then the player on the left will not use the bridge path. d) Consider the situation that ALICE and BOB will go towards v_1 and v_2 respectively. Then BOB loses if he tries to use the bridge path.

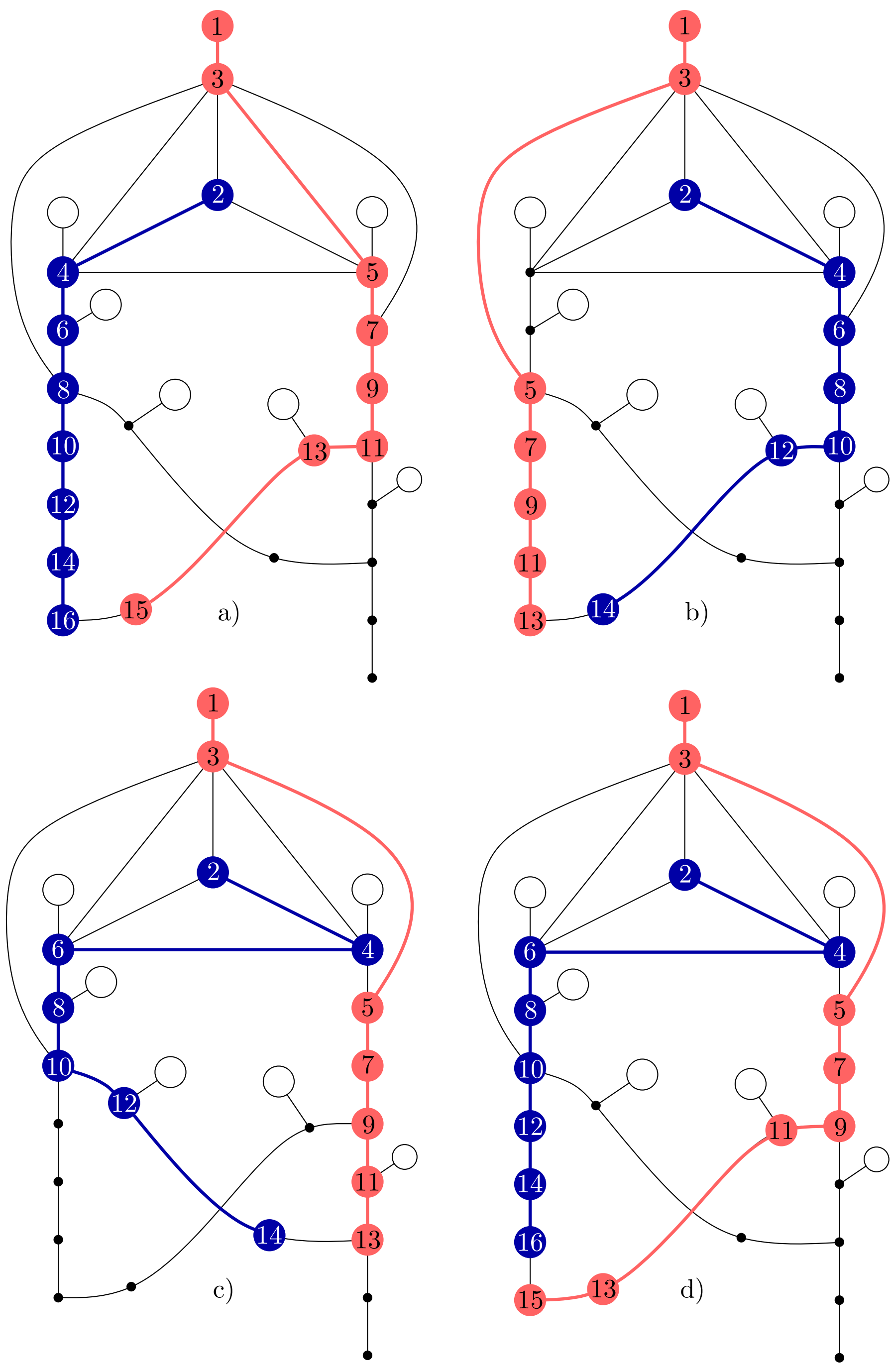


Figure 3.23: a)If Bob goes from x to w_1 Alice wins. b) **Claim 4:** Alice loses if she uses the edge (y, w_3). c) and d) **Claim 5:** Alice loses if she uses the edge (y, w_2).

hand, it is obvious that one can kill oneself within n turns in G, if n specifies the number of vertices of G.

For the following, refer to Figure 3.22 and Figure 3.23. Let G be an undirected graph with start vertices v_1 and v_2 for which we want to decide if ALICE has a winning strategy in the misère game. We construct a new graph $H(G)$ with the property that under optimal play ALICE and BOB will go to v_1 and v_2 respectively.

For the construction of $H(G)$, see Figure 3.22 a). The vertices w_1, w_2, u_1, a_1, b_1 and u_5 are connected via an edge to a cycle of length $2n$. This makes it difficult to die in these vertices, because it is always possible to enter the cycle, where one has to traverse $2n$ more vertices. As a matter of fact, if a player enters one of these cycles, he or she will lose because the other player can kill herself or himself faster in G. To kill yourself in G do not use one of the paths to w for 3 turns. Then go to w and from w take a path to one of the vertices already visited in G. After less than 30 turns you are stuck. We will assume that neither player ever goes into one of these cycles, unless there is no other move.

Claim 1: ALICE has to start at the top vertex labeled ALICE START or she will lose.

If ALICE starts at vertex y, BOB kills himself with one turn. If ALICE starts anywhere else, BOB will start at y and kill himself in two turns and win.

Claim 2: BOB has to start at one of the vertices x, w_1, w_2, u_1, or u_2; otherwise, ALICE can kill herself at x.

The moves for ALICE to kill herself are y, w_1, u_1, x. As there are no quick deaths in the remaining graph, ALICE can win this way. This shows Claim 2.

We show that BOB will lose unless he starts at x. We consider the cases that BOB starts at w_1 and w_2. The other cases are similar.

Case 1 BOB starts at w_1. Thereafter BOB has to go to x. ALICE will go to u_2 and BOB has to enter a cycle at u_1 and thus lose.

Case 2 BOB starts at w_2. Thereafter he has to go to w_1. ALICE goes to x and BOB imprisons her by going to u_1.

This shows BOB has to start in x.

Before we exclude all possible moves of ALICE and BOB further, we want to mention already the paths ALICE and BOB have to take, see Figure 3.22 b). BOB goes from x to u_1 and further down to v_2 and ALICE goes to w_1 and further down to v_1. Thus they arrive at the start vertices of G. In the following, we will exclude all other options.

Consider the two *bridge paths* a_1a_2 and b_1b_2 from the left to the right and vice versa. It is clear that those paths cannot be traversed in the order a_2a_1 or b_2b_1. because cycles of length $2n$ are attached to a_1 and b_1. When w_3 or u_4 is already occupied one would need to go into the cycle and loose by the discussion above. Also note that if both players use the bridge paths the player coming from the left will be able to kill itself in w_4. The other player cannot kill itself in u_5 quick enough, because of the long cycle attached to it. The other player cannot kill itself in G quick enough either. Thus we will assume that the player on the left will not go to the right if the player on the right already went to the left or still has this option.

Claim 3: BOB has to go to u_1 and further to v_2.

In case BOB goes to w_1 the sequence of moves, that lead to a win for ALICE is depicted in Figure 3.23 a).

Claim 4: ALICE loses if she uses the edge (y, w_3).

In case ALICE uses the edge (y, w_3) the sequence of moves, that lead to a win for BOB is depicted in Figure 3.23 b).

Claim 5: ALICE loses if she uses the edge (y, u_2).

In case ALICE uses uses the edge (y, u_2) the sequence of moves, that lead to a win for BOB is depicted in Figure 3.23 c) and d).

This shows ALICE and BOB indeed have to move as in Figure 3.22 b), because we excluded all other cases. □

3.5 Conclusion

This chapter deals with the question by how much one player can win over the other and the constructions give an intuition about the conditions when this happens.

The results also suggest that no optimal algorithm can be found to play the game.

From the study of these results, we can learn general techniques how to make reductions for these kind of games. TRON is interesting for its mathematical simplicity and its popularity. The game might be a motivation for students to learn graph theory or getting a better intuition for complexity. But we admit that there are already many examples found for these purposes, which might be more suitable. Another aspect of TRON is its suitability to study the performance of learning algorithms.

However in most computer implementations of TRON you play on an empty grid. Good strategies might be much easier. It remains an open question to get more insight into these situations. In particular, we want to ask: What is the complexity of TRON on a planar graph? NP-hardness and co-NP-hardness follows easily from [20].

The most Fun question we want to pose is about trees. By how much can BOB win on a tree at most. Recall that Lemma 3.4 is not tight. We also think the question about trees is easiest to attack and might be a nice project for a bachelor/master thesis. The techniques used in this chapter will not work. New fresh ideas are needed to solve this problem.

Acknowledgements We thank Justin Iwerks for suggesting this research topic and initial discussions. For proofreading and general advice, I want to thank Rafel Jaume, Tobias Keil, Wolfgang Mulzer, Lothar Narins and Michael Payne. For the visage on the right-hand side of Figure 3.6 I thank Torsten Ueckerdt. I also want to thank the reviewers for all their many comments and apologize for my bad English. Special thanks goes to Günter Rote for a thoroughly reading and for finding a mistake in the construction in the original proof of Theorem 3.25.

Chapter 4

Pareto Optimal Matchings

4.1 Introduction

4.1.1 Definitions

The house allocation problem is motivated by the following setup: a set of people is interested to be allocated to a certain set of houses. Each person has a ranking over the set of houses and wants to be assigned to the house with her highest preference. As soon as two people have the same favorite house this is not possible. Motivated by this picture we abstract the set up and start with some definitions.

In an instance of the house allocation problem two sets A and B are given. The set A represents applicants and the set B represents houses. We denote by m and n the size of A and B respectively. In the house allocation problem, we assume that every $a \in A$ has a preference list over the set B. A preference list can be formally defined as a total order of B. We call an injective mapping τ from A to B a matching. A *blocking coalition* of τ is a subset A' of A such that there exists a matching τ' that differs from τ only on elements of A', and every element of A' improves in τ', compared to τ according to its preference list. If there exists no blocking coalition, we call the matching τ a *Pareto optimal matching* (POM).

We represent the preference lists by an $m \times n$ matrix. Every row represents the preference list of one of the applicants in A, i.e., in a given row r corresponding to some applicant $a \in A$, the leftmost house is the one that a prefers most, etc., house b_1 is left to b_2 in r if and only if a prefers b_1 over b_2. Note that no row contains an element from B twice. We usually denote this matrix by M and following this interpretation we usually denote the applicants of A by $r_1, r_2, \ldots r_m$ and the houses of B by $1, 2, \ldots, n$. Because of this matrix representation, we usually refer to applicants of A only as rows and to houses of B as elements (of the matrix).

To illustrate the notion consider the following matrix and observe that the matching indicated by circles is indeed Pareto optimal.

$$\begin{pmatrix} \textcircled{1} & 5 & 3 & 2 & 4 \\ 3 & 1 & \textcircled{4} & 5 & 2 \\ 1 & \textcircled{3} & 5 & 4 & 2 \end{pmatrix}$$

The image set of τ corresponds to the set of houses of B in these positions. Thus, we say that τ *selects* some position p of M (resp. some element b of B), if p is in τ (resp. b is in the image set of τ). Similarly, we say that a row a *selects* a position P in row a (resp. element b) if this holds for the matching τ under consideration.

In a POM the positions after the m-th column will never be assigned, because at least one of the previous m elements in that row is preferred and not assigned to any other element on A. Therefore it is sufficient to consider only $m \times m$ square matrices.

If some POM τ assigns p (resp. b), then it is a *reachable* position (resp. *reachable* house). More generally, a set $E \subseteq B$ is *(exactly) reachable* if there exists a POM τ with $E \subseteq s(\tau)$ ($E = s(\tau)$). In this case we also say that τ reaches E. An element b is *unavoidable* if it belongs to the set $s(\tau)$ for every Pareto optimal matching τ of M and *avoidable* otherwise. A set E is *avoidable* if there exists a POM τ with $s(\tau) \cap E = \varnothing$. Note that for a set $|E| = m$ it is exactly reachable if and only if $B \setminus E$ is avoidable. We will also study matrices with fewer than m columns, precise definitions will be given in Subsection 4.1.4. In this case preference lists are shorter and it can happen that some elements of A are not assigned.

4.1.2 Results

Enumerating reachable elements and sets

In Section 4.2 we deal with enumerative problems related to reachable elements. Our main result here is the following.

Theorem 4.1. *Let M be an $m \times m$ matrix and E^* be the set of all reachable elements. Then*

$$|E^*| \le \sum_{i=1}^{m} \lfloor m/i \rfloor \le m(\ln m + 1).$$

This improves the trivial upper bound of m^2 which appears in [37]. In [37] the authors also showed a lower bound construction which has asymptotically as many reachable elements as is implied by our upper bound. Thus Theorem 4.1 is asymptotically tight.

Denote by $\mathcal{E}(M)$ the family of all (exactly) reachable m-element sets of M. For example, if all the elements in the first column of M are distinct, then $|\mathcal{E}(M)| = 1$. With Theorem 4.1 we can bound $\mathcal{E}(M)$.

Corollary 4.2. *For any matrix M, we have $|\mathcal{E}(M)| \le \binom{m(\ln m+1)}{m}$.*

This is the only non-trivial upper bound that we found, improving $\binom{m^2}{m}$ of [37]. As an important consequence, our upper bound also improves the upper bound on the pattern matching problem regarded in [37]. The best known lower bound is $\binom{m}{\lceil m/2 \rceil}$ [37]. The construction in that paper is a matrix where in the first $\lfloor m/2 \rfloor$ columns the i-th column contains only element i and in the $(\lfloor m/2 \rfloor + 1)$-st column there are m different elements which are also all different from $1, 2, \dots \lfloor m/2 \rfloor$.

Characterization of avoidable elements and sets

Section 4.3 concentrates on the notion of avoidable elements. Let x be the element suspect to be avoidable. Given some set of rows R we denote by $E_x(R)$ the set of elements left of x in the rows R (i.e., y is in $E_x(R)$ if and only if there exists a row $r \in R$ in which y appears to the left of x; if x does not appear in r then all elements in r are regarded to be left of x).

Theorem 4.3. *An element x of a matrix M is avoidable if and only if for every set R of rows of M, we have:*

$$|E_x(R)| \geq |R|$$

Extremal results and algorithmic results in connection to avoidable elements are included in Section 4.3, as they follow from the proof of Theorem 4.3. We prove that:

Corollary 4.4. *Deciding if an element b is avoidable can be done in $O(m^2\sqrt{m+n})$ and also in $O(m^3)$ time. Listing all unavoidable elements can be done in $O(m^2 n\sqrt{m+n})$ and also in $O(m^5)$ time.*

Both results follow from an easy reduction to matchings in a bipartite graph.

Complexity of reachability

Computational questions about reachable elements are considered in Section 4.4. We considered all reasonable computational questions connected to the notions we considered. The problems are defined as follows:

Problem 1. *(Deciding Reachability)*
Input: *A matrix M and a set $D \subseteq B$.*
Question: *Is D reachable?*

Problem 2. *(Deciding Exact Reachability)*
Input: *A matrix M and a set $E \subseteq B$*
Question: *Is E exactly reachable?*

Problem 3. *(Counting Reachable Sets)*
Input: *A matrix M.*
Question: *How many sets $D \subseteq B$ are reachable?*

Problem 4. *(Counting Exactly Reachable Sets)*
Input: *A matrix M.*
Question: *How many sets E are exactly reachable?*

Problem 5. *(Counting Exactly Reachable Supersets)*
Input: *A matrix M and some set $D \subseteq B$.*
Question: *How many sets E with $D \subseteq E \subseteq B$ are exactly reachable?*

The next table summarizes our findings about algorithmic questions. The general case is always the same as with 3 column matrices. Problems 1 and 2 are already complete if D contains exactly 1 element. It was already observed by Henze, Jaume and Keszegh that Problem 1 is NP-complete [37]. Our contribution among others is to show NP-completeness also for matrices with only 3 columns. The running time of Problem 3 follows from the discussion in Section 4.3 and Corollary 4.4.

Problem	2 columns	proof	3 columns	proof
1)	polynomial	(Thm 4.12)	NP-complete	(Thm 4.9)
2)	polynomial	(Cor 4.4)	polynomial	(Cor 4.4)
3)	explicit formula	(Thm 4.13)	?	
4)	#P-complete	(Thm 4.14)	#P-complete	(Thm 4.14)
5)	#P-complete	(Thm 4.14)	#P-complete	(Thm 4.9)

It remains an open question whether Problem 3 is hard for general matrices. We conjecture it is already #P-complete for 3 column matrices.

Problem 4 is a special case of Problem 5 for the case that $D = \varnothing$.

4.1.3 Motivation and related work

One-sided matchings have natural practical uses, e.g. consider the house-allocation problem where the set A consists of people and the set B consists of houses, see for instance [1].

A recent book on matchings under preferences is by David Manlove [51]. In this chapter we tried, whenever applicable, to follow the notation therein.

A field that evidently seems to be related to our topic is that of stable matchings. This field is very broad and belongs to economic game theory. The seminal work Gale and Shapley is the starting point for this field [31]. Some work in this field and different variations of the problem can be found in the phd thesis of Sandy Scott [64], recent papers can be found in the proceedings of the Second International Workshop on Matching Under Preferences called MATCH UP [17]. In these works there are many different concepts of preferences and stability and they ask for efficient computable solutions that maximize the outcome for the participants in one way or the other. Readers interested more broadly in the topic of algorithmic game theory are referred to the book edited by Nisan, Roughgarden, Tardos and Vazirani [57].

In contrast to most research done in these areas, our question is more combinatorial in nature. The underlying algorithmic question of computing a Pareto optimal matching is trivial. Thus, instead of existence questions, rather the enumerative questions become interesting. However, for the original definition of stability many authors have tried to upper and lower bound the number of stable matchings and some combinatorial structures have been unfolded. See [51, Section 2.2.2] for an overview of results in this direction.

Further some complexity results similar to ours have been found earlier. The first dates back to 2005 [2]. Their main result is an efficient algorithm to compute a POM with maximum cardinality. Here the preference lists are incomplete. Further they show hardness to compute a minimum maximal POM. This result has already some ideas of the proof of Theorem 4.3. Further the proof of Theorem 4.14 implies hardness of computation about minimum maximal matchings. Although they show an easy 2-approximation, it is open whether there exists a PTAS for a minimum maximal POM.

We are aware of 4 more papers that considered similar results to our complexity results [63] [13] [14] [21]. All appeared in 2013 three of them in December. Their main motivation is to study the behavior of the randomized serial dictatorship also called randomized priority allocation. The randomized serial dictatorship picks a permutation at random and thereafter computes the corresponding greedy matching.

The first is by Saban and Sethuraman [63]. Their results, reformulated in our context, is NP-hardness of Problem 1, for arbitrary matrices. Aziz, Brandt and Brill [13] show #P hardness for a variant of Problem 5 for arbitrary matrices. We improve these results, as we can show this holds also for matrices with only 3 columns. Aziz and Meske show that constraint versions are solvable in polynomial time [14]. At last Cechlárová et al. [21] consider a generalized setting. However they show NP-hardness of computing a minimum maximal matching even for matrices with 2 columns by an elegant reduction from vertex-cover very similar in spirit to the proof in Theorem 4.14.

Another important connection is that this work is originally motivated by a work that was presented at the EuroCG 2012 in Braunschweig [37]. The authors considered a generalisation of Voronoi diagrams under the assumption that not just one point, but many points are matched injectively in a way that minimizes the sum of the squares of distances between matched points. From the definitions in their paper, the Pareto optimality comes as a natural property. They asked explicitly for the number of exactly reachable sets, as it gives an upper bound on the number of Voronoi cells in the above setting. Motivated by this, they gave lower and upper bounds on the number of exactly reachable stable sets. To do this, first they gave lower and upper bounds for the number of reachable elements. In this chapter we improve their upper bound for the number of reachable elements and by that we prove that their lower bound is asymptotically correct. This also yields a significant improvement on the previous upper bound on the number of exactly reachable stable sets, although in this case our new upper bound still does not meet the lower bound they had.

Their work is based on a work by Rote presented at the EuroCG 2010 (2 years earlier) in Dortmund [61].

4.1.4 Preliminaries

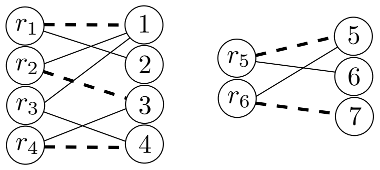
As we also want to study matrices with fewer than m columns, we need to define what we mean by a matching under these assumptions. There are two equivalent ways. First we could say that every row, for which all elements are already picked by other rows just do not get assigned to anything. A nicer way is to add columns, with all elements in one column being the same and not appearing before. If we want to know if some set E is exactly reachable in the first way, we construct E' from E by adding the elements from the first $m - |E|$ additional columns (and vice versa). The following is an example of a 2 column matrix.

$$\begin{pmatrix} 1 & 4 \\ 2 & 1 \\ 2 & 5 \\ 4 & 3 \end{pmatrix} \sim \begin{pmatrix} 1 & 4 & c_1 & c_2 \\ 2 & 1 & c_1 & c_2 \\ 2 & 5 & c_1 & c_2 \\ 4 & 3 & c_1 & c_2 \end{pmatrix}$$

We use the first approach. However, using the second approach, some hardness results will carry over from 2 or 3 column matrices to k column matrices ($2 \leq k \leq m$). In such a case, we will point this out again at the appropriate places.

To see the correspondence between matchings in a graph theoretical sense and in our context we define the *bipartite row element graph* G as follows. The vertices are defined as the set of rows and elements; an element e is adjacent to some row r if and only if e appears in r. See an example for the special case of a matrix with only 2 columns.

$$\begin{pmatrix} \textcircled{1} & 2 \\ 1 & \textcircled{3} \\ 1 & \textcircled{4} \\ 3 & 4 \\ \textcircled{5} & 6 \\ \textcircled{7} & 6 \end{pmatrix}$$



The circled POM corresponds to the dashed matching on the right side.

If there is no blocking coalition of size $\leq i$, we call the matching an *i-Pareto optimal matching* (i-POM). In particular this implies that every POM is an i-POM. We call a matching 1-*POM* if there is no blocking coalition of size one. The next matching is one-Pareto optimal but not Pareto optimal.

$$\begin{pmatrix} 1 & \textcircled{5} & 3 \\ 5 & 1 & \textcircled{4} \\ 5 & \textcircled{1} & 1 \end{pmatrix}$$

Observation 4.5. *If τ is a POM and τ selects position p in row a, then τ selects every element that appears in row a left of p.*

This observation also holds for 1-POMs. A matching τ is *greedy* if there exists a permutation π of A such that the matching can be generated in the following manner: we process the rows of M in the order determined by π, and in each row we pick the leftmost element that was not picked earlier. Given some permutation π we call the corresponding greedy matching τ_π.

Lemma 4.6 brings all the introduced notions together, showing that POM, 1-POM and greedy matchings select exactly the same sets. The equivalence of POM and greedy matchings was already proved in [37].

For the proof of the next lemma we need an equivalent definition of POM. We say τ_1 is *better than* τ_2 if every element $a \in A$ is matched in τ_1 to a better or equally good element than in τ_2, according to the preference list of a. And we require $\tau_1 \neq \tau_2$. Note POM are exactly the maximal elements to this *better* relation.

Lemma 4.6. *Let $E \subseteq [n]$ with $|E| = m$. The following statements are equivalent.*

1. *E is (exactly) reachable, i.e. there exists a POM τ with $s(\tau) = E$.*
2. *There exists a permutation π such that for the greedy matching τ_π we have $s(\tau_\pi) = E$.*
3. *There exists an one-Pareto optimal matching (1-POM) τ with $s(\tau) = E$.*

Proof. [1 $\Rightarrow$ 2] Let τ be a POM matching such that $s(\tau) = E$. We construct a permutation π inductively. If possible take as the next row, in the order of our permutation, the one that has a position of τ in its first entry. Delete the element a at this position from all other rows and continue. We show that at each stage there must be such a row. For the purpose of contradiction assume such a row does not exist. Take any row, denoted by q_1 and let e_1 be some element which is left to the element selected by τ in row q_1. Because τ is Pareto optimal, there exists some row r_2 selecting e_1. Let e_2 be any element left to e_1 in row r_2. In this way we can define a sequence (e_i) and (r_i). As we have only finitely many elements, at some point we get a first e_j that appears earlier in the sequence $e_i = e_j$, $i < j$. This implies that in the rows $r_i, \dots r_j$ we can improve simultaneously (i.e., it is a blocking coalition), which is a contradiction to the assumption that τ is Pareto optimal.

[2 $\Rightarrow$ 3] As every row picks the best element, not yet selected, it is clear that no single row can improve.

[3 $\Rightarrow$ 1] Let τ_0 be some 1-Pareto optimal matching and $E = s(\tau_0)$. Observe that all the elements left to the elements picked by τ_0 are in E. The set of matchings that are better or equal to τ_0 is non-empty as it contains τ_0 and the set is of course finite, so there exists a best matching τ_1 among them, i.e. one for which there is no better matching. This must be a POM and by Observation 4.5 $s(\tau_1) \subseteq s(\tau_0) = E$, and the size of $s(\tau_1)$ is also m. This implies $s(\tau_1) = E$. $\square$

Note that this lemma implies that also for any i, i-POMs select the same sets as POMs/1-POMs. Note also that the proof of Lemma 4.6 implies that actually every POM matching is greedy. The inverse is also true and left as an exercise, as we will not use it later.

4.2 Counting reachable elements and sets

We start with a discussion of reachable positions. For every row r, there exists a reachable position p_r furthest to the right in that row, we call such a position *last reachable.* However note, that not all positions left of the last reachable position must be reachable. Consider the following matrix together with the matching τ indicated by circles.

$$\begin{pmatrix} \textcircled{5} & 4 & 3 & 2 \\ 5 & \textcircled{1} & 6 & 7 \\ 1 & \textcircled{2} & 8 & 9 \\ 2 & 1 & 5 & \textcircled{4} \end{pmatrix}$$

The matching τ is a Pareto optimal and thus the circled position in the bottom row with element 4 is the last reachable position in that row. However, it is easy to check, that the two positions left to this circled position (with elements 1 and 5) are not reachable.

Theorem 4.1. *Let M be an $m \times m$ matrix and E^* be the set of all reachable elements. Then*

$$|E^*| \leq \sum_{i=1}^{m} \lfloor m/i \rfloor \leq m(\ln m + 1).$$

The proof of the theorem uses the following lemma.

Lemma 4.7. *Let T be some set of k POMs. We denote by $E(T)$ the set of elements reached by at least one POM of T. Then*

$$|E(T)| \leq \sum_{i=1}^{k} \lfloor m/i \rfloor.$$

proof Lem. 4.7. The proof goes by induction on k. The base case $k = 1$ is true as one POM selects exactly m different elements.

Consider now a set T of $k \geq 2$ POMs and the set of positions reached by T. Among these positions we denote by p_i the position furthest to the right in row i and we denote $F = \{p_1, \ldots, p_m\}$. We say that an element e (resp. position p) is uniquely reachable by some τ if τ is the only POM in T that reaches e (resp. selects p). Consider the set $G \subseteq F$ of those rightmost reachable positions that are reachable by exactly one POM of T. By the pigeon-hole principle there exists a POM τ in T that reaches at most $1/k$ portion of G. Denote the set of elements in these positions by H ($|H| \leq \lfloor m/k \rfloor$).

By the definition of H all elements $s(\tau) \setminus H$ are not selected uniquely by τ, i.e. some other matching of T also selects it.

Let us explain this in more detail. Consider and element $e \in s(\tau) \setminus H$ and denote by p the position of e. As $p \notin G$ there exists another matching $\tau' \neq \tau$ that either also selects p or τ' selects a position further to the right of p in the same row. In both cases we have: τ' must select e by Observation 4.5.

Thus the rest of the reached elements are also reachable by $T - \tau$. By induction we get

$$E(T) \leq E(T \setminus \tau) \ + \lfloor m/k \rfloor \leq \left(\sum_{i=1}^{k-1} \lfloor m/i \rfloor \right) \ + \ \lfloor m/k \rfloor = \sum_{i=1}^{k} \lfloor m/i \rfloor .$$

□

proof Thm.4.1. Let τ_i be a POM selecting the last reachable position p_i in row i ($1 \leq i \leq m$) (these matchings are not necessarily different.).

Let e be some element that can be reached by some POM. We show e is selected by one of the POMs $\tau_1, \dots, \tau_m$. Indeed, if e is at some last reachable position then this is clear. Otherwise, e appears in some row r not at the last position p_r. By Observation 4.5, e must be picked by τ_r. Thus the matchings $\tau_1 \dots, \tau_m$ reach together all reachable elements. As $\tau_1 \dots, \tau_m$ are m POMs, the first inequality follows from Lemma 4.7. Finally, it is well-known that the harmonic series is bounded by $\ln m+1$, thus the second inequality holds as well. □

Asymptotic tightness of Theorem 4.1 follows from the following construction by Henze, Jaume and Keszegh [37].

Example 4.8 ([37]). *For each k, a matrix M_k with $m = 2^k$ rows and $(m/2)\log 4m = (k+2)2^{k-1}$ reachable elements is constructed recursively as follows.*

$$M_0 = \left(\ 1 \ \right);$$

and, for $k \geq 0$,

$$M_{k+1} = \left(\begin{array}{c|c} 1 & \\ \vdots & M'_k \\ 2^k & \\ \hline 1 & \\ \vdots & M''_k \\ 2^k & \end{array} \right),$$

where M'_k and M''_k are relabelings [1] of M_k with no common element and all elements different from $1, 2, \dots, 2^k$. The undefined entries of the matrix can be filled arbitrarily.

4.3 Characterization of avoidable elements

In this section we characterize avoidable elements and sets. Recall that we define $E_x(R)$ as the set of elements left of x in the rows of R (i.e., y is in $E_x(R)$ if and only if there exists a row $r \in R$ in which y appears to the left of x; if x does not appear in r then all elements in r are regarded to be left of x)

Theorem 4.3. *An element x of a matrix M is avoidable if and only if for every set R of rows of M, we have:*

$$|E_x(R)| \geq |R|$$

[1] A matrix M' is a *relabeling* of a matrix M if there is a bijective function between the elements (not positions!) of M and M' such that applying this function to the elements in all the positions of M we get M'. Clearly two matrices that are relabelings of each other are equivalent from our perspective.

Proof. [$\Rightarrow$] Let τ be a POM which does not pick x and let R be a set of rows. In each row a different element is picked by τ, which is left of x. This shows the claim.

[$\Leftarrow$] W.l.o.g. x is present in all the rows. Consider the bipartite graph on $A \cup B$, defined by all pairs $(a, b) \in A \times B$ such that b appears in row a before x. The above condition says, that for all subsets $R \subset A$ the neighbourhood of R is larger or equal to R in terms of size.

By Hall's theorem, there exists a matching τ that picks elements to the left of x. We can modify τ so that each row picks an element farthest to the left in M not chosen by any other row. In other words τ is a 1-POM. By Lemma 4.6 there is a POM τ' selecting the same set of elements as τ, thus τ' does not choose x and so x is avoidable. $\square$

Corollary 4.4. *Deciding if an element b is avoidable can be done in $O(m^2\sqrt{m+n})$ and also in $O(m^3)$ time. Listing all unavoidable elements can be done in $O(m^2 n\sqrt{m+n})$ and also in $O(m^5)$ time.*

Proof. For an element x to be avoidable is equivalent to the existence of a matching that connects all the elements of A with elements left to x. Thus, we need to find a maximum bipartite matching. The fastest known algorithm (Hopcroft-Karp) checks this in $O(\sqrt{V}E)$ time. In our case $|V| \leq m+n$, $|E| \leq m^2$. Thus, it can be checked whether x is avoidable in $O(m^2\sqrt{m+n})$ time. Consequently, listing of all unavoidable elements by checking all the elements of B can be done in $O(m^2 n\sqrt{m+n})$ time. The bounds without n follow from the fact that we can limit the number of relevant elements in B easily to m^2 (the number of elements in the first m columns of the matrix). $\square$

Recall that a set X is said to be *avoidable*, if there exists a matching that avoids every element of X. Note that it is possible that all elements of a set are avoidable, while the set itself is not. Theorem 4.3 extends to set X, by replacing all elements of X by the single element x. If x is avoidable, then X was avoidable and vice versa.

4.4 Complexity of reachability

In this section we show that Problems 1 to 5 defined in the introduction lie in the indicated complexity classes.

That Problem 2 is polynomially solvable is seen as follows: let M be a matrix and E be a set of m elements occurring in the matrix. We define $F := B \setminus E$, then E is exactly reachable if and only if F is avoidable, which can be checked in polynomial time by Corollary 4.4.

We start with a brief and informal explanation of the class #P,. A detailed introduction is given by Arora and Barak [8].

Informally #P is a class of counting problems. Counting problems get an object and count some other objects satisfying some condition. The only restriction is that the size of the binary representation of the number of objects counted is polynomial in the input. But this is given in all that follows, because we have exponential upper bounds on the number of objects to be counted.

Famous examples of #P-complete problems are:

- How many variable assignments does a given boolean formula satisfy?
- How many perfect matchings does a given bipartite graph have?

Note that the decision version of the first problem is NP-hard and of the second problem is in P.

A problem L is defined to be #P-complete if it lies in #P and there exists a polynomial time counting reduction from all other problems in #P to L. Due to transitivity of counting reductions, it is sufficient to reduce one #P-complete problem to L.

At last a counting reduction transforms the input of one problem to the input of a second problem in a way that the counting problem for the first problem can be solved via the reduction and the solution to the second problem. It will be very clear in our context, that our reductions are indeed counting reductions.

We will use the result from Creignou and Hermann [24] that #1-in-3-SAT is #P-complete. Here it is asked how many assignments exist such that every clause of a given 3CNF-formula has exactly one true and two false literals. We will also use later, that the corresponding decision problem is NP-complete.

We start our discussion with Problems 1 and 5 for 3-column matrices.

Theorem 4.9. *Problem 1 (Deciding Reachability) is NP-complete and Problem 5 (Counting Exactly Reachable Supersets) is #P-complete, even when D (the set we want to decide if it is reachable) is a 1-element set and the matrix has only 3 columns.*

We will give a transformation that converts 1-in-3-SAT formulas ϕ to matrices M with the special property that the number of good assignments for ϕ equals the number of reachable sets containing some special element x. This transformation will reduce 1-in-3-SAT to Problem 1 and #1-in-3-SAT to Problem 5 and thus proves the essential part of Theorem 4.9.

$$M_\Phi = \begin{pmatrix}
a_1 & x_1 & * & * & & \\
a_1 & \overline{x_1} & * & * & & \\
\vdots & \vdots & & & & \\
a_n & x_n & * & * & & \\
a_n & \overline{x_n} & * & * & & \\
& & & & & \\
b_1 & c_1 & L_1^1 & \overline{L_2^1} & \overline{L_3^1} & C_1 \\
b_1 & c_1 & L_2^1 & \overline{L_1^1} & \overline{L_3^1} & C_1 \\
b_1 & c_1 & L_3^1 & \overline{L_1^1} & \overline{L_2^1} & C_1 \\
& & & & & \\
& & \vdots & \vdots & & \\
& & & & & \\
b_m & c_m & L_1^m & \overline{L_2^m} & \overline{L_3^m} & C_m \\
b_m & c_m & L_2^m & \overline{L_1^m} & \overline{L_3^m} & C_m \\
b_m & c_m & L_3^m & \overline{L_1^m} & \overline{L_2^m} & C_m \\
& & & & & \\
C_1 & C_2 & \dots & & C_m & x
\end{pmatrix}$$

Figure 4.1: Illustration of the matrix M_Φ

Lemma 4.10. *Let Φ be an instance of 1-in-3-SAT. Then there exists a matrix M with some element x such that the 1-in-3-satisfying truth assignments of the variables of Φ are in one-to-one correspondence with the exactly reachable sets E with $x \in E$.*

Proof. Let Φ be of the form $C_1 \wedge C_2 \wedge \ldots \wedge C_m$. And each of the clauses C_i is of the form $L_1^i \vee L_2^i \vee L_3^i$, where each literal L_j^i is one of the variables $x_1, \ldots, x_n$ or its negation. The matrix M_Φ is defined in Figure 4.1.

Here a_i, b_i, c_i are only used where indicated. The asterisks mark that any element could stand there. Every x_i and its negation get an element named after them. The same holds for every clause C_i. The literals L_i^j need to be replaced by the corresponding variable element.

Let $F : \{x_1, \ldots, x_n\} \longrightarrow \{0,1\}$ be an assignment of the variables such that exactly one literal of each clause in Φ becomes true. We construct the corresponding matching τ_F. The matching τ_F selects x_i whenever F gives it the value 1 and its negation otherwise.

There exists exactly one way to select the element C_i for each i. To see this consider as an example $C = x_1 \vee x_2 \vee x_3$ and assume w.l.o.g. that x_1 is true and x_2 and x_3 are false in assignment F. Then τ_F has already selected $x_1, \overline{x_2}, \overline{x_3}$. Thus C can be selected in the following way, indicated by circles:

$$\begin{pmatrix} b & c & x_1 & \overline{x_2} & \overline{x_3} & \textcircled{C} \\ b & \textcircled{c} & x_2 & \overline{x_1} & \overline{x_3} & C \\ \textcircled{b} & c & x_3 & \overline{x_1} & \overline{x_2} & C \end{pmatrix}$$

In the same way all clause labels can be selected and it is possible to select x.

In this manner we get from every satisfying assignment F a matching τ_F selecting x. It is easy to see that τ_F is a 1-POM. And thus there also exists a POM selecting the same set. In this way we get from every satisfying truth assignment a 1-POMs selecting x.

We show that we can get a satisfying truth assignment for every 1-POM τ selecting x. We get a truth assignment by taking x_i to be true if x_i is selected in the first $2n$ rows and x_i false if $\overline{x_i}$ is selected. τ will always select one of x_i and $\overline{x_i}$. Clearly, $C_1, \ldots, C_m$ are not taken by the last row. The only way to select C_i is if one of its three literals is assigned to be true and the other two are assigned to be false. Thus this assignment is an 1-in-3-satisfying assignment. $\square$

Let M be some matrix with the element x. We define a matrix M' with exactly three columns by replacing each row by a block. Let $(a_1 a_2 \ldots a_m)$ be some row, with $m > 3$. We replace it by the following block:

$$\begin{pmatrix} a_1 & a_2 & \beta_1 \\ \beta_1 & a_3 & \beta_2 \\ \beta_2 & a_4 & \beta_3 \\ & \vdots & \\ \beta_{m-4} & a_{m-2} & \beta_{m-3} \\ \beta_{m-3} & a_{m-1} & a_m \end{pmatrix}$$

The β_i elements are different for every block. If τ be a Pareto optimal matching of M, then it is easy to see that there exists a unique Pareto optimal matching τ' in M' that selects in each block the same element as τ in M. In every block at most one element of the original row can be selected by any POM τ'. Every POM τ' in M' that selects in every block exactly one element of the original row corresponds to exactly one POM τ of M. Consider the three column matrix M' of the matrix M in Lemma 4.10. Clearly, the POMs selecting x in M stand in a one to one correspondence with the POMs in M' selecting x. The next lemma summarizes our discussion.

Lemma 4.11. *Let Φ be some boolean 3DNF formula. Then there exists a* ***three column*** *matrix M with some element x such that the 1-in-3-satisfying truth assignments of the variables of Φ are in one-to-one correspondence with the exactly reachable sets E with $x \in E$.*

proof Thm 4.9. #1-in-3SAT is #P-complete. Given a 3-DNF formula Φ, the reduction above gives a 3 column matrix, such that the exactly reachable sets containing x are in a 1-to-1 correspondence with the satisfying assignments. Thus also Counting Exactly Reachable Supersets is #P-hard. And as 1-in-3SAT is also NP-complete, deciding Reachability is also NP-hard. Membership to these classes (and thus completeness) follows from the exponential upper (2^{m^2}) bound on the number of possible sets. Thus the binary representation is linear in the size of the input. □

The natural question to ask is, what happens, when there are only **two columns**. Understanding the structure of reachability for 2 column matrices gives many interesting results at once. We consider first Problem 1 (Deciding Reachability) and will see easily from the algorithm, that we can find an explicit way to count all reachable sets, for 2 column matrices. The complexity of counting reachable sets for general matrices remains open. On the other hand, we will see that it is #P-complete to count all exactly reachable sets E already for 2-column matrices.

Theorem 4.12. *Problem 1 (Deciding Reachability) is in P for 2 column matrices.*

Proof. We assume the bipartite row element graph G is connected, otherwise we treat each component separately. Let m denote the number of rows, D the set of elements we wish to reach, and X the set of elements of G.

It is easy to see, by induction on m, that the number of different elements in G is at most $m+1$. (This is true for $m = 1$ and at most 1 new element is added when a new row is added.)

We distinguish two cases.

Case (1) $|X| = m + 1$

Case (2) $|X| \leq m$

In the first case we cannot reach X but we will show that for every avoidable element $x \in X$ we can reach $X \setminus \{x\}$. This implies D is reachable if and only if it does not contain all avoidable elements of this component.

In the second case we will show that X (and subsequently D) can be selected.

In the first case, there is one more element than rows and thus it is clear, that at least one avoidable element cannot be selected. Further, the number of vertices is $2m + 1$ and the number of edges is $2m$. This implies that G is a tree. Let x be any avoidable element not in D. We orient all edges away from x in G. Clearly the in-degree of every element is exactly one. For every edge oriented from row r to element e let r select e. As x is avoidable it is not in the first column. As all elements, except x is selected there is no 1-blocking coalition. Thus the described matching is a 1-POM, by Lemma 4.6 there is a POM selecting $D \subseteq X \setminus x$.

Consider now case 2. As every row is incident to exactly 2 elements the number of edges is exactly $2m$ and as the number of vertices is at most $2m$ G contains a cycle. Let $r_0, e_0, r_1, e_1, \ldots, r_t, e_t$ be our cycle. We assign r_i to e_i. In this way all elements on the cycle are selected.

We will repeat in the following. Pick a row r that has at least one of the elements already selected. Such a row exists because we assume that G is connected. There is at most one element which can be selected by r. Let r select this element if possible.

This procedure leads to a matching τ that selects all elements and thus τ is automatically 1-POM, by Lemma 4.6 there is a POM selecting $D \subseteq X$.

All the above steps are computationally easy. □

Let M be a 2 column matrix. We assume the corresponding bipartite row element graph has k components. Denote with a_i the number of avoidable elements E_i of the ith component and u_i the number of unavoidable elements F_i. Further χ_i is the indicator variable for the event that the ith component is a tree. Recall that every subset of a reachable set is reachable.

Theorem 4.13. *With the notation and assumptions above:*

$$|\{E : E \text{ is reachable } \}| = \prod_{i=1}^{k} (2^{a_i} - \chi_i)\, 2^{u_i}$$

This implies Problem 3 (Counting Reachable Sets) is in P for 2 column matrices.

Proof. It is clear that we can look at each component of the bipartite row element graph separately. Consider the case that component i is a tree. We know from the proof of Theorem 4.12 that every proper subset of E_i is reachable, i.e., at least one element must not be chosen. These are $2^{a_i} - 1$ subsets. From the unavoidable elements all subsets can be chosen. Consider now the case that component i has a cycle. Then we know that all elements from this component can be chosen at once. This leads to $2^{a_i+u_i}$ many reachable subsets. As the choice is independent for each component, we take the product. □

Theorem 4.14. *Problem 4 (Counting Exactly Reachable Sets) is #P-complete already for 2-column matrices.*

We will show #P-hardness for 2-column matrices. By adding $k-2$ columns each consisting of a single new element we get a k-column matrix with the same number of exactly reachable sets. Thus the problem is also #P-hard for k-column matrices with $k \geq 2$.

Proof. We reduce the problem from the #Independent Set Problem. #Independent Set Problem is the problem of counting the number of independent sets in a given graph and it is #P-complete. Membership to #P follows from the exponential bound (2^{m^2}) on the number of exactly reachable sets. More on the complexity of counting independent sets in a graph can be found in [68, 33].

Let $G = (V, E)$ be some connected graph. We construct a two column matrix M_G with the property that non-empty independent sets are in one-to-one correspondence with exactly reachable sets. The elements of M_G are the edges and the vertices of G. For each edge $e = (u, v)$ we insert two rows (e, u) and (e, v). This implies, that the edges are unavoidable and for every edge e at most one of its vertices will be selected in the two rows corresponding to e.

Let $W \subseteq V$ be some non-empty independent set. The set X_W is defined as $X_W = E \cup V \setminus W$. We show: X_W is exactly reachable. We do this in three small steps. First we define an orientation on G. Then we define a permutation π on M_G. At last we will argue that the greedy matching τ_π defined by π selects exactly X_W. Let T be some spanning tree of G and let x be any element in W. We orient every edge away from x in T, to get an orientation O_1 on T. Note except for x every vertex has indegree 1 in T.

We construct O_2 from O_1 by changing the orientation of each edge incident to W such that the in-degree of every vertex in W is zero and orient every remaining edge in G arbitrarily. For this orientation we have: $v \in W$ if and only if $indegree(v) = 0$.

Let π be any permutation such that the following holds: for every edge $e = (u, v)$ oriented towards v, row (e, u) is before (e, v) in π. This implies that the vertices not selected by τ_π are exactly those with in-degree zero. This shows X_W is exactly reachable.

Conversely, let X be some exactly reachable set. All edge elements are unavoidable, thus are in the set X. Further the vertex elements W not selected form an independent set in the graph. To see this observe that two adjacent vertex elements cannot be avoided simultaneously. Note that W can be the empty.

We have shown that every non-empty independent set W in G corresponds to an exactly reachable set X in M_G. Note that the empty set is also independent in G and would correspond to $X = E \cup V$. However $E \cup V$ is not necessarily reachable. We can easily test if $E \cup V$ is exactly reachable. (We leave it as an exercise to check that this is the case whenever G is a tree.)

Thus the number of exactly reachable sets equals the number of independent sets, maybe minus 1. And we can determine efficiently if this "minus 1" is the case. $\square$

This last theorem also implies #P-hardness of Problem 5 (Counting Exactly Reachable Supersets) for 2 column matrices with $D = \varnothing$.

Acknowledgments We want to thank Matthias Henze and Rafel Jaume for posing Theorem 4.1 as an open problem. We also want to thank Rob Irving, Ágnes Cseh and David Manlove for helping us to find related work to our problem. Special thanks goes to Nieke Aerts for enjoyable and interesting discussions on attempts to improve Corollary 4.2.

Chapter 5

Unique Bichromatic Matchings

5.1 Introduction

5.1.1 Basic notation and definitions

Let F be a set of n blue points and n red points in the plane, such that the whole set is in general position (that is, no three points of F lie on the same line). Throughout the paper, such sets will be referred to as *bichromatic sets.* A *perfect bichromatic non-crossing straight-line matching* of F is a perfect matching of points of F realized by non-crossing straight segments, where each segment connects points of different colors. In many sources, such matchings are referred to as *BR-matchings.* In order to simplify the notation and the drawings, we shall instead color the points of F white and black and denote them by $\circ$ and $\bullet$.

It is well known that any bichromatic set has at least one BR-matching. One easy way to see this is to use recursively the Ham-Sandwich Theorem; another way is to show that the bichromatic matching that minimizes the total length of segments is necessarily non-crossing. The main goal of our work is to characterize bichromatic sets with *exactly one* BR-matching. On the way to answering this modest-looking question, we will study several related issues. For our main characterization of unique BR-matchings (Theorem 5.20) we give, besides an elementary proof by contradiction, another proof that puts more structure on the problem in the form of the so-called Fishnet Lemma (Lemma 5.22), which might be of independent interest (Section 5.3.2).

In what follows, M usually denotes a BR-matching.

Definition 5.1. *We say that a BR-matching M is a unique matching if it is the unique BR-matching of F, the bichromatic set of its endpoints.*

The convex hull of F will be denoted by $\mathrm{CH}(F)$, and its boundary by $\partial\mathrm{CH}(F)$. Consider the circular sequence of colors of the points of F that lie on $\partial\mathrm{CH}(F)$; a *color interval* is a maximal subsequence of this circular sequence that consists of points of the same color. For example, in Figure 5.1 (a), $\partial\mathrm{CH}(F)$ has four color intervals: two $\circ$-intervals (of sizes 1 and 2) and two $\bullet$-intervals (of sizes 2 and 3).

In order to state our main results, we need the notion of a chromatic cut.

Definition 5.2. *A* chromatic cut *of M is a line ℓ that crosses two segments of M so that their $\bullet$-ends are on different sides of ℓ (ℓ can cross other segments of M as well).*

For example, the lines ℓ_1 and ℓ_2 in Figure 5.1 (a) are chromatic cuts. The matchings in Figure 5.1 (b) and (c) have no chromatic cuts. Aloupis, Barba, Langerman, and

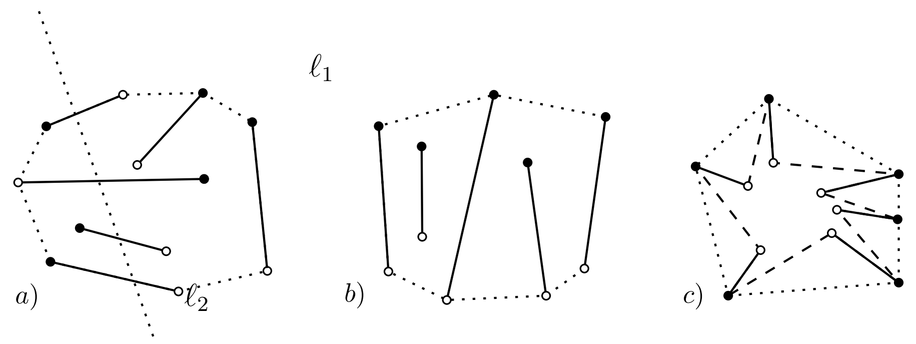


Figure 5.1: (a) A matching with chromatic cuts. (b) A linear matching. (c) A circular matching. Another matching for the same point set is indicated by dashed lines.

Souvaine [7, Lemma 9] proved that a BR-matching M that has a chromatic cut cannot be unique. (They actually proved a stronger statement: in such a case there is a BR-matching $M' \neq M$ such that M' is *compatible* to M, which means that the union of M and M' is non-crossing.) Thus, having no chromatic cut is a *necessary condition* for a unique BR-matching. However, it is *not sufficient*, as shown by the example in Figure 5.1 (c).

We will give a thorough treatment of BR-matchings without chromatic cuts. We shall prove in Lemma 5.12 that BR-matchings without chromatic cuts can be classified into the following two types. A *matching of linear type* (or, for shortness, *linear matching*) is a BR-matching without a chromatic cut such that $\partial\mathrm{CH}(F)$ consists of exactly two color intervals (both necessarily of size at least 2). A *matching of circular type* (or *circular matching*) is a BR-matching without a chromatic cut such that all points of $\partial\mathrm{CH}(F)$ have the same color. The reason for these terms will be clarified below. Figure 5.1 (b–c) shows a linear and a circular matching. We shall prove that the unique BR-matchings are precisely the linear matchings. This will be a part of our main result, Theorem 5.5 below.

The segments in M are considered directed from the $\circ$-end to the $\bullet$-end. For $A \in M$, the line that contains A is denoted by $g(A)$, and it is considered directed consistently with A. For two directed segments A and B such that the lines $g(A)$ and $g(B)$ do not cross, we say that the segments (resp., the lines) are *parallel* if they have the same orientation; otherwise we call them *antiparallel.* If we delete the points of A from $g(A)$, we obtain two *outer rays*: the $\circ$*-ray* and the $\bullet$*-ray*, according to the color of the respective initial points.

Definition 5.3. *For two (directed) segments A and B, the* sidedness relation $\lhd$ *is defined as follows: $A \lhd B$ if B lies strictly to the right of $g(A)$* and *A lies strictly to the left of $g(B)$.*

The definition implies directly that the relation $\lhd$ is asymmetric: $A \lhd B$ and $B \lhd A$ cannot hold simultaneously. However, it is not necessarily transitive, as the example in Figure 5.2 (c) shows: We have $A \lhd B \lhd C$ but not $A \lhd C$. In Figure 5.2 (a) and (b), the two edges are incomparable by the $\lhd$ relation.

5.1.2 The main results

Our main results are the following three theorems.

Theorem 5.4. *Let M be a BR-matching without a chromatic cut. Then M is either of linear or circular type.*

Theorem 5.5 presents several equivalent characterizations of unique BR-matchings, or, equivalently (in view of $1 \Leftrightarrow 2$), those of linear matchings. The definition of *bichromatic quasi-parallel matchings* in condition 5 will be given later (see Definition 5.16 and Figure 5.5). They are a variation of (monochromatic) quasi-parallel matchings, introduced in [60].

Theorem 5.5 (Characterization of unique BR-matchings)**.** *Let M be a BR-matching of F. Then the following conditions are equivalent:*

1. *M is a unique matching.*
2. *M is a linear matching.*
3. *The relation $\lhd$ is a linear order on M.*
4. *No subset of segments forms one of the three patterns in Figure 5.2.*
5. *M is a bichromatic quasi-parallel matching.*

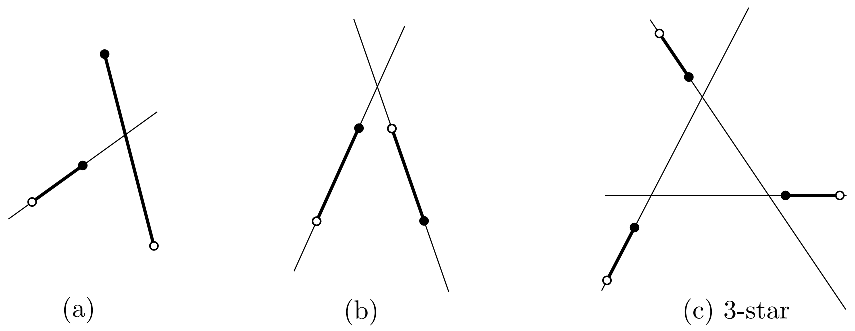


Figure 5.2: Forbidden patterns for quasi-parallel matchings. All patterns should be understood up to reversal of the colors and reflection of the plane. The pattern (b) includes the case of antiparallel segments, and the pattern (c) includes the case where three lines go through a common point. In cases (a) and (b), a chromatic cut ℓ is shown, see Lemma 5.8.

Remark 5.6. *If M satisfies any of the conditions of Theorem 5.5, then any submatching of M satisfies the conditions as well. Indeed, it obvious that conditions 3 and 4 directly imply that they hold for all subsets. For the other conditions, it follows from the equivalence stated in the theorem.*

Theorem 5.7 (Properties of circular matchings)**.** *Let M be a BR-matching of F. Then the following conditions are equivalent:*

1. *M is a circular matching.*
2. *The sidedness relation $\lhd$ is a total relation but not a linear order.*
3. *No two segments from M form one of the patterns in Figure 5.2 (a–b), but there are three segments in M that form the 3-star pattern in Figure 5.2 (c).*

Furthermore, if these conditions hold, then:

p1. The sidedness relation $\lhd$ induces naturally a circular order, explained in Section 5.4.

p2. M is not unique: There are at least two BR-matchings M' and M'' of F so that each of them is disjoint (and moreover, compatible) to M.

The following table compares linear and circular matchings with respect to the properties mentioned in Theorems 5.5 and 5.7.

	Linear Type	Circular Type
Uniqueness	M is unique	M is not unique
Patterns from Figure 5.2	(a), (b) and (c) are avoided	(a) and (b) are avoided; (c) is present
Relation $\lhd$	Linear order	Total, not linear; induces a circular order

5.1.3 Related work

Our work belongs to the study of straight-line graph drawings. One of the directions intensively studied in the recent years is that of straight-line matchings (monochromatic and bichromatic).

Given a bichromatic set F, one can consider the *bichromatic compatible matching graph* of F whose nodes correspond to the BR-matchings of F, and two nodes are connected by an edge if and only if the corresponding matchings are compatible. Aloupis, Barba, Langerman, and Souvaine [7] proved that for any F, the bichromatic compatible matching graph is connected. Aichholzer, Barba, Hackl, Pilz, and Vogtenhuber [3] proved that the diameter of this graph is at most $2n$ (which is asymptotically tight). In our work we study the situation when this graph is as small as possible—namely, when it consists of a single node.

For non-colored point sets, one can speak about the *(monochromatic) compatible matching graph.* Aichholzer et al. [4] showed that for any set of n points, the diameter of this graph is $O(\log n)$.

A related direction of research is that of geometric augmentation, see Hurtado and Tóth [41] for a survey. The general pattern of problems can be described as follows. Given a geometric graph, one wants to determine whether it is possible to add edges (segments) in order to get a bigger graph with a certain property, under what conditions this can be done, how many segments one has to add, etc. Hurtado, Kano, Rappaport, and Tóth [40] proved that any BR-matching can be augmented to a non-crossing spanning tree in $O(n \log n)$ time.

For monochromatic perfect matchings, García, Noy, and Tejel [32] showed that the number of such matchings is minimized among all n-point sets when the points are in convex position. Ishaque, Souvaine, and Tóth [43] showed that for any monochromatic perfect matching, there is a *disjoint* monochromatic compatible matching. As for the *maximal* number of BR-matchings that a set of n blue and n red points can admit, Sharir and Welzl [65] established a bound of $O(7.61^{2n})$.

5.1.4 Outline

In Section 5.2 we prove several preliminary results about chromatic cuts and the sidedness relation $\lhd$. In particular, we give a simple proof of the fact that a BR-matching which has a chromatic cut is not unique. Section 5.3 is devoted to linear matchings. We give several characterizations of them, and we give two proofs that a linear matching is unique. One proof, via the Fishnet Lemma, requires more effort to set up some additional geometric structure, but it makes the argument more transparent (Section 5.3.2). Section 5.4 analyzes circular matchings in depth, and we prove that they are never unique. Then we complete the proof of the main theorem about unique BR-matchings, Theorem 5.5.

In Section 5.5 we turn to algorithmic issues. We describe an algorithm that recognizes point sets F which admit a unique matching, an algorithm that recognizes circular matchings, and an algorithm that detects the existence of a chromatic cut by computing a so-called *balanced line.* All these algorithms run in $O(n \log n)$ time.

We conclude with some open problems and directions for future research in Section 5.6.

5.2 Preliminary results

5.2.1 Chromatic cuts

We start with a simple geometric description of BR-matchings that admit a chromatic cut.

Lemma 5.8. *Let M be a BR-matching of F. M admits a chromatic cut if and only if contains two segments A, B forming the pattern in Figure 5.2 (a) or (b), or more explicitly, if an outer ray of one segment crosses the second segment (a), or the intersection point of $g(A)$ and $g(B)$ belongs to outer rays of different colors (b), or A and B are antiparallel, which is a special case of pattern (b).*

Proof. [$\Leftarrow$] If an outer ray of the segment A crosses the second segment B, then, if we rotate $g(A)$ around an inner point of A by a small angle in one of two possible directions, depending on the orientation of A and B, then a chromatic cut is obtained, see Figure 5.2 (a). If the $\bullet$-ray of one segment and the $\circ$-ray of the second segment cross each other, then any line through inner points of A and B is a chromatic cut, see Figure 5.2 (b). The same is true if A and B are antiparallel.

[$\Rightarrow$] Let ℓ be a chromatic cut of M, and let A and B be two segments that have their $\bullet$-ends on the opposite sides of ℓ. Consider the lines $g(A)$ and $g(B)$. If $g(A)$ and $g(B)$ do not cross, they clearly must be antiparallel. If they cross, then it is not possible that the two outer rays of the same color meet, because they are on opposite sides of ℓ. $\square$

A line ℓ is a *balanced line* if in each open halfplane determined by ℓ, the number of $\bullet$-points is equal to the number of $\circ$-points. The next lemma reveals a relation between chromatic cuts and balanced lines.

Lemma 5.9. *Let M be a BR-matching. M has a chromatic cut if and only if there exists a balanced line that crosses a segment of M.*

Proof. [$\Leftarrow$] Let ℓ be a balanced line that crosses a segment A of M. We can assume that ℓ does not contain points from F: it cannot contain exactly one point of F; and if it contains two points of F of different colors, we can translate it slightly, obtaining a balanced line that still crosses A but does not contain points of F. If it contains two points of the same color, we rotate it slightly about the midpoint between these two points.

Now, A has a $\bullet$-end in one half-plane of ℓ and a $\circ$-end in the other half-plane. Since ℓ is balanced, there must be another segment B that crosses ℓ in such a way that ℓ is a chromatic cut.

[$\Rightarrow$] First, let A be a segment in M, and let p be an inner point of A that does not belong to any line determined by two points of F, other than the endpoints of A. We claim that if there is no balanced line that crosses A at p, then $g(A)$ is a balanced line.

Assume that there is no balanced line that crosses A at p. We use a continuity argument. Let $m = m_0$ be any directed line that crosses A at p. Rotate m around p counterclockwise until it makes a half-turn. Denote by m_α the line obtained from m after rotation by the angle α; so, we rotate it until we get m_π. Let φ $(0 < \varphi < \pi)$ be the angle such that m_φ coincides with $g(A)$ (as a line, ignoring the orientations). Assume without loss of generality that the right halfplane bounded by m is *dominated* by $\bullet$, in the sense that it contains more $\bullet$-points than $\circ$-points. Then the right halfplane bounded by m_π is dominated by $\circ$. As we rotate m, the points of F change sides *one by one*, except at $\alpha = \varphi$. When one point changes sides, m_α cannot change from $\bullet$-dominance to $\circ$-dominance without becoming a balanced line. Therefore, for each $0 \leq \alpha < \varphi$, the right side of m_α is dominated by $\bullet$, and for each $\varphi < \alpha \leq \pi$, the right side of m_α is dominated by $\circ$. At $\alpha = \varphi$, exactly two points of different colors change sides. The only possibility is that the $\bullet$-end of A passes from from the right side to the left side and the $\circ$-end of A passes from the left side to the right side of the rotated line. It follows that at this moment the value of $\#(\bullet) - \#(\circ)$ in the right halfplane changes from 1 to -1, and that $m_\varphi = g(A)$ is a balanced line.

Now, let ℓ be a chromatic cut that crosses $A, B \in M$ so that the $\bullet$-end of A and the $\circ$-end of B are in the same half-plane bounded by ℓ. Denote by p and q the points of intersection of ℓ with A and B, respectively. We assume without loss of generality that p and q do not belong to any line determined by points of F.

If there is a balanced line that crosses A at p, or a balanced line that crosses B at q, we are done. By the above claim, it remains to consider the case when the lines $g(A)$ and $g(B)$ are balanced. Assume without loss of generality that ℓ is horizontal, p is left of q, and the $\bullet$-end of A is above ℓ, see Figure 5.3 for an illustration.

We start with the line $k = g(A)$, directed upwards, rotate it clockwise around p until it coincides with ℓ, and then continue to rotate it clockwise around q until it coincides with $g(B)$, directed down. As above, we monitor $\#(\bullet) - \#(\circ)$ on the right side of the line k: this quantity is 0 in the initial and the final position. Just after the initial position it is -1, and just before the final position it is $+1$. In between, it makes only ± 1 jumps, since the points of F change sides of the rotated line k one by one. It follows that for some intermediate position it is 0, and thus we have a balanced line crossing one of the edges. $\square$

In Section 5.5.3, we discuss the algorithmic implementation of this proof.

Corollary 5.10. *Let M be a BR-matching of F with a chromatic cut. Then M is not unique.*

Proof. By Lemma 5.9, there is a balanced line ℓ crossing a segment $A \in M$. We construct matchings on both sides of ℓ, and denote their union by M'. Then M' is a matching of F, and we have $M' \neq M$ since M' does not use A. $\square$

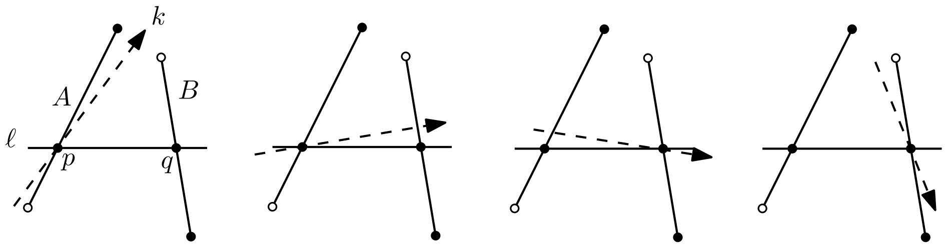


Figure 5.3: Finding a balanced line in the proof of Lemma 5.9.

Remark 5.11. *As mentioned in the introduction, Corollary 5.10 follows from the stronger statement of [7, Lemma 9]: the existence of a compatible matching $M' \neq M$. We have given a simpler alternative proof.*

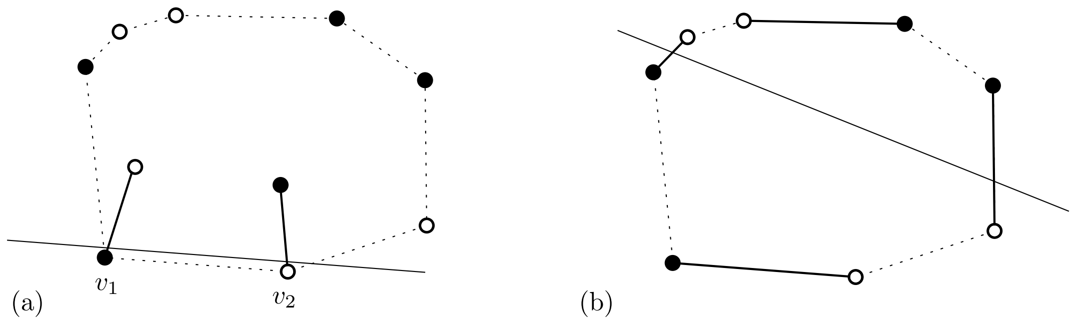


Figure 5.4: Chromatic cuts in the proof of Lemma 5.12.

Lemma 5.12. *Let M be a BR-matching of F that has no chromatic cut. Then*

- *either all points of $\partial CH(F)$ have the same color,*
- *or the points of $\partial CH(F)$ form precisely two color intervals, each of which must have size at least 2.*

In the latter case, the two boundary segments connecting points of different color necessarily belong to M.

Proof. Assume that $\partial\mathrm{CH}(F)$ has points of both colors.

If v_1 and v_2 are two neighboring points on $\partial\mathrm{CH}(F)$ with different colors, then they are matched by a segment of M. Indeed, let ℓ' be the line through v_1 and v_2. If v_1 and v_2 are not matched by a segment of M, then each of them is an endpoint of some segment of M. When we shift ℓ' slightly so that it crosses these two segments, a chromatic cut is obtained, see Figure 5.4 (a).

Therefore, if the points of $\partial\mathrm{CH}(F)$ form more than two color intervals, then at least four segments of M have both ends on $\partial\mathrm{CH}(F)$. At least two among them have the $\bullet$-end before the $\circ$-end, with respect to their circular order. Any line that crosses these two segments will be then a chromatic cut, see Figure 5.4 (b).

Thus, we have exactly two color intervals. If one of them consists of one point, then this point has two neighbors of another color of $\partial\mathrm{CH}(F)$. As observed above, this point must be matched by M to both of them, which is clearly impossible. $\square$

We recall the definition from the introduction: a *linear matching* is a BR-matching without a chromatic cut such that $\partial\text{CH}(F)$ consists of exactly two color intervals, both of size at least 2; a *circular matching* is a BR-matching without a chromatic cut such that all points of $\partial\text{CH}(F)$ have the same color. So, we have established that a BR-matching without a chromatic cut necessarily belongs to one of these two types, and we have thus completed the proof of Theorem 5.4. In the next sections we study these types in more detail.

5.2.2 The sidedness relation between segments

Now we show how the presence or absence of a chromatic cut affects some properties of the sidedness relation $\lhd$.

Lemma 5.13. *Let M be a BR-matching. M has no chromatic cut if and only if the sidedness relation $\lhd$ is a total relation, that is, for any two segments $A, B \in M$, $A \neq B$, we have $A \lhd B$ or $B \lhd A$.*

Proof. If two segments A and B have a chromatic cut, then the lines $g(A)$ and $g(B)$ must intersect as in Figure 5.2 (a) or (b), and the segments are not comparable by $\lhd$; otherwise, the lines $g(A)$ and $g(B)$ are parallel or intersect in the outer rays of the same color, and then the segments are comparable. □

Recall that the relation $\lhd$ is asymmetric by definition: we never have $A \lhd B$ and $B \lhd A$. Moreover, we will see that if M has no chromatic cut, then, in order to prove $A \lhd B$, it suffices to prove only one condition from the definition of $\lhd$:

Lemma 5.14. *Let M be a BR-matching without chromatic cut, and let $A, B \in M$ ($A \neq B$). If B lies to the right of $g(A)$, or if A lies to the left of $g(B)$, then $A \lhd B$.*

Proof. If M has no chromatic cut, then we have either $A \lhd B$ or $B \lhd A$ by Lemma 5.13. Given one of the above conditions, $B \lhd A$ is ruled out. □

5.3 Quasi-parallel, or linear, matchings

5.3.1 Characterizations of linear matchings

In this section we give several characterizations of linear matchings and prove that such matchings are unique for their point sets.

Lemma 5.15. *Let M be a linear matching. There exist $A_1, A_n \in M$, the "minimum" and the "maximum" element, such that for every $B \in M \setminus \{A_1\}$ we have $A_1 \lhd B$, and for every $C \in M \setminus \{A_n\}$ we have $C \lhd A_n$.*

Proof. By Lemma 5.12, the two boundary segments connecting points of different color belong to M. For one of them, to be denoted by A_1, all other segments of M belong to the right half-plane bounded by $g(A_1)$; for the second, to be denoted by A_n, all other segments of M belong to the left half-plane bounded by $g(A_n)$. Since M has no chromatic cut, the claim follows directly from Lemma 5.14. □

Definition 5.16. *A BR-matching M is* (bichromatic) quasi-parallel *if there exists a directed* reference line ℓ *such that the following conditions hold:*

(i) *No segment is perpendicular to ℓ.*

(ii) *For every $A \in M$, the direction of its projection on ℓ (as usual, from $\circ$ to $\bullet$) coincides with the direction of ℓ.*

(iii) *For every non-parallel $A, B \in M$, the projection of the intersection point of $g(A)$ and $g(B)$ on ℓ lies outside the convex hull of the projections of A and B on ℓ.*

Figure 5.5 shows an example of quasi-parallel matching, with horizontal ℓ.

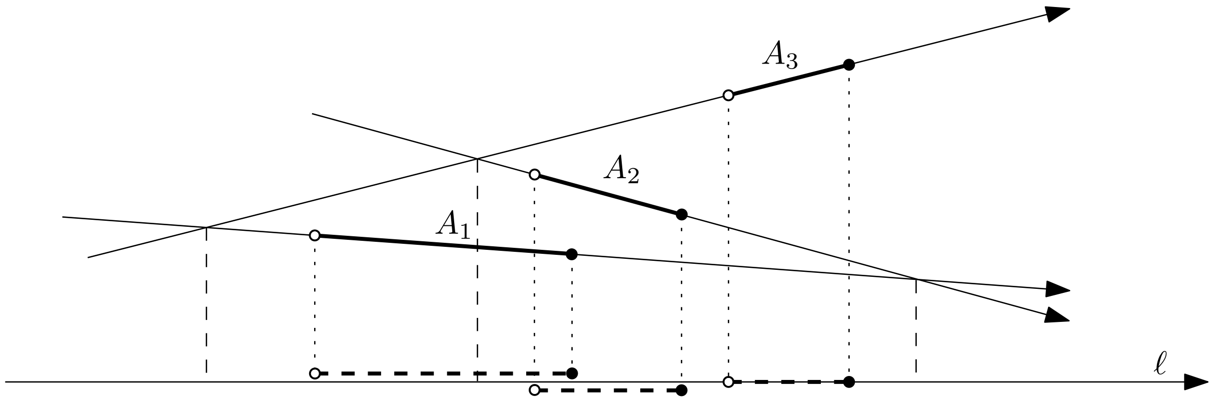


Figure 5.5: A quasi-parallel matching.

Remark 5.17. *In the monochromatic setting, the notion of quasi-parallel segments was introduced by Rote [60, 59] as a generalization of parallel segments, in the context of a dynamic programming algorithm for some instances of the traveling salesman problem. His definition can be obtained from our one by dropping condition* (ii)*.*

Lemma 5.18. *Let M be a BR-matching of a bichromatic set F. Then the following conditions are equivalent:*

1. *M is a linear matching.*
2. *The relation $\lhd$ in M is a strict linear order.*
3. *M has no patterns of the three kinds in Figure 5.2.*
4. *M is a quasi-parallel matching.*

Proof. [1 $\Rightarrow$ 2] By definition, the relation $\lhd$ is asymmetric, and according to Lemma 5.13, it is total.

It remains to establish transitivity. By Lemma 5.15, there exist $A_1, A_n \in M$ (the "minimum" and the "maximum" elements) such that for every $B \in M \setminus \{A_1\}$ we have $A_1 \lhd B$, and for every $C \in M \setminus \{A_n\}$ we have $C \lhd A_n$. We define inductively $A_2, \ldots, A_{n-1}$ as follows. Assume $A_1, \ldots, A_{i-1}$ are already defined. Let $M_i = M \setminus \{A_1, A_2, \ldots, A_{i-1}\}$. Then M_i is also a linear matching: indeed, it has no chromatic cut and has both colors on the boundary of the convex hull because A_n belongs to it. Denote the "minimum" element of M_i by A_i and repeat until all labels are assigned. (Note that we never label A_n as A_i with $i < n$.)

It follows from the construction that for all $i < j$, A_j lies to the right of $g(A_i)$. Thus, by Lemma 5.14, we have $i < j \ \Rightarrow\ A_i \triangleleft A_j$. This implies that $\triangleleft$ is a linear order.

[2 $\Rightarrow$ 3] It is easy to check that none of the configurations in Figure 5.2 is ordered linearly by $\triangleleft$.

[3 $\Rightarrow$ 4] In this proof, we follow the idea from [60]. As a preparation, one can establish by case distinction that any two or three segments which contain none of the patterns from Figure 5.2 are quasi-parallel (we omit the details).

Now, let M be a BR-matching without the forbidden patterns from Figure 5.2. For each $A \in M$, let $a(A)$ be the arc on the circle of directions corresponding to positive directions of lines m such that the angle between A and m is acute. (These are the lines that can play the role of a reference line ℓ in the definition of quasi-parallel matching, with respect to A.) Each $a(A)$ is an open half-circle, see Figure 5.6.

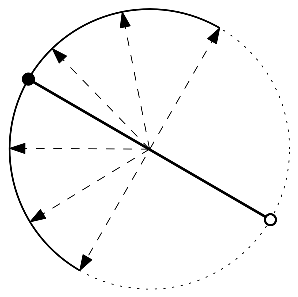


Figure 5.6: The open arc $a(A)$ for a matching segment A, used in the proof of Lemma 5.18, $3 \Rightarrow 4$.

We fix some segment $S \in M$. For any segment $A \in M$, $\{S, A\}$ is a quasi-parallel matching, and hence the intersection of the corresponding arcs $a(S) \cap a(A)$ is a non-empty sub-arc of $a(S)$, which we denote by $a'(A)$. Now, for any two segments $A, B \in M$, $\{S, A, B\}$ is a quasi-parallel matching, and hence the intersection of the corresponding arcs is non-empty. In other words, $a'(A) \cap a'(B) \neq \emptyset$. We apply Helly's Theorem to the arcs $a'(A)$ (considering them as sub-arcs of $a(S)$) and conclude that there exists a direction in the intersection of the arcs corresponding to all segments of M. A line ℓ in this direction will satisfy conditions (i) and (ii) of the definition of quasi-parallel matching. Finally, the absence of forbidden patterns implies that condition (iii) is satisfied as well.

[4 $\Rightarrow$ 1] Condition (iii) in the definition of quasi-parallel matchings implies that for any $A, B \in M$, $A \neq B$, the lines $g(A)$ and $g(B)$ are either parallel, or the outer rays of the same color cross. It follows from Lemma 5.8 that there is no chromatic cut.

A lowest and a highest (with respect to ℓ) points of F belong to the boundary of the convex hull and have different colors. Therefore, M is of linear type. $\square$

Remark 5.19. *A similar characterization of* monochromatic *quasi-parallel matchings by seven forbidden patterns was given by Rote [59]. (In the journal version [60], one of the forbidden patterns has been inadvertently omitted.) We have fewer forbidden patterns because avoiding certain monochromatic patterns becomes equivalent to the single pattern from Figure 5.2 (b) once colors are added.*

Lemma 5.18 proves the equivalence of conditions 2, 3, 4, and 5 in Theorem 5.5. Condition 3 justifies the term "matching of linear type". Now we prove that they imply the uniqueness of M.

Theorem 5.20. *Let M be a linear matching on the point set F. Then M is unique, that is, M is the only matching of F.*

Proof. By Lemma 5.18, the matching M is quasi-parallel, with reference line ℓ. We assume without loss of generality that ℓ is vertical.

Assume for contradiction that another matching M' exists. (In the figures below, the segments of M are denoted by solid lines, and the segments of M' by dashed lines.) The symmetric difference of M and M' is the union of alternating cycles. We now claim that *an alternating cycle must intersect itself.*

Consider the alternating cycle $\Pi = p_1q_1p_2q_2p_3q_3 \dots p_mq_mp_1$ that consists of segments $p_iq_i \in M$ and $q_ip_{i+1}; q_mp_1 \in M'$. We assume that p_i are $\circ$-vertices and q_i are $\bullet$-vertices. Let B be the minimum (with respect to $\lhd$) segment and let C be the maximum segment of M that belongs to Π. Then no points of Π lie left of $g(B)$ or right of $g(C)$. Since for both B and C the $\bullet$-end is higher than the $\circ$-end, the path Π must cross itself at least once, establishing the claim, see Figure 5.7.

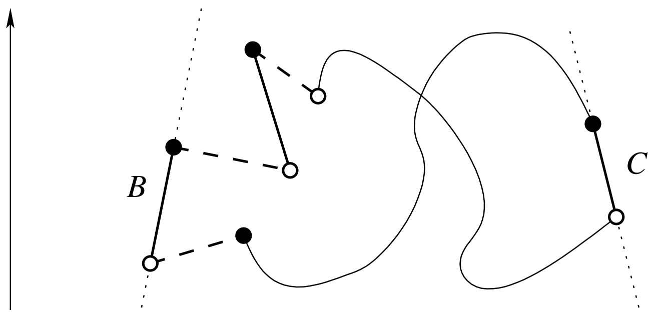


Figure 5.7: Illustration for the proof of Theorem 5.20: an alternating path for M crosses itself.

We now traverse the path Π, starting from $p_1q_1p_2q_2\dots$, until it crosses itself for the first time, say, at a point r. There can be no crossing r between two segments of M or two segments of M'. Hence, the first occurrence of r on Π is on a segment p_iq_i of M, and the second is on a segment q_jp_{j+1} of M', or vice versa. We consider only the first case, the other being similar. In this case, we consider the matching N that consists of segments $rq_i, p_{i+1}q_{i+1}, p_{i+2}q_{i+2}, \dots p_jq_j$ (that is, N consists of the segments of M that occur on Π between the two times that it visits r, and the part of segment of M that contains r). It is clear that N is also quasi-parallel, with respect to the same reference line ℓ.

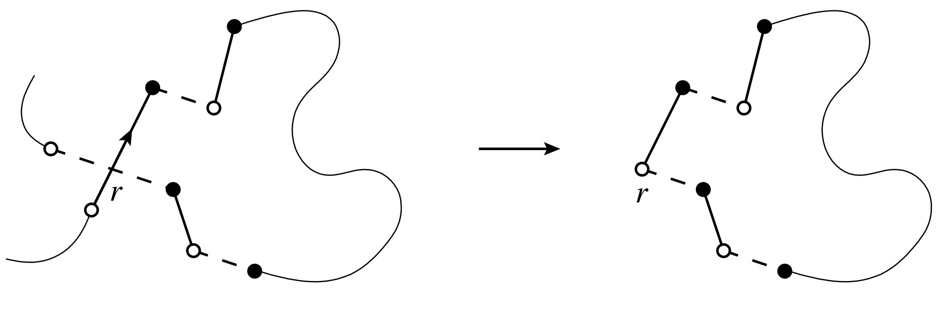


Figure 5.8: Illustration to the proof of Theorem 5.20: an alternating path for N.

The closed path $rq_ip_{i+1}q_{i+1}p_{i+2}q_{i+2}\dots p_jq_jr$ is an alternating path for N. By the choice of r, this path does not intersect itself, see Figure 5.8, which contradicts the claim proved above that an alternating path of a quasi-parallel matching always intersects itself. □

The proof of Theorem 5.20 tells us that a closed alternating path cannot exist. In contrast, it is always possible to construct at least two *open* alternating paths from the minimum to the maximum element of M:

Observation 5.21. *Let M be a linear matching. Then there exist two alternating paths containing all segments of M in the order $\lhd$.*

Proof. Let $A_1, \dots, A_n$ be the segments of M, ordered by $\lhd$. We proceed by induction, see Figure 5.9. Let R_k be a path from A_1 to A_k in which the segments of M appear according to $\lhd$. We obtain R_{k+1} by taking R_k and adding a color-conforming segment from A_k to A_{k+1}. This is possible because there is no other segment of M between A_k and A_{k+1}. The color of the starting point can be chosen and thus we have two such paths. $\square$

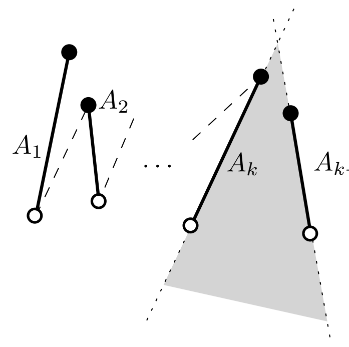


Figure 5.9: Extending an alternating path in the proof of Observation 5.21.

5.3.2 Proof of Theorem 5.20 by the Fishnet Lemma

Let us return to the proof of Theorem 5.20 for the following restricted case. Assume that all segments of the given matching M are vertical, and all segments of a supposed matching M' are horizontal. Then the alternating path as described in the proof will be y-(weakly)-monotone. This is essentially the reason why in this case we never get a *closed* alternating line.

This argument can be extended for the general case if we replace the horizontal and vertical lines (that contain the segments in the special case) by an appropriately constructed *topological grid*. Thus we obtain another proof of Theorem 5.20, which shows in a more clear and intuitive light why one cannot close the path. This approach will be made precise with the following *Fishnet Lemma*. We will apply it only to polygonal curves, but we formulate it for arbitrary curves, see Figure 5.10.

Consider a set $V = \{v_1, \dots, v_n\}$ of pairwise non-crossing unbounded Jordan curves ("ropes") such that the plane is partitioned into $n+1$ connected regions: R_0 bounded only by v_1; R_i, $1 \le i \le n-1$ bounded only by v_i and v_{i+1}; and R_n bounded only by v_n. These curves will be called the *vertical* curves. In the illustrations they will be black.

Consider another set $G = \{g_1, \dots, g_m\}$ of pairwise non-crossing Jordan arcs, called the *horizontal* arcs and drawn in green, such that every curve g_k has its endpoints on two different vertical curves v_i and v_j ($j > i$), has exactly one intersection point with each vertical curve $v_i, v_{i+1}, v_{i+2}, \dots, v_j$, and no intersection with the other curves. See Figure 5.10 (a) for an example. We say that the curves $V \cup G$ form a *partial (topological) grid.*

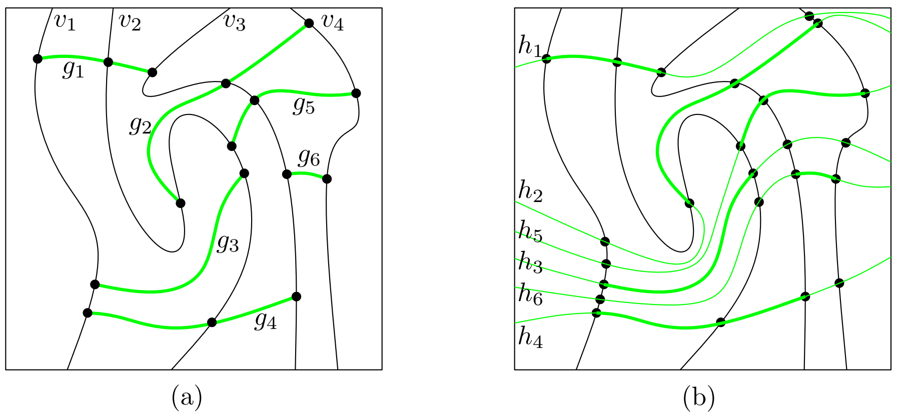


Figure 5.10: (a) A partial grid. (b) Extension to a complete grid of ropes.

Lemma 5.22 (The Fishnet Lemma)**.** *The horizontal arcs g_k of a partial topological grid $V \cup G$ can be extended to pairwise non-crossing unbounded Jordan arcs h_k in such a way that the curves $H = \{h_1, \dots, h_m\}$ together with V form a complete topological grid $V \cup H$: each horizontal curve h_k crosses each vertical curve v_i exactly once. See Figure 5.10 (b).*

Proof. We prove the lemma by a construction which incrementally extends the horizontal segments until a complete topological grid is obtained.

The bounded faces of the given curve arrangement $V \cup G$ are topological quadrilaterals: they are bounded by two consecutive vertical curves and two horizontal curves. The bounded faces of the desired final curve arrangement $V \cup H$ are also such quadrilaterals, with the additional property that they have no extra vertices on their boundary besides the four corner intersections. In $V \cup G$, such extra vertices arise as the endpoints of the segments g_k.

Let us take such a bounded face, between two vertical curves v_i and v_{i+1}, with an endpoint of g_k on one of its vertical sides, see Figure 5.11 (a)–(b). We can extend g_k to some point on the opposite vertical side, chosen to be distinct from all other endpoints, splitting the face into two and creating a new intersection point. (The existence of such an extension follows from the Jordan–Schoenflies Theorem, by which the bounded face is homeomorphic to a disc.) An unbounded face between two successive vertical curves

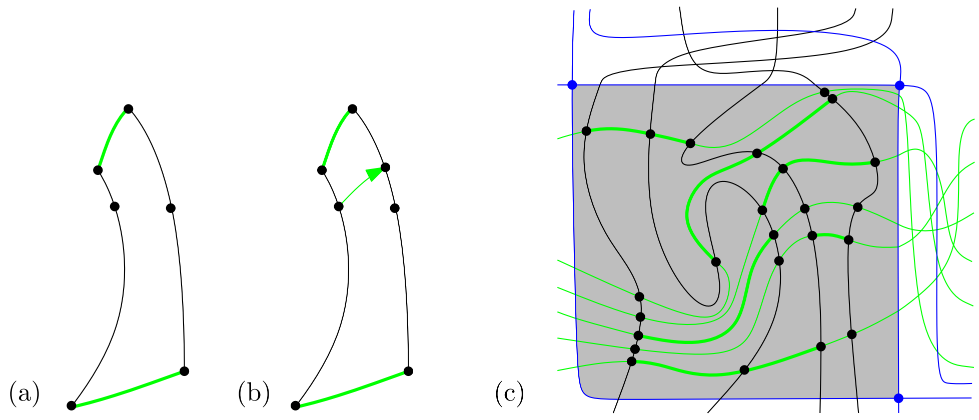


Figure 5.11: (a) A quadrilateral face with extra vertices; the shaded face from Figure 5.10 (a). (b) Adding an edge. (c) Embedding the grid into a pseudoline arrangement.

v_i and v_{i+1} that has an extra vertex on a vertical side can be treated similarly.

We continue the above extension procedure as long as possible. Since we are adding new intersection points, but no two curves can intersect twice, this must terminate. Now we are almost done: each horizontal curve extends from v_1 to v_n and crosses each vertical curve exactly once. We just extend the horizontal curves to infinity, into R_0 and R_n, without crossings. $\square$

This lemma can be interpreted in the context of pseudoline arrangements. In an arrangement of pseudolines, each pseudoline is an unbounded Jordan curve, and every pair of pseudolines has to cross *exactly* once. The grid construction can be embedded in a true pseudoline arrangement, see Figure 5.11 (c): simply enclose all crossings in a bounded region formed by three new (blue) pseudolines and let the crossings between vertical lines and between horizontal lines occur outside this region.

We return to the proof of Theorem 5.20.

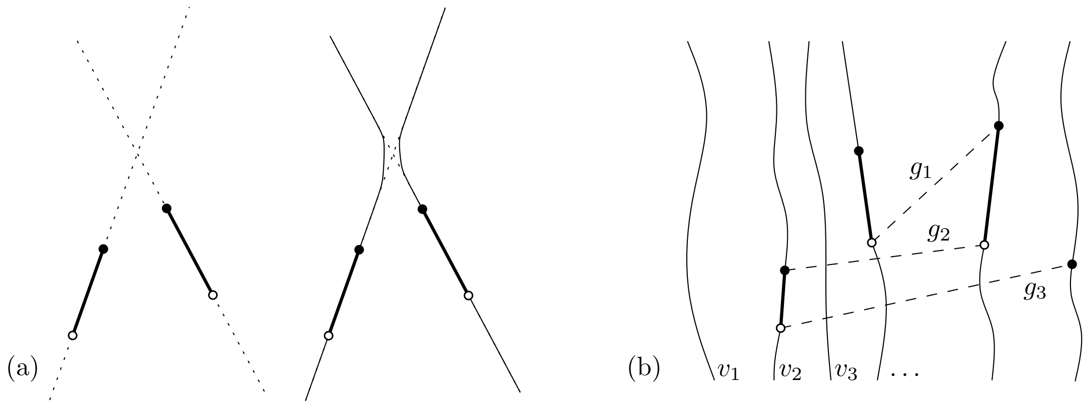


Figure 5.12: Applying the Fishnet Lemma.

Proof. Given a quasi-parallel matching M, we construct a set of Jordan curves V as in Lemma 5.22 by considering the line arrangement formed by the segments $s_1 \triangleleft \cdots \triangleleft s_n$ with the corresponding lines $g(s_1), \ldots, g(s_n)$. We construct curve v_i by starting from s_i and going along $g(s_i)$. At each intersection, the curves switch from one line to the other, and after a slight deformation in the vicinity of the intersections, they become non-crossing, see Figure 5.12 (a). These crossings lie outside the parts of the lines where the segments lie; therefore the switchings have no influence on the left-to-right order of the segments s_i. The setup of these n non-crossing "vertical" curves gives us the possibility to establish a common orientation and speak about "above" and "below" on each curve in a consistent way.

Now assume there is another matching M'. M and M' form at least one closed alternating path. Let $G = \{g_1, \ldots, g_m\}$ be the segments of M' on such a cycle in the order in which they are traversed. V and G satisfy the condition of the Fishnet Lemma and thus can be extended to a complete topological grid. Assume without loss of generality that g_1 lies above g_2 on the common incident edge of M, see Figure 5.12 (b). Since the relative order of $\circ$-vertices and $\bullet$-vertices is the same on all vertical curves, this property carries over to successive edges: g_i lies above g_{i+1} on the vertical curve containing their common segment of M, for $i = 1, \ldots, k-1$. Since the extended horizontal curves $h_1, \ldots, h_k$ intersect each vertical curve in the same order, we conclude that every vertical curve

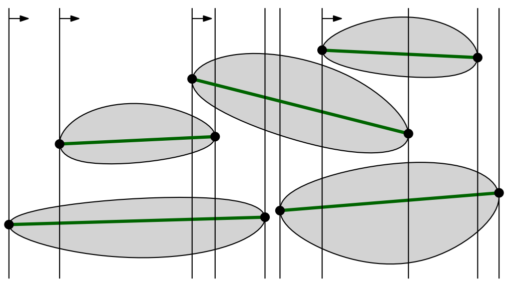


Figure 5.13: Separability by vertical translation

intersects h_i above h_{i+1}. But then g_m cannot reach the starting point above g_1 on the common incident edge of M, a contradiction. □

We mention separability by translation [26] as another easy consequence of the Fishnet Lemma: in any family of n disjoint convex (or even just x-monotone) sets in the plane, one can find one set that can be translated vertically upward to infinity without colliding with the others (Figure 5.13): Just draw a "horizontal" segment g_i between the leftmost and the rightmost point of each set, and vertical lines through all segment endpoints. The Fishnet Lemma will identify a horizontal curve h_i lying above all other curves, and the corresponding set can be translated to infinity. (There is, however, an easy direct proof of vertical separability, see Guibas and Yao [34, 35]: Among the sets whose left endpoint is visible from above, as marked by arrows in Figure 5.13, the one with the rightmost left endpoint can be translated to infinity.)

5.4 Circular matchings

5.4.1 The structure of circular matchings

In this section we study circular matchings in more detail. Recall that such a matching is a BR-matching without a chromatic cut for which all points on the convex hull have the same color. We assume without loss of generality that this color is $\bullet$.

We prove that if M is of circular type, then its point set has at least two other matchings. Moreover, we show that for a circular matching, the relation $\lhd$ induces a *circular order* (this will justify the term "matching of circular type"), and describe such matchings in terms of forbidden patterns.

Lemma 5.23. *A BR-matching M is of circular type if and only if it has no patterns (a) and (b) from Figure 5.2, and has at least one pattern (c) (a 3-star).*

Proof. We saw in Lemma 5.8 that a BR-matching has no chromatic cut if and only if it avoids the patterns (a) and (b). By Lemma 5.12, a BR-matching without chromatic cut is either of linear or of circular type. By Lemma 5.18, a BR-matching is of linear type if and only if it avoids (a), (b) and (c). Therefore, a BR-matching is of circular type if and only if it avoids (a) and (b), but contains (c). □

Theorem 5.24. *Let M be a matching of circular type on the point set F. Then there are at least two disjoint BR-matchings on F, compatible to M.*

Proof. By Lemma 5.23, there are three segments that form a 3-star. They split the plane into three convex regions Q_1, Q_2 and Q_3 and a triangle as in Figure 5.14 (a). The triangle is bounded (without loss of generality) by three $\circ$-rays, and no points of F lie in the interior of this triangle. Otherwise, if a whole segment A of M lies inside the triangle, then the $\bullet$-ray determined by A crosses a $\circ$-ray, and we have pattern (b). And if only one endpoint of A lies inside the triangle, then A itself crosses a $\circ$-ray, and we have pattern (a). Both cases contradict Lemma 5.23.

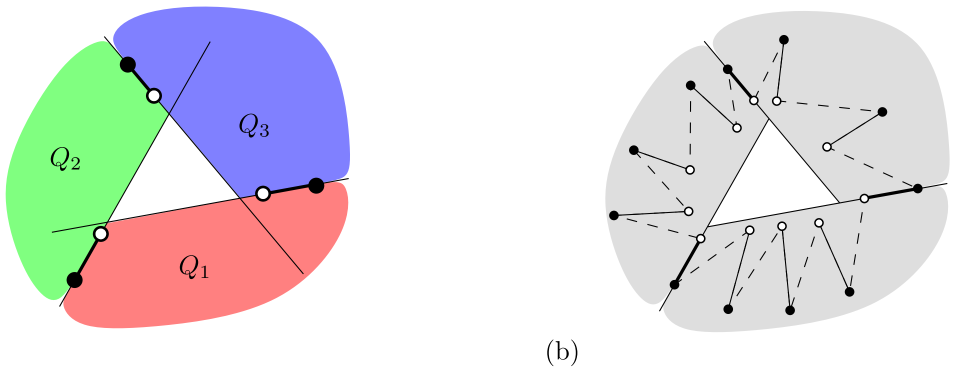


Figure 5.14: The three regions Q_1, Q_2, Q_3 and an alternating cycle in the proof of Theorem 5.24.

All segments in a region Q_i together with the two defining segments are of linear type (indeed, they have no chromatic cut but have both colors on the boundary of the convex hull). Thus, by Observation 5.21, in each region there is an alternating path from the $\circ$-point of the left bounding segment to the $\bullet$-point of the right bounding segment (or vice versa), see Figure 5.14 (b). The union of the three paths forms an alternating polygon and thus we have found a different compatible BR-matching M'. If we choose the paths in the other direction (from the $\bullet$-point of the left bounding segment to the $\circ$-point of the right bounding segment), we get another BR-matching M''. □

Remark 5.25. *If M is a matching of circular type on the point set F, then F can in fact have exponentially many BR-matchings, as the following construction shows. Let A_1, A_2, A_3 be three segments so that $g(A_1), g(A_2), g(A_3)$ intersect at one point O that belongs to their $\circ$-rays, and so that $\{A_1, A_2, A_3\}$ is a circular matching. Repeat this construction inside the triangle bounded by the $\circ$-ends of A_1, A_2, A_3, using the same point O and only taking care of general position, see Figure 5.15. This can be repeated an arbitrary number of times. Then in each "layer" we have three BR-matchings; hence, at least $3^{n/3}$ BR-matchings for the whole point set (n denotes the number of segments).*

Now we study in more detail the relation $\triangleleft$ for circular matchings. In the proof of Theorem 5.24 we saw that a circular matching is a union of three linear matchings, see Figure 5.14 (b). In the next lemma we prove that in fact it is a union of *two* linear matchings.

Lemma 5.26. *Let M be a circular matching, and let B be a segment of M. The match-*

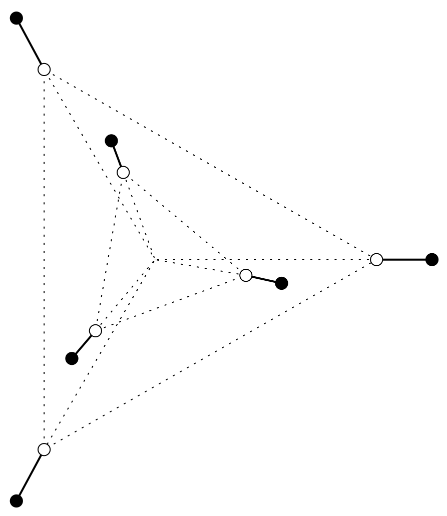


Figure 5.15: A circular matching whose point set has exponentially many BR-matchings.

ings

$$\begin{aligned}
M_B^R &= \{X \in M : B \lhd X\},\\
M_B^{R+} &= \{X \in M : B \lhd X\} \cup \{B\},\\
M_B^L &= \{X \in M : X \lhd B\},\\
M_B^{L+} &= \{X \in M : X \lhd B\} \cup \{B\}.
\end{aligned}$$

are not empty, and they are of linear type.

Proof. Consider first the matching M_B^{R+}. Since it contains B, it is non-empty. Since it is a submatching of M, it has no chromatic cut. Both the $\circ$- and the $\bullet$-end of B belong to the boundary of its convex hull; therefore M_B^{R+} must be of linear type. Similarly, M_B^{L+} is of linear type.

If M_B^R is empty, then $M_B^{L+} = M$, which is impossible since M is of circular type, and M_B^{L+} of linear type. Now, since M_B^{R+} is of linear type, and M_B^R is a subset of this matching, M_B^R is of linear type as well (this follows from 1 $\Leftrightarrow$ 3 in Lemma 5.18). The proof for M_B^L is similar. $\square$

Corollary 5.27. *The relation $\lhd$ in a matching M of circular type has neither "minimal" nor "maximal" element:*

$$\begin{aligned}
&\textit{for every } B \in M \textit{ there exists an } A \in M \textit{ such that } A \lhd B;\\
&\textit{for every } B \in M \textit{ there exists an } A \in M \textit{ such that } B \lhd A.
\end{aligned}$$

Proof. Otherwise, for such an element B, M_B^L or M_B^R would be empty. $\square$

Lemma 5.28. *Let M be a circular matching. Let B be any segment of M. Let A be the minimum (with respect to $\lhd$) element of M_B^L, and let Z be the maximum element of M_B^R. Then the triple $\{A, B, Z\}$ is a circular matching (a 3-star).*

Proof. If M is of size 3, that is, $M = \{A, B, Z\}$, there is nothing to prove. Thus, we assume that there is at least one more segment in M. Assume without loss of generality that M_B^R contains at least one segment in addition to Z.

Let C be a segment of M such that $C \lhd A$. (Such a segment exists by Lemma 5.26.) Since A is the minimum element of M_B^L, we have $C \in M_B^R$, that is, $B \lhd C$.

If $C = Z$ then $Z \lhd A \lhd B \lhd Z$, that is, the relation $\lhd$ in the triple $\{A, B, Z\}$ is not linear; therefore $\{A, B, Z\}$ is of circular type.

Suppose now that $C \neq Z$, and consider the matching $\{A, B, C, Z\}$. We have $C \lhd A \lhd B \lhd C$. Thus, the relation $\lhd$ in the matching $\{A, B, C, Z\}$ is not linear; therefore, $\{A, B, C, Z\}$ is of circular type. Now, by Lemma 5.26, some segment in $\{A, B, C, Z\}$ must lie to the right of Z according to the relation $\lhd$. Since $B \lhd Z$ and $C \lhd Z$, we have $Z \lhd A$. Thus, $Z \lhd A \lhd B \lhd Z$, and this means that $\{A, B, Z\}$ is of circular type. □

We shall show that if M is a circular matching, then there exists a natural *circular order* of its edges. A circular (or cyclic) order is a ternary relation which models the "clockwise" relation among elements arranged on a cycle. A standard way of constructing a circular order from j linear orders $A_{11} \leq A_{12} \leq \cdots \leq A_{1i_1}$, $A_{21} \leq A_{22} \leq \cdots \leq A_{2i_2}$, $\ldots$, $A_{j1} \leq A_{j2} \leq \cdots \leq A_{ji_j}$ is their "gluing": we say that $[X, Y, Z]$ (and, equivalently, $[Y, Z, X]$ and $[Z, X, Y]$) if X, Y and Z appear in the order XYZ or YZX or ZXY in the sequence

$$A_{11}, A_{12}, \ldots, A_{1i_1},\ A_{21}, A_{22}, \ldots, A_{2i_2},\ \ldots,\ A_{j1}, A_{j2}, \ldots, A_{ji_j}$$

We fix $B \in M$ and apply this procedure on M_B^{L+} and M_B^R in which $\lhd$ is linear by Lemma 5.26. Let $A_1, A_2, \ldots, A_k$ be the segments of M_B^L labeled so that $A_1 \lhd A_2 \lhd \cdots \lhd A_k$, and let $C_1, C_2, \ldots, C_m$ be the segments of M_B^R labeled so that $C_1 \lhd C_2 \lhd \cdots \lhd C_m$. By Lemma 5.28 we have $C_m \lhd A_1$. Thus, we consider the circular order $[*, *, *]$ induced by

$$B \lhd C_1 \lhd C_2 \lhd \cdots \lhd C_m \lhd A_1 \lhd A_2 \lhd \cdots \lhd A_k \lhd B. \tag{5.1}$$

That is, for $X, Y, Z \in M$ we have $[X, Y, Z]$ (and, equivalently $[Y, Z, X]$ and $[Z, X, Y]$) if and only if we have in (5.1) $X \lhd \cdots \lhd Y \lhd \cdots \lhd Z$, or $Y \lhd \cdots \lhd Z \lhd \cdots \lhd X$, or $Z \lhd \cdots \lhd X \lhd \cdots \lhd Y$. We always have either $[X, Y, Z]$ or $[X, Z, Y]$, but never both.

The circular order $[*, *, *]$ will be referred to as the *canonical circular order on M*. The next results describe the geometric intuition beyond this definition: we shall see that $[X, Y, Z]$ means in fact that these segments appear in this order clockwise. Moreover, we shall see that the definition of $[*, *, *]$ does not depend on the choice of B.

Lemma 5.29. *Let M be a circular matching, and let $X, Y, Z \in M$. Then we have $[X, Y, Z]$ if and only if at least two among the following three conditions hold: $X \lhd Y$; $Y \lhd Z$; $Z \lhd X$.*

If all three conditions hold, then $\{X, Y, Z\}$ is a 3-star; and if exactly two among the statement hold, then $\{X, Y, Z\}$ is a linear matching. All possible situations for $[X, Y, Z]$ (with respect to $\lhd$) appear in Figure 5.16.

Proof. The segment B from the definition of $[*, *, *]$ is the maximum element of M_B^{L+}. Therefore, it is convenient to denote $A_{k+1} = B$. Now we have four cases.

- Case 1: $X, Y, Z \in M_B^{L+}$.

  In this case $\{X, Y, Z\}$ is of linear type (by Lemma 5.26). Therefore either one or two of the conditions hold. If exactly two conditions hold: assume without loss of generality that $X \lhd Y \lhd Z$. Since $A_1 \lhd \cdots \lhd A_{k+1}$ is a linear order in M_B^{L+}, we have

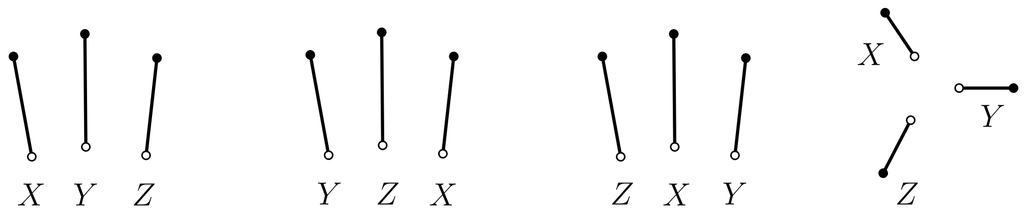


Figure 5.16: Possible configurations of three segments that satisfy $[X, Y, Z]$.

$X = A_\alpha, Y = A_\beta, Z = A_\gamma$ for some $1 \le \alpha < \beta < \gamma \le k+1$. Now we have $[X, Y, Z]$ by definition. If exactly one condition holds: assume that it is $X \lhd Y$; then we have $X \lhd Z \lhd Y$, which implies "not $[X, Y, Z]$".

- Case 2: two edges of $\{X, Y, Z\}$ belong to M_B^{L+}, and one to M_B^R. Assume without loss of generality that $X, Y \in M_B^{L+}, Z \in M_B^R$ and that $X \lhd Y$.

  Then we have $X = A_\alpha, Y = A_\beta$ for some $\alpha < \beta$ and $Z = C_\gamma$ for some γ, and, therefore, $[X, Y, Z]$.

  At the same time in this case at least two of the conditions hold: indeed, assume $X \lhd Z \lhd Y$. Then B is distinct from X, Y, Z (in particular, $B \neq Y$ because $B \lhd Z$). Now, in the matching $\{X, Y, Z, B\}$ there is a minimum element, X, but there is no maximum element. Therefore, $\{X, Y, Z, B\}$ is neither of linear nor of circular type—a contradiction.

- Case 3: one edge of $\{X, Y, Z\}$ belongs to M_B^{L+}, and two to M_B^R, and Case 4: all edges of $\{X, Y, Z\}$ belong to M_B^R, are similar to cases 2 and 1. Therefore, we omit their proofs. □

Corollary 5.30. *The canonical circular order does not depend on the choice of B.*

Proof. By Lemma 5.29, we have an equivalent definition of the circular order that only depends on relations between triples of segments. □

Lemma 5.31. *Let M be a circular matching, and let $X \in M$. Then the immediate successor of X in the canonical circular order is the minimum element of M_X^R.*

Proof. This is immediate for B (as in definition of $[*, *, *]$), and, since we saw in Corollary 5.30 that the circular order $[*, *, *]$ does not depend on the choice of B, this is true for all segments. □

Lemmas 5.29 and 5.31 show that the canonical circular order describes the combinatorial structure of circular matchings in a natural way, similarly to the way in which $\lhd$ describes the structure of linear matchings.

5.4.2 Proofs of Theorems 5.5 and 5.7

At this point we are ready to complete the proofs of Theorems 5.5 and 5.7.

Proof of Theorem 5.5. Equivalence of conditions 2, 3, 4, 5 is proven in Lemma 5.18. Finally, 2 ⇒ 1 (if a BR-matching M is of linear type, then it is unique) is proven in Theorem 5.20; and 1 ⇐ 2 (if a BR-matching M is unique, then it is of linear type) follows from Corollary 5.10 (if M is unique, then it has no chromatic cut), Lemma 5.12

(if M has no chromatic cut, then it is either of linear or circular type), and Theorem 5.24 (if M is of circular type, then it is not unique). □

Proof of Theorem 5.7. First we observe that the following statements are equivalent:

1′. M contains no chromatic cut.

2′. The sidedness relation $\lhd$ is a total relation.

3′. No two segments from M form one of the patterns in Figure 5.2 (a–b).

The equivalence $1' \Leftrightarrow 2'$ is Lemma 5.13, and the equivalence $1' \Leftrightarrow 3'$ is Lemma 5.8. Each of the three conditions $1', 2', 3'$ is equivalent to the corresponding condition $1, 2, 3$ in the theorem with the additional constraint that M is not of linear type, by Theorem 5.5 (conditions $2, 3, 4$, respectively). This establishes that the three first conditions of the theorem are equivalent.

Property p1 is explained in Lemmas 5.29 and 5.31, and p2 is proved by Theorem 5.24. □

5.5 Algorithms

In this section we describe several algorithms. The first checks whether a given point set F has a unique BR-matching. This algorithm is based on yet another characterization of unique BR-matchings. The second checks if a given BR-matching is circular. Applying these algorithms together, we can check if a given matching has a chromatic cut. The third algorithm finds a balanced line through one of the segments involved in a chromatic cut (Lemma 5.9).

5.5.1 Testing a matching for uniqueness

Definition 5.32. *A BR-matching M has the* drum property *with respect to the segments $A, B \in M$ $(A \neq B)$ if A and B are the only segments from M on $\partial CH(F)$.*

Theorem 5.33. *Let $M = \{A_1, A_2, \ldots, A_n\}$ be a BR-matching such that $A_1 \lhd A_2 \lhd \cdots \lhd A_n$. Then the following conditions are equivalent:*

1. *M is the unique BR-matching.*

2. *For every $i < j$, every subset $S \subseteq \{A_i, A_{i+1}, \ldots, A_j\}$ with $A_i, A_j \in S$ has the drum property for A_i and A_j.*

3. *For every $j > 1$, the set $\{A_1, A_2, \ldots, A_j\}$ has the drum property for A_1 and A_j; and for every $i < n$, the set $\{A_i, A_{i+1}, \ldots, A_n\}$ has the drum property for A_i and A_n.*

Recall that the relation $\lhd$ is not necessarily transitive. Thus, the assumption of the theorem does not imply $A_i \lhd A_j$ for $i < j$.

Proof. [1 ⇒ 2] By Condition 3 of Theorem 5.5, $\lhd$ is a linear order on M, and by Condition 2, the convex hull has only two color intervals. Uniqueness of the matching, as well as all the linear order $\lhd$ and the property of having two color intervals, carry over to all subsets S of M, as mentioned in the remark after the theorem. A_i and A_j are the minimal and maximal elements of S. In particular, $A_i \lhd B$ for every $B \in S \setminus \{A_i\}$, and thus, A_i lies on the convex hull. Similarly, A_j lies on the convex hull. Since there are only two color intervals on the convex hull of S, there can be no other matching edges on the convex hull. Thus, S has the drum property for A_i and A_j.

[2 ⇒ 3] is clear since 3 is a special case of 2.

[3 ⇒ 1]: Since $\{A_1, A_2, \dots, A_j\}$ has the drum property for A_1 and A_j, all segments $A_1, \dots, A_{j-1}$ lie on the same side of $g(A_j)$. Since $A_{j-1} \lhd A_j$ by assumption, we know that the segment A_{j-1} lies left of A_j, and hence we conclude that all segments A_i lie left of $g(A_j)$, for $i < j$. Similarly, from the drum property for $\{A_i, A_{i+1}, \dots, A_n\}$ we conclude that the segments A_j lie right of $g(A_i)$, for $j > i$. These two conditions together mean that $A_i \lhd A_j$ for $i < j$. Therefore, Condition 3 of Theorem 5.5 holds, and M is unique. □

From Property 3 of Theorem 5.33 we can derive a linear-time algorithm for testing whether M is unique, once an ordering with $A_1 \lhd A_2 \lhd \dots \lhd A_n$ has been computed: We incrementally compute $P_j := \mathrm{CH}(\{A_1, A_2, \dots, A_j\})$ for $j = 2, \dots, n$ and check the drum property as we go.

The algorithm that we shall describe proceeds similarly as the folklore linear-time algorithm for computing the convex hull of points that are given in sorted order by x-coordinate, but it is valid for a different reason. We start from the general paradigm for computing convex hulls incrementally (see, for example, [44]), which is the basis for more elaborate randomized incremental algorithms that also work in higher dimensions, see [25, Chapter 11]. The current convex hull H is extended by a new point p as follows:

C1. Check whether $p \in H$. If this is the case, stop.

C2. If not, find a boundary point $q \in \partial H$ that is visible from p.

C3. Walk from q in both directions to find the tangents pq_1 and pq_2 from p to H.

C4. Update the convex hull: remove the part between q_1 and q_2 that has been walked over, and replace it with q_1pq_2.

If H is maintained as a linked list, Steps C3 and C4 take only linear time overall, because everything that is walked over is deleted. The "expensive" steps that are responsible for the superlinear running time of convex hull algorithms are C1 and C2. However, just as for the linear-time hull computation of sorted points, we will see that these steps are trivial in our case. (We extend the convex hull by inserting not a single point but two points of A_{j+1} at a time.)

We want to check the drum property for $\{A_1, A_2, \dots, A_{j+1}\}$. If it holds, then we *know* that the new points of A_{j+1} do not lie in P_j; and since A_j lies on the boundary of P_j but not of P_{j+1}, we know that A_j is visible from at least one point of A_{j+1}. We can start the search in Step C3 from there. This visibility assumption can be checked in constant time, and it guarantees that A_j disappears from the boundary of P_{j+1}. The other part of the drum property, that A_1 and A_{j+1} lie on the boundary of P_{j+1}, is trivial to check after Step C4 is completed. The overall running time is linear.

In a second symmetric step, we start from the end and compute $\mathrm{CH}(\{A_i, A_{i+1}, \dots, A_n\})$ for $i = n-1, \dots, 1$.

Theorem 5.34. *It can be checked in $O(n \log n)$ time whether a bichromatic set has a unique non-crossing BR-matching.*

Proof. First we have to compute some BR-matching $M = \{A_1, A_2, \dots, A_n\}$. It is well-known that this can be done by recursive ham-sandwich cuts in $O(n \log n)$ time. A ham-sandwich cut is a line ℓ that partitions a bichromatic set such that each open half-plane contains at most $\lfloor \frac{n}{2} \rfloor$ points of each color. If n is odd, ℓ must go through a red and a blue point. We can match these points to each other, and recursively find a BR-matching in the $\left(\frac{n-1}{2} + \frac{n-1}{2}\right)$-sets in each half-plane. If n is even, ℓ may go through one or two points, but by shifting ℓ slightly we can push these points to the correct side such that each half-plane contains an $\left(\frac{n}{2} + \frac{n}{2}\right)$-set. We recurse as above. A ham-sandwich cut can be found in linear time [49]. Hence this procedure leads to a running time of $T(n) = O(n) + 2 \cdot T(n/2)$, which gives $T(n) = O(n \log n)$.

Next, we compute an ordering such that

$$A_1 \lhd A_2 \lhd \cdots \lhd A_n. \tag{5.2}$$

We do this by a standard sorting algorithm in $O(n \log n)$ time, as if the relation $\lhd$ were a linear order. If, at any time during the sort, we find two segments that are not comparable by $\lhd$, we quit. Finally, we check condition (5.2) in $O(n)$ time. (This final check is not necessary, if, for example, mergesort is used as the sorting algorithm.) This step is guaranteed to find an ordering (5.2) if the matching is unique. If $\lhd$ is not a linear order, it may succeed or fail.

As the last step, we check Property 3 of Theorem 5.33 in linear time, as outlined above. □

5.5.2 Testing for a circular matching

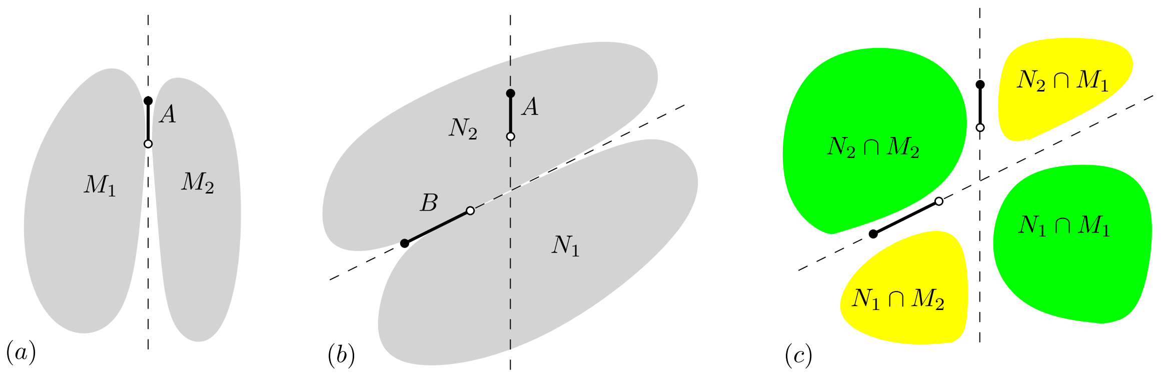


Figure 5.17: Separation of M into 6 overlapping BR-Matchings

It is also possible to determine in $O(n \log n)$ if a BR-matching M is circular, by an easy divide-and-conquer algorithm. Let A and B be two arbitrary segments in M. Let $M_1 = M_A^{L+}$ and $M_2 = M_A^{R+}$ (that is, the segments that lie to the left or to the right of A, including A itself) and likewise $N_1 = M_B^{L+}$ and $N_2 = M_B^{R+}$ (see Figure 5.17; recall that the segments are implicitly directed from white to black). M_i and N_i are linear matchings by Lemma 5.26. Finally, define $Q_1 := (M_2 \cap N_2) \cup (M_1 \cap N_1)$ and $Q_2 := (M_1 \cap N_2) \cup (M_2 \cap N_1)$.

Observation 5.35. *A BR-matching M has a chromatic cut if and only if at least one of the six matchings defined above has a chromatic cut.*

Proof. Consider two segments in M. Then they must be both in one of the matchings M' as defined above. If they have a chromatic cut, then M' has a chromatic cut. The other direction is obvious. □

Theorem 5.36. *It can be checked in $O(n \log n)$ time whether a BR-matching M is of circular type.*

Proof. The algorithm starts to compute the convex hull of M. If all points on $\partial CH(M)$ are of the same color we know it is not a linear matching and it remains to check if M has no forbidden pattern as in Figure 5.2 (a–b).

We pick any segment A_0 and split M along $g(A_0)$. We compute the linear order of both parts. This gives a potential circular order. We remember this order for the remaining part.

The rest of the algorithm works recursively. We start by defining M_1 and M_2 as above for any segment A. Let B be the median of the larger of the M_i with respect to $\triangleleft$. The BR-matchings N_i and Q_i are also defined as above. For the BR-matchings M_i and N_i, it can be checked in linear time if they are of linear type, because we have already precomputed the order. As B is the median of the larger of the M_i, $n/4 \leq |Q_i| \leq 3n/4$ for every i. We check recursively if Q_1 and Q_2 has no chromatic cut. For the running time $T(n)$, we have $T(n) \leq O(n) + \max_{1/4 \leq \alpha \leq 3/4}[T(\alpha n) + T((1-\alpha)n)]$. Thus $T(n) = O(n \log n)$.

If any of these steps in the algorithm fails, a forbidden configuration is present. In this case we just stop and return that M has a chromatic cut. Otherwise we return the correct circular order. □

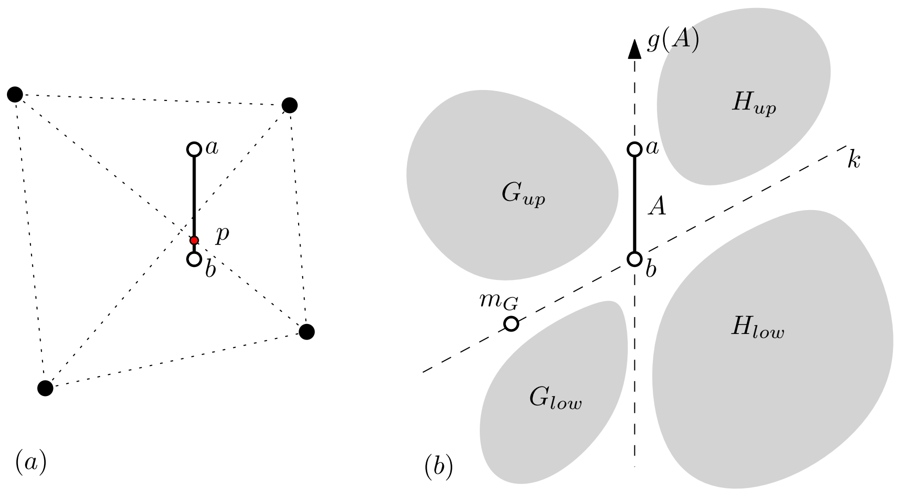


Figure 5.18: The line $g(A)$ splits the point set F into G and H.

5.5.3 Finding a balanced line

The last algorithm we want to present computes a balanced line as in Lemma 5.9. As a preprocessing step we need to find a point on a segment in general position with respect to the remaining points F.

Lemma 5.37. *Let F be a point set in the plane and $A = (a, b)$ be a vertical segment such that $F \cup \{a, b\}$ lies in general position, that is, no three points lie on a line. Then the lowest intersection p of A with a segment formed by two points in F can be computed in deterministic $O(n \log n)$ time.*

Proof. Consider the point sets G and H left and right of $g(A)$. Let m_G be the median of the larger set G, with respect to the order defined by a ray rotating around b. The line k through m_G and b defines the four sets G_{up}, G_{low}, H_{up} and H_{low}, as in Figure 5.18 (b). Now any two points defining the lowest intersection with C are either in G_{up} and H_{up}, or in two opposite sets (that is, G_{up} and H_{low} or G_{low} and H_{up}). The lowest intersecting segment of G_{up} and H_{up} is the convex hull edge of G_{up} and H_{up} intersecting the line $g(A)$. It can be found in linear deterministic time with a subroutine of the convex hull algorithm by Kirkpatrick and Seidel [45] or by an algorithm by Aichholzer, Miltzow and Pilz [5]. The second algorithm only uses order type information. The two opposite sets are treated recursively. Note that $n/4 \leq \#(G_{\mathrm{low}} \cup H_{\mathrm{up}}) \leq 3n/4$ and likewise $n/4 \leq \#(G_{\mathrm{up}} \cup H_{\mathrm{low}}) \leq 3n/4$. Therefore, for the running time we get $T(n) = O(n) + \max_{1/4 \leq \alpha \leq 3/4}[T(\alpha n) + T((1-\alpha)n)]$, which gives $T(n) = O(n \log n)$. □

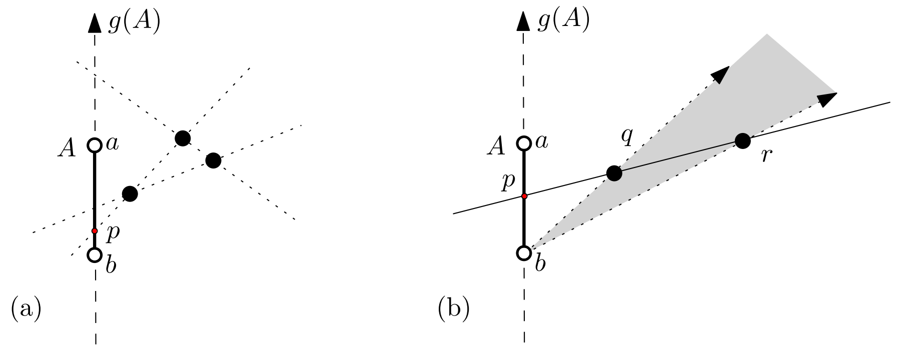


Figure 5.19: (a) the line arrangement formed by the points in F and the lowest crossing with A; (b) the cone with apex b spanned by the minimal pair of points is empty of points of F

For the next Lemma we refer to Figure 5.19.

Lemma 5.38. *Let A be a segment and F a point set right of $g(A)$ in general position. Then the lowest intersection p of A with a line through two points in F can be computed in deterministic $O(n \log n)$ time.*

Proof. First consider the points $q, r \in F$ which form the lowest crossing with A. We show they are neighbors in the radial order around b. Consider the area swept by a ray from q to r. If it contained any point s then either the line through q and s or r and s would have a lower intersection with A.

Thus we merely compute the radial order around b and for any neighboring pair the intersection point with A. The running time $T(n) = O(n \log n)$ is dominated by the sorting procedure. □

Theorem 5.39. *Let F be a point set in the plane and $A = (a, b)$ be a vertical segment such that $F \cup \{a, b\}$ lies in general position (that is, no three points lie on a line). Then the lowest intersection of A with a line through two points in F can be computed in deterministic $O(n \log n)$ time.*

Proof. Compute the lowest intersection point with a line separately for the points left and right of A according to Lemma 5.38 and all possible intersections with A by pairs of points on opposite sites of A according to Lemma 5.37. □

Corollary 5.40. *Given a point set F and a segment A without three points on a line, a point on A in general position with respect to F can be computed in $O(n \log n)$ time.*

Proof. Any point between the lowest intersection and the lower endpoint of A is in general position with respect to F. □

Lemma 5.41. *Let M be a BR-matching of a point set F in general position and A, B be two segments as in Figure 5.2 (a) and (b). Then we can compute a balanced line through the interior of A or B in $O(n \log n)$ time.*

Proof. Let $p \in A$ and $q \in B$ be points as in Corollary 5.40. We know by the proof of Lemma 5.9 that a balanced line through p or q exists. The algorithm in [5] can be adapted to find the desired balanced line through p or q in $O(n)$ time. □

Remark 5.42. *Once we have an $O(n \log n)$ time algorithm to test whether a BR-matching is linear or circular we automatically get an algorithm to test if a BR-matching has a chromatic cut in $O(n \log n)$ time. Note that both algorithms above can be executed until they find a forbidden configuration. Thus we are able to compute a forbidden configuration also in $O(n \log n)$ time. In the case of linear matchings we compute the linear order and for circular matchings the circular order.*

The remaining notions that were introduced in this paper can also be computed efficiently: It is easy to construct a reference line in linear time, as in the Definition 5.16 of quasi-parallel segments. Given a forbidden configuration, it is possible to compute in constant time a chromatic cut (that is, the actual line). Finally, given a forbidden configuration, we can compute a balanced line intersecting one of the segments.

5.6 Open questions

Our method for testing whether a point set F has a unique non-crossing BR-matching starts by finding such a BR-matching M, in $O(n \log n)$ time, by repeated ham-sandwich cuts. This algorithm does not care whether M is unique, and it is in fact the fastest known algorithm for finding *any* non-crossing BR-matching in an arbitrary point set. Is there a faster algorithm for checking whether M is unique (without necessarily constructing M)?

Our paper can also be seen as the study of sets of segments with certain forbidden patterns. These particular segment sets have a lot of nice geometric structure. We wonder whether other forbidden patterns also lead to interesting geometric properties.

Consider n blue, n red and n green points in $\mathbb{R}^3$. By repeatedly applying ham-sandwich cuts we know that there exists a non-crossing colorful 3-uniform geometric matching: Each hyperedge is represented by the convex hull of its vertices. Thus we ask for a geometric characterization of point sets with just one such matching.

Acknowledgments.

We thank Michael Payne, Lothar Narins and Veit Wiechert for helpful discussions, and we thank Stefan Felsner for a hint to the literature. We also thank the anonymous referees for valuable comments.

Acknowledgments

Let me start to thank first the person, who is responsible that I went into the area of discrete mathematics/theoretical computer science. It goes back to autumn 2008, when I attended a preliminary discussion to a seminar with the title "Ausgewählte Kapitel der Graphentheorie". Although I took this course only, because I thought it was an easy way to pass my last seminar, I quickly got excited about the world of discrete mathematics, which is due to Stefan Felsner, who later supervised my Diplom.

Next I want to thank my PHD Supervisor Günter Rote. Among all the things that a supervisor should do most important above all is that he gives you a feeling of trust and encouragement, especially at times when not everything goes smoothly.

I also want to thank all my coauthors, for bearing with me. This is Oswin Aichholzer, Andrei Asinowski, Michael Hoffmann, Balázs Keszegh, Vincent Kusters and Alexander Pilz. Among my coauthors special thanks goes to Andrei Asinowski who also shared long time an office with me. He taught me a lot and I enjoyed many interesting discussions with him.

During this endeavor, I visited several researchers.

I thank Michael Hoffmann and Vincent Kusters to host me in Zürich. I enjoyed many games of Töggele, ice-cream, games evenings, their company in general and of course the collaboration.

I want to thank Oswin Aichholzer and his workgroup to work with me in Graz. Even in this difficult time, when they organized a workshop.

I also want to thank Rom Pinchasi, whom I visited in Haifa at the Technion. I learned a lot from him, about Mathematics, live in general and Israel in particular.

I thank the workgroup, for a friendly and productive atmosphere.

And it should not be forgotten, that I spent during my phd most of my time for teaching. Whenever I felt that students actually listened to me or even learned something from my clumsy explanations, it made me feel excited. Thanks too all students.

At last, I thank my family and friends who went through all the up and downs of the last years with me.

Bibliography

[1] Atila Abdulkadiroğlu and Tayfun Sönmez. Random serial dictatorship and the core from random endowments in house allocation problems. *Econometrica*, 66(3):689–701, 1998.

[2] David J. Abraham, Katarína Cechlárová, David F. Manlove, and Kurt Mehlhorn. Pareto optimality in house allocation problems. In Xiaotie Deng and Ding-Zhu Du, editors, *Algorithms and Computation*, volume 3827 of *Lecture Notes in Computer Science*, pages 1163–1175. Springer Berlin Heidelberg, 2005.

[3] Oswin Aichholzer, Luis Barba, Thomas Hackl, Alexander Pilz, and Birgit Vogtenhuber. Linear transformation distance for bichromatic matchings. In *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*, SOCG'14, pages 154:154–154:162, New York, NY, USA, 2014. ACM.

[4] Oswin Aichholzer, Sergey Bereg, Adrian Dumitrescu, Alfredo García, Clemens Huemer, Ferran Hurtado, Mikio Kano, Alberto Márquez, David Rappaport, Shakhar Smorodinsky, Diane Souvaine, Jorge Urrutia, and David R. Wood. Compatible geometric matchings. *Electronic Notes in Discrete Mathematics*, 31(0):201 – 206, 2008. The International Conference on Topological and Geometric Graph Theory.

[5] Oswin Aichholzer, Tillmann Miltzow, and Alexander Pilz. Extreme point and halving edge search in abstract order types. *Computational Geometry*, 46(8):970–978, 2013.

[6] Noga Alon, Meir Katchalski, and William R. Pulleyblank. Cutting disjoint disks by straight lines. *Discrete & Computational Geometry*, 4:239–243, 1989.

[7] G. Aloupis, L. Barba, S. Langerman, and D. L. Souvaine. Bichromatic compatible matchings. *Computational Geometry: Theory and Applications*, 2014. Also in: Symposium on Computational Geometry, 2013, pp. 267–276, ACM Press; and arXiv:1207.2375 (July 2012).

[8] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 2009.

[9] Andrei Asinowski, Balázs Keszegh, and Tillmann Miltzow. Counting houses of pareto optimal matchings in the house allocation problem. *CoRR*, abs/1401.5354, 2014.

[10] Andrei Asinowski, Balázs Keszegh, and Tillmann Miltzow. Counting houses of pareto optimal matchings in the house allocation problem. In Alfredo Ferro, Fabrizio Luccio, and Peter Widmayer, editors, *FUN*, volume 8496 of *Lecture Notes in Computer Science*, pages 301–312. Springer, 2014.

[11] Andrei Asinowski, Tillmann Miltzow, and Günter Rote. Quasi-parallel segments and characterization of unique bichromatic matchings. *CoRR*, abs/1302.4400, 2013.

[12] Andrei Asinowski, Tillmann Miltzow, and Günter Rote. Quasi-parallel segments and characterization of unique bichromatic matchings. *Journal of Computational Geometry*, to appear.

[13] Haris Aziz, Felix Brandt, and Markus Brill. The computational complexity of random serial dictatorship. *Economics Letters*, 121(3):341–345, 2013.

[14] Haris Aziz and Julian Mestre. Parametrized algorithms for random serial dictatorship. *arXiv*, 2014. arXiv:1403.0974 (March 2014).

[15] Gill Barequet. A lower bound for Heilbronn's triangle problem in d dimensions. *SIAM Journal on Discrete Mathematics*, 14(2):230–236, 2001.

[16] Sergey Bereg, Adrian Dumitrescu, and János Pach. Sliding disks in the plane. *International Journal of Computational Geometry & Applications*, 18(5):373–387, 2008.

[17] Péter Biró, Tamás Fleiner, David Manlove, and Tamás Solymosi, editors. *MATCH-UP 2012: the Second International Workshop on Matching Under Preferences*, Corvinus University of Budapest, Hungary, 2012.

[18] Manuel Blum, Robert W. Floyd, Vaughan Pratt, Ronald L. Rivest, and Robert E. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973.

[19] Hans L. Bodlaender. Complexity of path-forming games. *Theoretical Computer Science*, 110(1):215 – 245, 1993.

[20] Hans L. Bodlaender and Ton Kloks. Fast algorithms for the Tron game on trees. Technical report, Department of Computer Science, 1990.

[21] Katarína Cechlárová, Pavlos Eirinakis, Tamás Fleiner, Dimitrios Magos, Ioannis Mourtos, and Eva Potpinková. Pareto optimality in many-to-many matching problems. *preprint*, 2013.

[22] Josef Cibulka, Jan Kynčl, Viola Mészáros, Rudolf Stolař, and Pavel Valtr. Solution of peter winkler's pizza problem. In Jivrí Fiala, Jan Kratochvíl, and Mirka Miller, editors, *Combinatorial Algorithms*, volume 5874 of *Lecture Notes in Computer Science*, pages 356–367. Springer Berlin Heidelberg, 2009.

[23] Richard Cole, Jeffrey S. Salowe, William L. Steiger, and Endre Szemerédi. An optimal-time algorithm for slope selection. *SIAM J. Comput.*, 18(4):792–810, 1989.

[24] Nadia Creignou and Miki Hermann. On P completeness of some counting problems. Rapport de recherche RR-2144, INRIA, 1993.

[25] Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Cheong. *Computational Geometry: Algorithms and Applications.* Springer-Verlag, Berlin, Germany, 2008.

[26] Nicolaas Govert de Bruijn. Problems 17 and 18. *Nieuw Archief voor Wiskunde*, 2:67, 1954. Answers in *Wiskundige Opgaven met de Oplossingen* 20:19–20, 1955.

[27] Erik D. Demaine and Robert A. Hearn. Playing games with algorithms: Algorithmic combinatorial game theory. In Michael H. Albert and Richard J. Nowakowski, editors, *Games of No Chance 3*, volume 56 of *Mathematical Sciences Research Institute Publications*, pages 3–56. Cambridge University Press, 2009.

[28] Reinhard Diestel. *Graph theory*, volume 3 of *Graduate texts in mathematics*. Springer, 2005.

[29] Sándor P. Fekete, Rudolf Fleischer, Aviezri Fraenkel, and Matthias Schmitt. Traveling salesmen in the presence of competition. *Theoretical Computer Science*, 313(3):377 – 392, 2004.

[30] Aviezri S. Fraenkel, Edward R. Scheinerman, and Daniel Ullman. Undirected edge geography. *Theoretical Computer Science*, 112(2):371 – 381, 1993.

[31] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[32] Alfredo García, Marc Noy, and Javier Tejel. Lower bounds on the number of crossing-free subgraphs of K_n. *Computational Geometry*, 16(4):211–221, August 2000.

[33] Catherine Greenhill. The complexity of counting colourings and independent sets in sparse graphs and hypergraphs. *Computational Complexity*, 9(1):52–72, 2000.

[34] L. J. Guibas and F. F. Yao. On translating a set of rectangles. In *Proc. 12th Annual ACM Symposium Theory of Computing (STOC 1980)*, pages 154–160, 1980.

[35] L. J. Guibas and F. F. Yao. On translating a set of rectangles. In F. P. Preparata, editor, *Computational Geometry*, volume 1 of *Advances in Computing Research*, pages 61–77. JAI Press, Greenwich, Conn., 1983.

[36] Martin Held and Joseph S.B. Mitchell. Triangulating input-constrained planar point sets. *Information Processing Letters*, 109(1):54–56, 2008.

[37] Matthias Henze, Rafel Jaume, and Balász Keszegh. On the complexity of the partial least-squares matching voronoi diagram. In *Proceedings of the 29th European Workshop on Computational Geometry (EuroCG)*, pages 193–196, March 2013.

[38] Michael Hoffmann, Vincent Kusters, and Tillmann Miltzow. Halving balls in deterministic linear time. *CoRR*, abs/1405.1894, 2014.

[39] Michael Hoffmann, Vincent Kusters, and Tillmann Miltzow. Halving balls in deterministic linear time. In Andreas S. Schulz and Dorothea Wagner, editors, *Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings*, volume 8737 of *Lecture Notes in Computer Science*, pages 566–578. Springer, 2014.

[40] Ferran Hurtado, Mikio Kano, David Rappaport, and Csaba D. Tóth. Encompassing colored planar straight line graphs. *Computational Geometry*, 39(1):14–23, 2008.

[41] Ferran Hurtado and Csaba D. Tóth. Plane geometric graph augmentation: a generic perspective. In J. Pach, editor, *Thirty Essays on Geometric Graph Theory*, volume 29 of *Algorithms and Combinatorics*, pages 327–354. Springer, 2013.

[42] A. E. Ingham. On the difference between consecutive primes. *The Quarterly Journal of Mathematics*, 8:255–266, 1937.

[43] Mashhood Ishaque, Diane L. Souvaine, and Csaba D. Tóth. Disjoint compatible geometric matchings. *Discrete Comput. Geom.*, 49(1):89–131, 2013.

[44] Lutz Kettner, Kurt Mehlhorn, Sylvain Pion, Stefan Schirra, and Chee-Keng Yap. Classroom examples of robustness problems in geometric computations. *Computational Geometry*, 40:61–78, May 2008.

[45] David G. Kirkpatrick and Raimund Seidel. The ultimate planar convex hull algorithm? *SIAM Journal on Computing*, 15(1):287–299, 1986.

[46] Kolja Knauer, Piotr Micek, and Torsten Ueckerdt. How to eat 4/9 of a pizza. *Discrete Mathematics*, 311(16):1635 – 1645, 2011.

[47] Hanno Lefmann. On Heilbronn's problem in higher dimension. *Combinatorica*, 23(4):669–680, 2003.

[48] David Lichtenstein and Michael Sipser. Go is polynomial-space hard. *Journal of the ACM*, 27:393–401, April 1980.

[49] Chi-Yuan Lo, Jiří Matoušek, and William L. Steiger. Algorithms for ham-sandwich cuts. *Discrete & Computational Geometry*, 11:433–452, 1994.

[50] Maarten Löffler and Wolfgang Mulzer. Unions of onions: Preprocessing imprecise points for fast onion decomposition. *Journal of Computational Geometry*, 5(1):1–13, 2014.

[51] David Manlove. *Algorithmics of matching under preferences.* World Scientific Publishing, 2013.

[52] Horst Martini and Anita Schöbel. Median hyperplanes in normed spaces – a survey. *Discrete Applied Mathematics*, 89(1):181–195, 1998.

[53] Jiří Matoušek. Efficient partition trees. *Discrete & Computational Geometry*, 8(1):315–334, 1992.

[54] Gary L. Miller, Shang-Hua Teng, William Thurston, and Stephen A. Vavasis. Separators for sphere-packings and nearest neighbor graphs. *J. ACM*, 44(1):1–29, January 1997.

[55] Tillmann Miltzow. Tron, a combinatorial game on abstract graphs. *CoRR*, abs/1110.3211, 2011.

[56] Tillmann Miltzow. Tron , a combinatorial game on abstract graphs. In Evangelos Kranakis, Danny Krizanc, and Flaminia L. Luccio, editors, *FUN*, volume 7288 of *Lecture Notes in Computer Science*, pages 293–304. Springer, 2012.

[57] Noam Nisan. *Algorithmic game theory.* Cambridge University Press, 2007.

[58] János Pach and Micha Sharir. *Combinatorial geometry and its algorithmic applications: The Alcalá lectures*, volume 152 of *Mathematical Surveys and Monographs.* AMS, 2009.

[59] Günter Rote. *Two solvable cases of the traveling salesman problem.* PhD thesis, Technische Universität Graz, 1988.

[60] Günter Rote. The N-line traveling salesman problem. *Networks*, 22:91–108, 1992.

[61] Günter Rote. Partial least-squares point matching under translations. In *26th European Workshop on Computational Geometry (EuroCG 2010)*, pages 249–251, March 2010.

[62] Klaus F. Roth. On a problem of Heilbronn. *Journal of the London Mathematical Society*, 26(3):198–204, 1951.

[63] Daniela Saban and Jay Sethuraman. The complexity of computing the random priority allocation matrix. In Yiling Chen and Nicole Immorlica, editors, *Web and Internet Economics*, volume 8289 of *Lecture Notes in Computer Science*, pages 421–421. Springer Berlin Heidelberg, 2013.

[64] Sandy Scott. *A study of stable marriage problems with ties.* PhD thesis, University of Glasgow, 2005.

[65] Micha Sharir and Emo Welzl. On the number of crossing-free matchings, cycles, and partitions. *SIAM Journal Computing*, 36(3):695–720, 2006.

[66] Michael Sipser. *Introduction to the theory of computation*, volume 2. PWS Publishing Company, 1997.

[67] Helge Tverberg. A separation property of plane convex sets. *Mathematica Scandinavica*, 45:255–260, 1979.

[68] Salil Vadhan. The complexity of counting. B.S. thesis, Harvard University, 1995.

[69] wikipedia foundation. Tron franchise.
http://en.wikipedia.org/, accessed 2013.