Diplomarbeit
Suche in gewichteten Arrangements

angefertigt von Astrid Kaffanke
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

Januar 1999

Inhaltsverzeichnis

Einleitung

Häufig im Leben muß man Dinge suchen und vertut damit – je nachdem, wie gut die gewählte Strategie ist – mehr oder weniger viel Zeit. Sucht man beispielsweise einen Begriff in einem mehrbändigen Lexikon, so ist es nicht besonders geschickt, einfach alle Bände nacheinander durchzublättern, bis man die gewünschte Stelle erreicht hat. Wenn auf derselben Menge häufig Suchen durchgeführt werden, z.B. bei einem elektronischen Lexikon auf dem Rechner, ist es wichtig, effiziente Suchstrategien zu verwenden. Die Suche im Lexikon ist ein eindimensionales Suchproblem, die Reihenfolge der Begriffe ist durch das Alphabet vorgegeben.

Mehrere Lexikonbände bilden eine Zerlegung des eindimensionalen Suchbereiches. Möglicherweise werden die einzelnen Bände verschieden oft nachgefragt, dadurch können sie mit Abfragewahrscheinlichkeiten gewichtet werden. Es ist dann das Ziel, eine optimale Suchstrategie für das gewichtete eindimensionale Suchproblem *Finde zu einem Begriff den richtigen Lexikonband* zu entwickeln.

Für eine solche Suche in einer gewichteten Zerlegung eines eindimensionalen Suchbereiches gibt es Algorithmen, die eine optimale Suchstrategie liefern.

Andere Suchprobleme gibt es in der Ebene. Beim Auswildern von in Gefangenschaft großgezogenen Tieren wird häufig ein kleiner Sender implantiert, der es ermöglicht, die Koordinaten des aktuellen Aufenthaltsortes des Tieres zu bestimmen. Damit ist es möglich, den Weg des Tieres zu verfolgen. Der Lebensraum ist durch Straßen, Wasserläufe, Besiedlung, unterschiedliche Vegetation und ähnliches in verschiedene Regionen aufgeteilt (zur Vereinfachung wird von geradlinigen Unterteilungen ausgegangen). Um aus den Koordinaten auf die entsprechende Region auf einer Landkarte zu schließen, muß eine Suche in der Ebene durchgeführt werden. Für die Suche in einer Unterteilung der Ebene gibt es verschiedene Algorithmen.

Nun kann das Modell noch weiter präzisiert werden. Bei einer genaueren Beobachtung der Lebensgewohnheiten einer Art wird man feststellen, daß bestimmte Regionen gegenüber anderen bevorzugt aufgesucht werden. Dicht besiedelte Gebiete werden gemieden, der Schutz des Waldes dem freien Feld vorgezogen, Täler eher als Anhöhen aufgesucht und ähnliches. So ist es möglich, den verschiedenen Regionen bestimmte Wahrscheinlichkeiten zuzuordnen, daß sich das Tier dort aufhält. Damit ergibt sich ein gewichtetes Suchproblem in der Ebene, eine Verallgemeinerung der eindimensionalen gewichteten Suche.

Eine spezielle Zerlegung der Ebene wird durch ein Arrangement gegeben. Ein

Arrangement ist dabei eine Menge von Geraden in der Ebene, die Regionen sind durch die Geraden begrenzt.

So kommt es zu folgender Problemstellung: Wie kann in einem Arrangement mit gewichteten Regionen zu einem gegebenen Punkt möglichst schnell die Region, in der er liegt, bestimmt werden? Dabei sollen nur Vergleiche zwischen dem Punkt und den Geraden erlaubt sein. Ein Vergleich zwischen einem Punkt und einer Geraden liefert als Ergebnis, ob der Punkt links oder rechts von der Geraden liegt.

Diese Frage ist besonders reizvoll, weil dabei die kombinatorische Suchtheorie mit der algorithmischen Geometrie verbunden wird.

Zu einem solchen Suchproblem läßt sich ein Suchbaum T konstruieren: Die Knoten werden mit den Geraden identifiziert und die Blätter mit den Regionen des Arrangements. Bezeichne $\mathcal{R}$ die Menge aller Regionen eines gegebenen Arrangements, so ist die Anzahl der Fragen, die gebraucht werden, um zu einer Region $r \in \mathcal{R}$ zu gelangen, genau ihre Tiefe $l_T(r)$ im Suchbaum.

Trägt nun jede Region r eine gewisse Wahrscheinlichkeit α_r, daß ein gemäß der Wahrscheinlichkeitsverteilung gegebener Punkt in ihr liegt, so ist das Ziel ein Baum T, der im Durchschnitt wenig Abfragen braucht, d.h. $p(T) = \sum_{r \in \mathcal{R}} l_T(r)\alpha_r$ soll minimiert werden.

Die Arbeit ist folgendermaßen gegliedert: Zuerst werden Grundlagen bereitgestellt, indem der bekannte eindimensionale Fall, also Suchbäume für gewichtete Intervalle, betrachtet wird. Dabei werden Schranken für die Güte eines optimalen Suchbaumes angegeben, und ein Konstruktionsalgorithmus wird vorgestellt.

Ausgehend davon habe ich die vorgestellten Ideen weiterentwickelt. Dabei wird zunächst exemplarisch die Verallgemeinerung auf den zweidimensionalen Fall vorgeführt. In diesem Fall ist der Suchbereich, in dem ein Punkt gefunden werden soll, ein Gitter mit gewichteten Zellen.

Der folgende Schritt ist die Verallgemeinerung auf ein d-dimensionales Gitter, das durch Scharen paralleler Hyperebenen den Raum in gewichtete Zellen aufteilt. Für diesen Fall habe ich Schranken für die Güte eines optimalen Suchbaumes bewiesen sowie ein Konstruktionsverfahren angegeben.

Anschließend habe ich gezeigt, daß damit auch ein anderes Problem gelöst ist: Die Suche nach einem gegebenen Punkt in einem Arrangement von d Parallelenscharen mit gewichteten Regionen in der Ebene.

Im zweiten Teil der Arbeit wird die Struktur von Arrangements näher untersucht. Ausschlaggebend dafür war die Überlegung, daß ein optimaler Suchbaum nicht allein von der gegebenen Gewichtung der Regionen abhängt, sondern ebenso von der Lage der Geraden zueinander. Wenn durch einen

Punkt-Geraden-Vergleich jeweils nur eine geringe Anzahl von Regionen ausgeschlossen werden kann, wird der Suchbaum zwangsläufig sehr tief werden.

Damit stellte sich mir die Frage, ob es in einem Arrangement immer eine Gerade gibt, die die Anzahl der rechts und links von ihr liegenden Regionen „gut" aufteilt. Ich kam dabei zu Ergebnissen, die von der Größe des betrachteten Arrangements abhängen. Beispielsweise gibt es in einem Arrangement mit $n \geq 12$ Geraden immer eine Gerade, bei der auf jeder Seite mindestens $8,6\%$ aller Regionen liegen.

Allerdings reicht auch das noch nicht aus, denn nach dem ersten Punkt-Geraden-Vergleich muß der Punkt nicht mehr in einem Arrangement, sondern nur noch in einem Teil davon lokalisiert werden. Es war nun das Ziel, auch in solchen „partiellen Arrangements" immer eine Gerade zu finden, die die Regionen möglichst gut balanciert. Dabei zeigte sich, daß eine solche Gerade in vielen partiellen Arrangements zu finden ist, daß es jedoch bei ungünstigen Konstellationen keine solche die Regionen balancierende Gerade gibt.

Abgesehen von Artikeln über eindimensionale Suchprobleme und ungewichtete Suche in der Ebene habe ich keine Literatur gefunden, die solche oder ähnliche Fragestellungen behandelt.

Dank der guten Betreuung durch Stefan Felsner, der immer wieder neue Ideen und Sichtweisen anbrachte, bin ich daran jedoch nicht verzweifelt, sondern habe im Gegenteil mit wachsendem Spaß an den Problemen gearbeitet.

Dank gebührt auch meinen kritischen Korrekturlesern und -leserinnen, vor allem Marco. Seine liebevolle Korrektur hat die Arbeit unter anderem um über 50 Kommata bereichert.

Carsten danke ich für den großartigen gemeinsamen Endspurt und allen anderen für die gute Versorgung mit frischgekochtem Tee, wunderbarer Musik passend zu allen Gemütszuständen und Aufmunterung im richtigen Augenblick sowie dafür, daß sie meine Launen der letzten Zeit relativ klaglos ertragen haben.

1 Suchbäume für eindimensionale Suche

In diesem Kapitel werden Suchprobleme auf einem linear geordneten Suchbereich behandelt. Ein Beispiel dafür ist die Suche nach einem bestimmten Begriff in einem Wörterbuch: Die Worte sind lexikographisch sortiert und durch Vergleiche mit jeweils dem ersten Wort auf einer Seite wird festgestellt, ob der gesuchte Begriff weiter vorne oder weiter hinten zu finden ist.

Das Ziel ist es, ein Suchproblem so zu strukturieren, daß ein Objekt – wie hier das Wort – mit möglichst wenigen Vergleichen lokalisiert werden kann.

Im folgenden werden zuerst Grundlagen bereitgestellt, um dann eine Suchstruktur für ein spezielles Problem zu entwickeln und zu analysieren.

1.1 Bäume

Grundlegend für die Lösung von Suchproblemen sind Bäume. Dieser Begriff wird hier erklärt und einige Eigenschaften werden gezeigt.

Definition 1.1 *Es sei V eine endliche Menge, genannt* Knotenmenge.

- *Jeder Knoten $v \in V$ ist ein* Baum. *Dann wird v als die* Wurzel *dieses Baumes bezeichnet.*

- *Sind $T_1, \ldots, T_k$ Bäume mit disjunkten Knotenmengen und v ein Knoten, der in keinem der T_i vorkommt, so ist das Tupel $(v, T_1, \ldots, T_k)$ ein* Baum *mit der* Wurzel *v und den* Unterbäumen *$T_1, \ldots, T_k$.*

Ist T ein Baum mit der Wurzel v und Unterbäumen $T_1, \ldots, T_k$ mit Wurzeln $v_1, \ldots, v_k$, so heißen $v_1, \ldots, v_k$ die Nachfolger *von v, und v ist der* Vorgänger *der v_i, $\quad i = 1 \ldots n$.*
Ein Knoten ohne Nachfolger heißt Blatt, *alle anderen Knoten* innere Knoten.
Ein binärer Baum *ist ein Baum, in dem jeder Knoten höchstens zwei Nachfolger hat, in einem* vollständigen binären Baum *hat jeder Knoten genau zwei oder gar keinen Nachfolger.*
Ein Weg *in T ist eine Folge von Knoten $w_0, ..., w_m$, so daß w_i Nachfolger von w_{i-1} ist für $i = 1...m$. Die* Länge *dieses Weges ist m.*

In der graphischen Darstellung von Bäumen werden Unterbäume durch Dreiecke, innere Knoten durch Kreise und Blätter durch Quadrate symbolisiert. Die Vorgänger-Nachfolger-Beziehung wird durch *Kanten* zwischen den entsprechenden Knoten dargestellt, siehe Abbildung 1. Da bei einem binären Baum von jedem Knoten v nur zwei Unterbäume ausgehen können, ist es hier sinnvoll, vom *rechten* und *linken* Unterbaum von v zu sprechen.

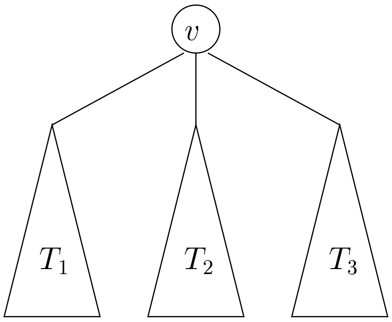


Abbildung 1: Ein Baum mit Wurzel v und drei Unterbäumen.

Bemerkung. In einem Baum gibt es einen eindeutigen Weg von der Wurzel des Baumes zu jedem anderen Knoten.

Definition 1.2 *Das* Niveau (engl. level) *eines Knotens v in einem Baum T ist die Länge des eindeutigen Weges von der Wurzel zu v und wird mit $l_T(v)$ bezeichnet. Die* Höhe *eines Baumes ist das Maximum der Niveaus der Knoten dieses Baumes. Ein Baum heißt* balanciert*, falls sich das Niveau von je zwei Blättern um höchstens eins unterscheidet.*

Eine einfache Beobachtung liefert das folgende Lemma.

Lemma 1.1 *Ein vollständiger binärer Baum T hat genau ein Blatt mehr als innere Knoten.*

Beweis. Induktion über die Anzahl der Knoten.
Wenn es nur einen Knoten gibt, besteht T aus einem Blatt und hat keinen inneren Knoten.
Sei T ein Gegenbeispiel mit kleinstmöglicher Knotenanzahl. T habe die Höhe l und b sei ein Blatt auf dem Niveau l. Der eindeutige Vorgänger von b sei a. Weil T vollständig ist, hat a einen weiteren Nachfolger $b' \neq b$, und b' ist auch ein Blatt.

Betrachte den Baum $T' := T - \{b, b'\}$. Nach Konstruktion ist auch T' vollständig und hat nach Voraussetzung genau ein Blatt mehr als innere Ecken. Es gilt
$\#$ Blätter von $T = \#$ Blätter von $T' + 1$ und
$\#$ innere Knoten von $T = \#$ innere Knoten von $T' + 1$. □

Die folgende Ungleichung klärt, wann es einen binären Baum mit n Blättern auf den Niveaus $l(1), \ldots, l(n)$ gibt.

Lemma 1.2 (Kraftsche Ungleichung) .

(i) Für einen binären Baum T mit Blättern $1, \ldots, n$ gilt: $\sum_{i=1}^{n} 2^{-l_T(i)} \leq 1$ *mit Gleichheit genau dann, wenn T vollständig ist.*

(ii) Sind natürliche Zahlen $l(1), \ldots, l(n)$ mit $\sum_{i=1}^{n} 2^{-l(i)} \leq 1$ *gegeben, so gibt es einen binären Baum T mit n Blättern, bei dem die Blätter genau auf den Niveaus $l(i)$ sind.*

Beweis.

(i) Jedem Knoten auf dem Niveau k wird der Wert 2^{-k} zugeordnet, siehe Abbildung 2. Damit ist für jeden Knoten die Summe der Werte der direkten Nachfolger höchstens so groß wie der Wert des Knotens. Induktiv folgt, daß für jeden Unterbaum von T die Summe der Werte aller Blätter höchstens so groß wie der Wert der Wurzel des Unterbaumes ist. Somit gilt dies auch für T selbst. Gleichheit gilt nur, falls jeder innere Knoten genau zwei Nachfolger hat, also für einen vollständigen binären Baum.

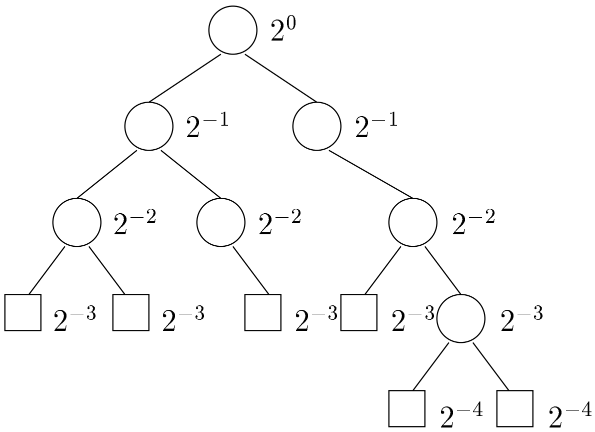


Abbildung 2: „Durchsickern“ von $1 = 2^0$.

(ii) Es gelte $\sum_{i=1}^{n} 2^{-l(i)} \leq 1$, und es sei $l := \max l(i)$ sowie $b_k := |\{i : l(i) = k\}|$, $k = 0 \ldots l$. b_k wird die Anzahl der Blätter des zu konstruierenden Baumes auf dem Niveau k sein. Die Ungleichung läßt sich nun schreiben als
$b_0 2^l + b_1 2^{l-1} + \ldots + b_{l-1} 2^1 + b_l \leq 2^l$.

T wird induktiv konstruiert:

- Falls $b_0 = 1$, so ist $l = 0$, und T besteht nur aus einem Knoten, der Wurzel.
- Sei T schon konstruiert bis zum Niveau k. Es interessiert jetzt $I_k :=$ # innere Ecken auf dem Niveau k. Es kann maximal 2^k Knoten auf dem Niveau k geben, aber unterhalb der Blätter auf den Niveaus 0 bis k folgen keine weiteren Knoten. Daher ist

$$I_k = 2^k - \sum_{i=0}^{k} b_i 2^{k-i},$$

und aus der Ungleichung folgt

$$b_{k+1} 2^{l-k-1} \leq 2^l - \sum_{i=0}^{k} b_i 2^{l-i}$$

und damit

$$b_{k+1} \leq 2^{k+1} - \sum_{i=0}^{k} b_i 2^{k+1-i} = 2\left(2^k - \sum_{i=0}^{k} b_i 2^{k-i}\right) = 2I_k.$$

Also können alle Blätter des Niveaus $k+1$ plaziert werden. $\square$

1.2 Suchbäume

Es wird nun an einem Beispiel erklärt, wie ein Suchproblem auf einer linear geordneten Menge, d. h. alle Elemente sind miteinander vergleichbar, mit einem geeigneten Baum identifiziert werden kann. Das vorgestellte Verfahren ist insofern universell, als es auf *jede* Suche auf einem linear geordneten Suchbereich übertragen werden kann.

Ein einfaches Suchproblem ist das folgende: Ein Intervall $[x_0; x_n]$ der reellen Achse wird durch $S = \{x_1 < \ldots < x_{n-1}\}$ in Teilintervalle aufgeteilt, und zu einer Zahl x soll ermittelt werden, in welchem der Teilintervalle sie liegt. Es ist also ein x_i zu finden, so daß $x = x_i$, oder ein Intervall $(i) :=]x_{i-1}; x_i[$, so daß $x_{i-1} < x < x_i,\ i = 1 \ldots n$. Ein Beispiel für einen solchen Suchbereich zeigt Abbildung 3. Die Suche kann auf alle reellen Zahlen ausgedehnt werden, indem $x_0 := -\infty$ und $x_n := +\infty$ gesetzt werden.

Zu diesem Suchproblem kann ein Baum konstruiert werden, bei dem die inneren Knoten mit $x_1, \ldots, x_{n-1}$ identifiziert werden und die Blätter mit den Intervallen (i) , $i = 1 \ldots n$, so daß gilt:

Ist x_i im linken und x_j im rechten Unterbaum eines Baumes mit der Wurzel x_k, so ist $x_i < x_k < x_j$. Ist der linke Teilbaum an einem Knoten x_i nur

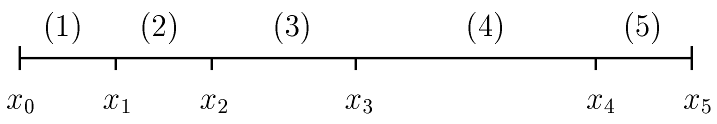


Abbildung 3: Suchbereich mit Unterteilungen.

ein Blatt, so repräsentiert es das offene Intervall $]x_{i-1}; x_i[$ und entsprechend wenn der rechte Teilbaum nur ein Blatt ist, repräsentiert es das offene Intervall $]x_i; x_{i+1}[$. Die Blätter müssen somit nicht explizit angegeben werden, ihr Inhalt ist durch die inneren Knoten eindeutig festgelegt.

Ein Baum mit diesen Eigenschaften wird *Suchbaum* oder, in Anlehnung an das Wörterbuchproblem, *alphabetischer Baum* genannt. Insbesondere heißt das: In einem Suchbaum nehmen die Werte der Blätter von links nach rechts gelesen zu.

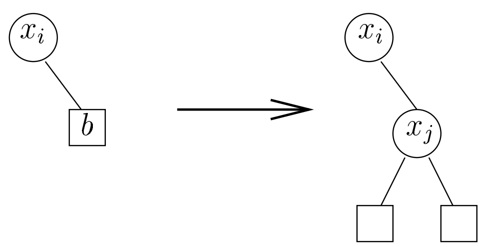
Die Suche nach einem Wert x in einem Suchbaum T läßt sich folgendermaßen realisieren:

Suche x in T

- Ist x die Wurzel von T, so ist die Suche erfolgreich beendet.
- Ist die Wurzel $> x$, so setze die Suche im linken Unterbaum fort.
- Ist die Wurzel $< x$, so setze die Suche im rechten Unterbaum fort.
- Wird ein Blatt erreicht, so liegt x in dem von dem Blatt repräsentierten offenen Intervall.

Zu einer gegebenen Menge $\{x_1, \ldots, x_{n-1}\}$ läßt sich ein Suchbaum durch sukzessives Einfügen der Elemente erzeugen:

- Wähle ein beliebiges $i \in \{1, \ldots, n-1\}$, x_i sei die Wurzel des Suchbaumes mit den Blättern b_1 und b_2.
- Solange es ein nicht einsortiertes x_j gibt: Suche x_j in T.
  Endet die Suche in einem Blatt b, so ersetze es:



Für verschiedene Einfügereihenfolgen können verschiedene Suchbäume entstehen. Als Suchbäume sind nur vollständige binäre Bäume interessant. Wenn ein Knoten nur einen Nachfolger hat, ist diese Abfrage nicht notwendig, und der Knoten kann gestrichen werden. Im weiteren wird ein Suchbaum von vornherein als vollständig angesehen.

Interessant ist nun, wie ein Suchbaum beschaffen sein muß, damit ein unbekanntes Element x mit möglichst wenigen Abfragen lokalisiert werden kann. Dabei ist weniger entscheidend, wieviele Vergleiche im schlechtesten Fall benötigt werden, als vielmehr der durchschnittliche Wert. Ein sinnvolles Maß für die Güte eines Suchbaumes T ist somit die *durchschnittliche Pfadlänge*

$$p(T) := \frac{1}{\#\text{Blätter von } T} \sum_{b \text{ Blatt von } T} l_T(b).$$

T_0 ist ein *optimaler Suchbaum* mit n Blättern, falls

$$p(T_0) = \min_{T \text{ Suchbaum mit } n \text{ Blättern}} p(T).$$

Der durch zufälliges Einfügen konstruierte Suchbaum ist im allgemeinen nicht optimal, aber gut. Ein optimaler Suchbaum entsteht, wenn durch jede Abfrage die Zahl der möglichen Ergebnisse halbiert wird. Die Suche nach x benötigt dann höchstens $\lceil \log_2 n \rceil$ Abfragen, der so konstruierte Suchbaum ist balanciert. In Kapitel 1.6 wird die Optimalität dieses Baumes bewiesen.

1.3 Gewichtete Suchbäume

In den meisten Fällen wird die Wahrscheinlichkeit, daß ein Wert x in einem Intervall (i) liegt, nicht für jedes Intervall gleich groß sein. In der folgenden Definition wird der Begriff der Wahrscheinlichkeit präzisiert.

Definition 1.3 *Eine* diskrete Wahrscheinlichkeitsverteilung $\alpha_1,\ldots,\alpha_n$ *ist gegeben, falls* $\alpha_i \geq 0$ *für* $i = 1 \ldots n$ *und* $\sum_{i=1}^{n} \alpha_i = 1$.

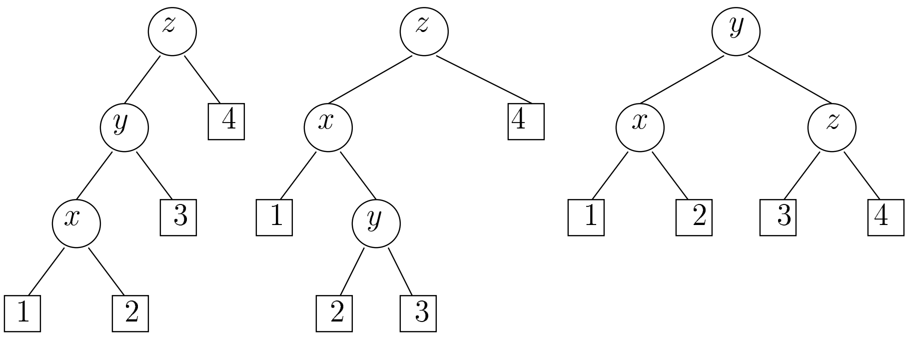
Die Wahrscheinlichkeit, daß x in (i) liegt, werde mit α_i bezeichnet, häufig wird α_i dann auch das *Gewicht* der Zelle (i) genannt. Ein *optimaler* gewichteter Suchbaum T_0 mit n Blättern für $\alpha_1, \ldots, \alpha_n$ ist ein Suchbaum, der die *gewichtete Pfadlänge*

$$p(T) := \sum_{i=1}^{n} l_T(i)\alpha_i$$

minimiert. Huffman [9] entwickelte ein einfaches Verfahren, wie zu einer gegebenen Wahrscheinlichkeitsverteilung ein optimaler Baum konstruiert werden kann. Dabei werden jedoch nur die Gewichte betrachtet, und nicht die Stelle ihres Auftretens. Es kann daher beispielsweise passieren, daß im linken Teilbaum eines Hufman-Baumes mit Wurzel x_k ein Knoten x_i steht mit $x_i > x_k$. Deshalb ist ein solcher Huffman-Baum im allgemeinen kein Suchbaum. Durch die geforderte Ordnung der Knoten ist das Problem *finde optimalen Suchbaum* deutlich schwieriger zu lösen als die Aufgabe *finde optimalen Baum*.

Bemerkung. Der besseren Übersichtlichkeit halber beschränken sich die weiteren Untersuchungen auf den Fall, daß die Wahrscheinlichkeit für $x = x_i$ Null ist für alle Intervallgrenzen x_i. Es interessieren also nur die Weglängen zu den Blättern (i), die die offenen Intervalle $]x_{i-1}; x_i[$ repräsentieren. Das Konzept läßt sich jedoch problemlos verallgemeinern auf den Fall echt positiver Wahrscheinlichkeiten für die Intervallgrenzen, wie beispielsweise bei Knuth [11].

Es gibt 5 verschiedene Suchbäume mit den inneren Knoten x, y und z, $x < y < z$. Jeder dieser Bäume ist für bestimmte gegebene Wahrscheinlichkeiten optimal:



Baum 1 Baum 2 Baum 3

Abbildung 4: Verschiedene Suchbäume mit den inneren Knoten x, y und z.

	α_1	α_2	α_3	α_4	$p(t)$
Baum 1	0,01	0,01	0,18	0,80	1,22
Baum 2	0,18	0,01	0,01	0,80	1,22
Baum 3	0,25	0,25	0,25	0,25	2,00
Baum 4	0,80	0,18	0,01	0,01	1,22
Baum 5	0,80	0,01	0,01	0,18	1,22

Für die nebenstehende Tabelle sei Baum 4 der achsensymmetrische Baum zu Baum 1 mit den entsprechenden Bezeichnungen der Knoten und genauso Baum 5 zu Baum 2.

Gewichtungen, zu denen die Bäume optimal sind.

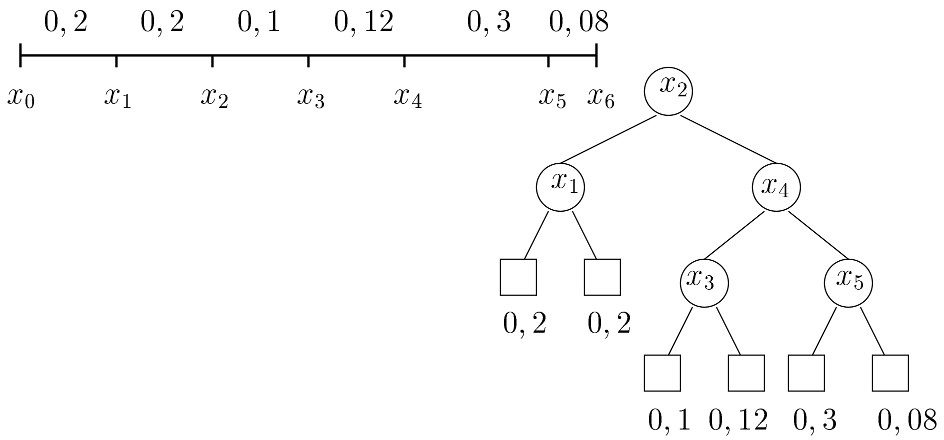


Abbildung 5: Das Intervall mit Wahrscheinlichkeit 0,30 steht im optimalen Suchbaum tiefer als das mit Wahrscheinlichkeit 0,20. Die Gewichte des rechten und linken Unterbaumes sind nicht gleich groß. $p(T) = 2,60$.

Haben alle Intervalle das gleiche Gewicht, so ist das Problem äquivalent zum ungewichteten Fall, und wie später noch gezeigt wird, ist dann ein balancierter Suchbaum optimal. Es kann passieren, daß im optimalen Suchbaum ein Intervall mit größerer Wahrscheinlichkeit auf einem tieferen Niveau steht als eines mit kleinerer Wahrscheinlichkeit.

Iverson [10] vermutete, daß bei einem optimalen Suchbaum die Wahrscheinlichkeiten für den rechten und linken Unterbaum so gut es geht gleich groß sein müssen. Das ist, wie Abbildung 5 zeigt, aber nicht notwendig.

Eine wichtige Beobachtung ist die folgende:
Jeder Unterbaum eines optimalen Suchbaumes T ist ein optimaler Suchbaum auf seiner Blattmenge mit entsprechend normierten Gewichten. Denn angenommen T hat einen Unterbaum T', der auf seiner Blattmenge nicht optimal ist. Dann kann T' durch einen optimalen Suchbaum auf dieser Blattmenge ersetzt werden, und es entsteht ein Baum, der geringere gewichtete Pfadlänge als T hat, was im Widerspruch zur Optimalität von T steht.

Ist also x_i die Wurzel eines optimalen Suchbaumes T, so ist der linke Unterbaum ein optimaler Suchbaum für die gewichteten Intervalle $(1), \ldots, (i)$, und der rechte Unterbaum ist optimal für $(i+1), \ldots, (n)$.

Bemerkung. Im weiteren wird angenommen, daß für alle Intervalle $(1), \ldots, (n)$ die Wahrscheinlichkeit $\alpha_i \neq 0$ ist. Das ist jedoch keine Beschränkung, wenn Intervalle mit $\alpha_i = 0$ bei der Konstruktion eines optimalen

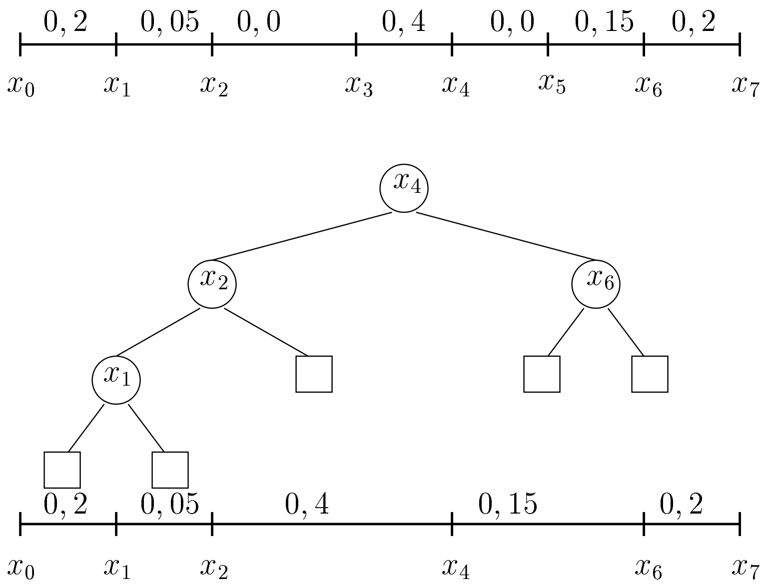


Abbildung 6: Behandlung von Intervallen mit Wahrscheinlichkeit Null.

Suchbaumes folgendermaßen behandelt werden:
Es wird im Fall $\alpha_k = 0$ das Intervall (k) mit einem angrenzenden Intervall, beispielsweise $(k+1)$, verschmolzen, und das neue Intervall bekommt als Wahrscheinlichkeit α_{k+1}. Dann ist x_k kein innerer Knoten und (k) kein Blatt des Suchbaumes, es fallen also genau ein innerer Knoten und ein Blatt weg. Ein Beispiel dazu zeigt Abbildung 6. Später wird das Ergebnis $x \in]x_{k-1}; x_{k+1}[$ interpretiert als $x \in (k+1)$.

Damit kann ohne Beschränkung angenommen werden, daß alle Intervalle $(1), \ldots, (n)$ echt positive Wahrscheinlichkeiten haben und $x_1, \ldots, x_{n-1}$ die inneren Knoten des Suchbaumes sind. Von einem durch Verschmelzen entstandenen Intervall ist ein Zeiger auf das ursprüngliche Intervall mit echt positiver Wahrscheinlichkeit gesetzt.

Diese Vorbehandlung braucht nur linearen Aufwand, da es reicht, die Intervalle einmal ihrer Reihenfolge nach durchzugehen.

1.4 Die Entropie einer diskreten Wahrscheinlichkeitsverteilung

Eine wichtige Größe für die Abschätzung von $p(T)$ ist die *Entropie* einer diskreten Wahrscheinlichkeitsverteilung. Sie ist ein Maß für die Unsicherheit: Je weniger die Verteilung eine Aussage über das mögliche Ergebnis einer

Suche zuläßt, umso größer ist ihre Entropie.

Es wird später gezeigt, daß die gewichtete Pfadlänge eines optimalen Suchbaumes bis auf eine Konstante durch die Entropie bestimmt ist.

Definition 1.4 *Zu einer diskreten Wahrscheinlichkeitsverteilung* $\alpha_1,\ldots,\alpha_n$ *ist die* ideelle Entropie $H(\alpha_1,\ldots,\alpha_n)$ *definiert als*

$$H(\alpha_1,\ldots,\alpha_n) := -\sum_{i=1}^{n} \alpha_i \log_2 \alpha_i \quad ,$$

wobei $0 \log_2 0 := 0$ *gesetzt wird.*

Bemerkung.

1. Für zwei diskrete Wahrscheinlichkeitsverteilungen $\alpha_1,\ldots,\alpha_n$ und $\beta_1,\ldots,\beta_n$ gilt:
$$H(\alpha_1,\ldots,\alpha_n) \leq -\sum_{i=1}^{n} \alpha_i \log_2 \beta_i$$
mit Gleichheit genau dann, wenn $\alpha_i = \beta_i$ für $i = 1 \ldots n$.

2. Für jede diskrete Wahrscheinlichkeitsverteilung gilt $0 \leq H(\alpha_1,\ldots,\alpha_n) \leq \log_2 n$.

Beweis.

1. $$-\sum_{i=1}^{n} \alpha_i \log_2 \alpha_i \leq -\sum_{i=1}^{n} \alpha_i \log_2 \beta_i \qquad \Longleftrightarrow \qquad \sum_{i=1}^{n} \alpha_i \log_2 \left(\frac{\beta_i}{\alpha_i}\right) \leq 0$$
Da $\log x \leq x - 1$ und $\log_2 x = \frac{\log x}{\log 2}$, ergibt sich
$$\begin{aligned} \sum_{i=1}^{n} \alpha_i \log_2 \left(\frac{\beta_i}{\alpha_i}\right) &= \frac{1}{\log 2} \sum_{i=1}^{n} \alpha_i \log \left(\frac{\beta_i}{\alpha_i}\right) \leq \frac{1}{\log 2} \sum_{i=1}^{n} \alpha_i \left(\frac{\beta_i}{\alpha_i} - 1\right) \\ &= \frac{1}{\log 2} \left(\sum_{i=1}^{n} \beta_i - \sum_{i=1}^{n} \alpha_i\right) = 0 \ , \end{aligned}$$
und Gleichheit gilt, falls $\log\left(\frac{\beta_i}{\alpha_i}\right) = \frac{\beta_i}{\alpha_i} - 1$ für alle i. Das ist der Fall genau dann, wenn $\alpha_i = \beta_i$ für alle i, und damit folgt die Behauptung.

2. $H \geq 0$ ist klar, denn die α_i liegen zwischen 0 und 1, und damit ist jeder der Summanden negativ, der gesamte Ausdruck wird positiv. Für $H \leq \log_2 n$ sei im Beweis von 1. $\beta_i = \frac{1}{n}$ für alle i. $\square$

1.5 Abschätzung der Pfadlänge eines optimalen Suchbaumes

Nun wieder zurück zum ursprünglichen Problem. Gegeben ist ein Intervall $[x_0; x_n]$ mit den Unterteilungen $x_1, \ldots, x_{n-1}$. (i) bezeichne das Intervall $]x_{i-1}; x_i[$ und α_i sei die Wahrscheinlichkeit, daß ein gegebener Wert x im Intervall (i) liegt. Wieviele Abfragen werden durchschnittlich benötigt, um das Intervall zu bestimmen, in dem x liegt? Eine offensichtliche obere Schranke dafür ist die Höhe des balancierten Suchbaumes mit n Blättern, also $\lceil \log_2 n \rceil$. Eine bessere Abschätzung liefert der folgende Satz, der Beweis orientiert sich an [2].

Satz 1.3 *Die gewichtete Pfadlänge $p(T_0)$ eines optimalen Suchbaumes T_0 für dieses Problem erfüllt $H(\alpha_1, \ldots, \alpha_n) \leq p(T_0) \leq H(\alpha_1, \ldots, \alpha_n) + 2$.*

Beweis. Die untere Schranke ist einfach zu zeigen: $l_T(i)$ bezeichne wieder das Niveau des Blattes im Suchbaum, das das Intervall (i) repräsentiert. Für alle i ist $2^{-l_T(i)}$ positiv, und nach Lemma 1.2 ist $\sum_{i=1}^{n} 2^{-l_T(i)} = 1$. Damit ist $\beta_i := 2^{-l_T(i)}$ eine Wahrscheinlichkeitsverteilung, und mit der Bemerkung aus Kapitel 1.4 folgt die erste Ungleichung.

Nun zur oberen Schranke. Es reicht, einen Suchbaum zu konstruieren, dessen gewichtete Pfadlänge die Ungleichung erfüllt. Ein optimaler Suchbaum ist höchstens besser. Für die Konstruktion werden die folgenden Hilfsgrößen verwendet: $r_i := \sum_{j=1}^{i-1} \alpha_j + \frac{\alpha_i}{2}$; $l_T(i) := \lceil -\log_2 \alpha_i \rceil + 1$. Weiter sei $w_i :=$ die ersten $l_T(i)$ Ziffern der binären Darstellung von r_i.

Nun werden die Ziffern von w_i interpretiert als Weg im Wurzelbaum von der Wurzel zu dem Blatt (i), und zwar geht man nach links bei einer 0 und nach rechts bei einer 1. Es entsteht ein Baum T.

Behauptungen:

1. Jedes (i) ist ein Blatt in T mit Niveau $l_T(i)$.
2. T ist Suchbaum für die (i).
3. $p(T) \leq H(\alpha_1, \ldots, \alpha_n) + 2$.
4. Es gibt einen vollständigen binären Suchbaum für das Problem, der höchstens besser ist.

Beweis.

1. Zu zeigen ist, daß der Weg von der Wurzel zu (i) nicht Anfangsstück des Weges zu einem (j), $j \neq i$, ist. Angenommen es wäre so. Dann hätten w_i und w_j dieselben $l_T(i)$ Anfangsziffern. Aus der Definition von w_i und $l_T(i)$ folgt somit $|r_i - r_j| < \frac{1}{2^{l_T(i)}} \leq \frac{1}{2^{-\log_2 \alpha_i + 1}} = \frac{\alpha_i}{2}$. Das ist ein Widerspruch, denn aus der Definition der r_i folgt $|r_i - r_j| \geq \frac{\alpha_i}{2} + \frac{\alpha_j}{2} > \frac{\alpha_i}{2}$. Der Weg von der Wurzel nach (i) hat nach Konstruktion die Länge $l_T(i)$.

2. Offensichtlich ist $r_1 < r_2 < \ldots < r_n$. Das gilt dann natürlich auch für die binäre Übersetzung w_i und damit für die Wege: Wenn $i < j$, dann ist (i) im Suchbaum links von (j).

3. Nachrechnen:

$$\begin{aligned} p(T) &= \sum_{i=1}^{n} \alpha_i l_T(i) = \sum_{i=1}^{n} \alpha_i(\lceil -\log_2 \alpha_i \rceil + 1) \\ &\leq \sum_{i=1}^{n} \alpha_i((-\log_2 \alpha_i + 1) + 1) = H(\alpha_1, \ldots, \alpha_n) + 2. \end{aligned}$$

4. Falls T nicht vollständig ist, entsteht durch Streichen von Knoten mit nur einem Nachfolger ein vollständiger Baum mit gleichen Eigenschaften, dessen gewichtete Pfadlänge kleiner als die von T ist.

Da ein vollständiger binärer Baum mit n Blättern genau $n-1$ innere Knoten hat, lassen sich die Abfragepunkte $x_1, \ldots, x_{n-1}$ eindeutig den inneren Knoten des Baumes zuordnen.

Für den optimalen Suchbaum T_0 ist $p(T_0) \leq p(T)$, und damit ist die obere Schranke bewiesen. □

1.6 Konstruktion eines optimalen Suchbaumes

Nachdem nun gezeigt ist, daß sich die gewichtete Pfadlänge eines optimalen Suchbaumes bis auf einen Fehler von 2 bestimmen läßt, stellt sich die Frage, wie ein solcher Baum zu einer gegebenen Wahrscheinlichkeitsverteilung erzeugt werden kann.

Es ist schon in Kapitel 1.3 erwähnt worden, daß für den Fall der Gleichverteilung ein balancierter Suchbaum bestmöglich ist. Das ist mit der Abschätzung aus Satz 1.3 sofort zu sehen: Für einen Suchbaum T mit n Blättern sei $\alpha_i = \frac{1}{n}$, damit ist

$$p(T) \geq -\sum_{i=1}^{n} \frac{1}{n} \log_2 \frac{1}{n} = \log_2 n.$$

Für eine Zweierpotenz n erreicht ein balancierter Suchbaum diese Schranke.

Um für eine beliebige Wahrscheinlichkeitsverteilung einen optimalen Suchbaum zu konstruieren, muß etwas mehr Aufwand betrieben werden. Das im folgenden vorgestellte Verfahren wurde 1959 von Gilbert und Moore [6] entwickelt und basiert auf dem Prinzip des *dynamischen Programmierens.* Wie bereits erwähnt, ist jeder Unterbaum eines optimalen Suchbaumes wieder optimal. Darum kann der optimale Suchbaum für $[x_0; x_n]$ konstruiert werden, indem ausgehend von den kleinsten Teilintervallen die optimalen Suchbäume für alle Teilintervalle $[x_i; x_j]$ konstruiert werden.

Bemerkung. Das ist mit vertretbarem Aufwand berechenbar, denn die Anzahl der Möglichkeiten, i und j zu wählen, so daß $1 \leq i < j \leq n$, ist $\frac{1}{2}n(n-1)$.

Definition 1.5 *Es sei T_l der linke und T_r der rechte Unterbaum eines Baumes T. Für einen Unterbaum T' von T ist das* Gewicht $w(T')$ *die Summe der Wahrscheinlichkeiten der Blätter, die in T' auftauchen, also*

$$w(T') := \sum_{(i) \text{ Blatt von } T'} \alpha_i.$$

Bemerkung. Es gilt $p(T) = p(T_l) + p(T_r) + w(T)$ für jeden Suchbaum T.

Definition 1.6 *$T_{i,j}$ bezeichne den optimalen Suchbaum für den Suchbereich $[x_{i-1}; x_j]$ und $p(i,j)$ die gewichtete Pfadlänge, $w(i,j)$ das Gewicht, sowie $R_{i,j}$ den Index der Wurzel von $T_{i,j}$.*

Daraus lassen sich sofort die folgenden Zusammenhänge ableiten:

1. $p(i,i) = 0$ und $w(i,i) = \alpha_i$, denn der Baum $T_{i,i}$ besteht nur aus dem Blatt (i).

2. $w(i,j) = w(i,j-1) + \alpha_j$.

3. $p(i, R_{i,j}) + p(R_{i,j}+1, j) = \min_{i \leq k < j} (p(i,k) + p(k+1,j)) = p(i,j) - w(i,j)$.

Es gilt also: $p(i,j) = p(i, R_{i,j}) + p(R_{i,j}+1, j) + w(i,j)$.

Unter Benutzung dieser Tatsachen läßt sich ein einfacher Algorithmus zur Konstruktion eines optimalen Suchbaumes angeben. Es werden sukzessive optimale Suchbäume $T_{i,j}$ für Bereiche $[x_{i-1}; x_j]$ konstruiert, indem die Wurzel $R_{i,j}$ bestimmt wird, die eine optimale Aufteilung in T_l und T_r gewährleistet.

Algorithmus zur Konstruktion eines optimalen Suchbaumes

- $p(i,i) := 0$ und $w(i,i) := \alpha_i$ für $i = 1 \dots n$.
- Berechne $w(i,j)$ für alle $1 \le i < j \le n$.
- Für die Intervallängen $k = 1 \dots (n-1)$:
  Für alle Intervalle $[x_i; x_j]$ mit $j - i = k$

(*) Wähle $m \in \{i, \dots, j-1\}$ so, daß $p(i,m) + p(m+1,j)$ minimal wird und setze
$R_{i,j} := m$
$p(i,j) := p(i,m) + p(m+1,j) + w(i,j)$.

Die Komplexität dieses Algorithmus berechnet sich folgendermaßen: Für alle $T_{i,j}$ mit $1 \le i < j \le n$ wird die Wurzel gesucht, und das sind $\frac{1}{2}n(n-1) \in O(n^2)$ Stück. Um die Wurzel eines optimalen Suchbaumes $T_{i,j}$ bestimmen zu können, muß jeweils von $j - i$ möglichen Unterteilungen das Minimum bestimmt werden. Damit werden

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (j-i) \;\in O(n^3)$$

Schritte benötigt, und es gilt:

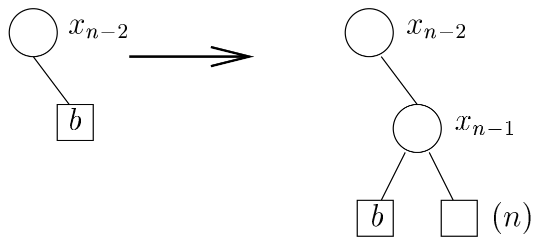
Satz 1.4 *Die dynamische Konstruktion eines optimalen Suchbaumes für einen gewichteten linear geordneten Suchbereich mit n Elementen läßt sich mit $O(n^2)$ Speicherplatz und in $O(n^3)$ Zeit realisieren.* □

1.7 Eine schnellere Konstruktion

Die Komplexität des betrachteten Algorithmus läßt sich verbessern, wenn die Wurzel eines Unterbaumes $T_{i,j}$ nicht unter allen $x_i, \dots, x_{j-1}$ gesucht werden muß, sondern der mögliche Bereich eingeschränkt werden kann. Die hier nachvollzogene Idee orientiert sich an einem Artikel von Knuth, der 1970 veröffentlicht wurde [11].

Zuerst muß die Struktur optimaler Suchbäume näher untersucht werden. Es hilft das folgende Lemma.

Lemma 1.5 *Es sei $\alpha_n = 0$. Dann entsteht ein optimaler Suchbaum $T_{1,n}$ für $[x_0; x_n]$, indem bei $T_{1,n-1}$ das rechte Blatt ersetzt wird durch*



Beweis.

- $w(i,n) = w(i,n-1) + \alpha_n = w(i,n-1)$.
- $p(n,n) = \alpha_n = 0$; $R_{n-1,n} = n-1$.
- $p(n-1,n) = 2\alpha_{n-1}$, denn wie gesehen ist

$$\begin{aligned} p(n-1,n) &= p(n-1,R_{n-1,n}) + p(R_{n-1,n}+1,n) + w(n-1,n) \\ &= p(n-1,n-1) + p(n,n) + w(n-1,n-1) \\ &= \alpha_{n-1} + \alpha_n + \alpha_{n-1} \\ &= 2\alpha_{n-1}, \end{aligned}$$

  da $\alpha_n = 0$ nach Voraussetzung.

- **Behauptung:** $p(i,n) = p(i,n-1) + \alpha_{n-1}$ und $R_{i,n} = R_{i,n-1}$.

  Beweis. Induktion über $(n-i)$:

  Es sei $n - i = 1$, also $i = n-1$. Dann ist

  $$p(n-1,n) = 2\alpha_{n-1} = p(n-1,n-1) + \alpha_{n-1} \text{ und } R_{n-1,n} = n-1 = R_{n-1,n-1}.$$

  Nach Induktionsvoraussetzung gelte:

$$\begin{aligned} p(n,n) &= 0 \\ p(n-1,n) &= p(n-1,n-1) + \alpha_{n-1} \\ p(n-2,n) &= p(n-2,n-1) + \alpha_{n-1} \\ &\vdots \\ p(r,n) &= p(r,n-1) + \alpha_{n-1} \\ &\vdots \\ p(i+1,n) &= p(i+1,n-1) + \alpha_{n-1} \quad , \end{aligned}$$

und damit folgt

$$\begin{aligned}
p(i,n-1)+p(n,n) &= p(i,n-1)+0 \\
p(i,n-2)+p(n-1,n) &= p(i,n-2)+p(n-1,n-1)+\alpha_{n-1} \\
&\vdots \\
p(i,r-1)+p(r,n) &= p(i,r-1)+p(r,n-1)+\alpha_{n-1} \\
&\vdots \\
p(i,i)+p(i+1,n) &= p(i,i)+p(i+1,n-1)+\alpha_{n-1}.
\end{aligned}$$

Es bezeichne $r := R_{i,n-1}$.
Sicher ist dann, daß $p(i,r) + p(r+1, n-1) + \alpha_{n-1}$ die rechte Seite des obigen Systems minimiert, denn α_{n-1} kommt in all diesen Summen vor, und $R_{i,n-1}$ wird ja genau so gewählt. Damit wird durch dieses r auch die linke Seite des Systems minimiert. Dort stehen aber genau die Ausdrücke, die bei der Bestimmung von $R_{i,n}$ minimiert werden. Also ist $R_{i,n} = r = R_{i,n-1}$, das heißt beim Einfügen von (n) mit $\alpha_n = 0$ ändert sich die Wurzel des Suchbaumes und jedes Unterbaumes nicht. Damit ist auch klar, daß sich die gewichtete Pfadlänge nur um α_{n-1} erhöht, denn nur $(n-1)$ „rutscht" um ein Niveau tiefer.
Also ist $p(i,n) = p(i,n-1) + \alpha_{n-1}$. □

Wenn also ein Suchbereich $[x_0; x_k]$ *nach rechts* vergrößert wird um ein Intervall $]x_k, x_{k+1}[$ und das neue Blatt Wahrscheinlichkeit Null hat, so hat der optimale Suchbaum für den vergrößerten Suchbereich $T_{1,k+1}$ dieselbe Wurzel wie der optimale Suchbaum des ursprünglichen Suchbereichs $T_{1,k}$.

Im nächsten Schritt wird gezeigt, daß es für eine echt positive Wahrscheinlichkeit des neuen Intervalls immer einen optimalen Suchbaum $T_{1,k+1}$ gibt, dessen Wurzel nicht kleiner als die Wurzel von $T_{1,k}$ ist.

Satz 1.6 *Ist $T_{1,k}$ ein optimaler Suchbaum für $[x_0; x_k]$ und wird das Intervall $]x_k; x_{k+1}[$ hinzugefügt, dann gibt es einen optimalen Suchbaum $T_{1,k+1}$, so daß für die Wurzeln gilt: $R_{1,k} \leq R_{1,k+1}$.*

Bemerkung. Diese Aussage ist links-rechts-symmetrisch, also gleichbedeutend zu $R_{i,n} \geq R_{i-1,n}$ für den Fall, daß der Suchbereich nach links um $]x_{i-1}; x_i[$ erweitert wird.

Beweis. Induktion über k.
Für $k = 1$ ist die Behauptung klar.
Falls $\alpha_{k+1} = 0$ ist für das neue Intervall $(k+1)$, so liefert das vorstehende

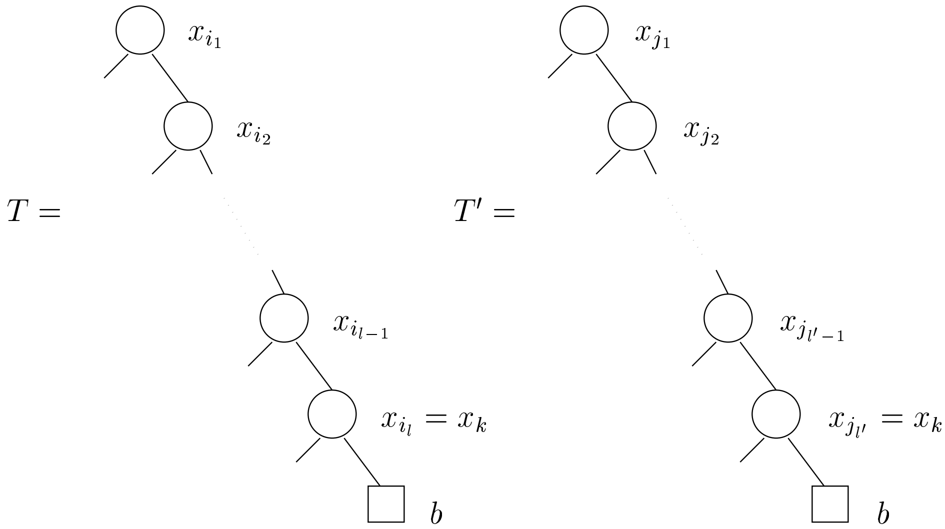


Abbildung 7: Die Suchbäume T und T' zum Beweis von Satz 1.6.

Lemma die Behauptung. Zu zeigen ist nun, daß für wachsendes α_{k+1} auch die Wurzel eines geeigneten optimalen Suchbaumes höchstens größer wird.

Es sei α ein Wert, so daß T ein optimaler Suchbaum ist, falls $\alpha_{k+1} = \alpha - \varepsilon$, und T' ein optimaler Suchbaum, falls $\alpha_{k+1} = \alpha + \varepsilon$, mit $T \neq T'$ (ε beliebig klein). T und T' sind in Abbildung 7 gezeigt. Falls $R_T \leq R_{T'}$ ist, ist die Behauptung bewiesen.

Sei nun $R_T > R_{T'}$.

> **Behauptung:** Dann gibt es einen Suchbaum T'' für $\alpha_{k+1} = \alpha + \varepsilon$, so daß $p(T'') = p(T')$ und $R_{T''} = R_T$.
> **Beweis.** Es ist $p(T) = l_T(1)\alpha_1 + \ldots + l_T(k+1)\alpha_{k+1}$ und $p(T') = l_{T'}(1)\alpha_1 + \ldots + l_{T'}(k+1)\alpha_{k+1}$. Für $\alpha_{k+1} = \alpha$ sind beide Bäume gleichgut, also $p(T) = p(T')$. Für $\alpha_{k+1} > \alpha$ ist T' der bessere Baum, und daher folgt $l_{T'}(k+1) < l_T(k+1)$. Das heißt aber auch, daß x_k in T' auf niedrigerem Niveau als in T steht.
>
> Es sei $l := l_T(x_k)$ und $l' := l_{T'}(x_k)$, dann ist $l' < l$ und $i_l = j_{l'} = k$.
>
> T_r ist ein Suchbaum für den Bereich $[x_{i_1}; x_{k+1}]$, T'_r für den Bereich $[x_{j_1}; x_{k+1}]$. Da jeder Unterbaum eines optimalen Suchbaumes wieder optimal ist, sind T_r und T'_r beide für ihren Suchbereich optimal. Nach der Annahme ist $j_1 < i_1$, also der Suchbereich von T'_r nach links größer als der von T_r. Beide Bäume sind kleiner als die ursprünglich betrachteten, daher kann auf sie die Induktionsvor-

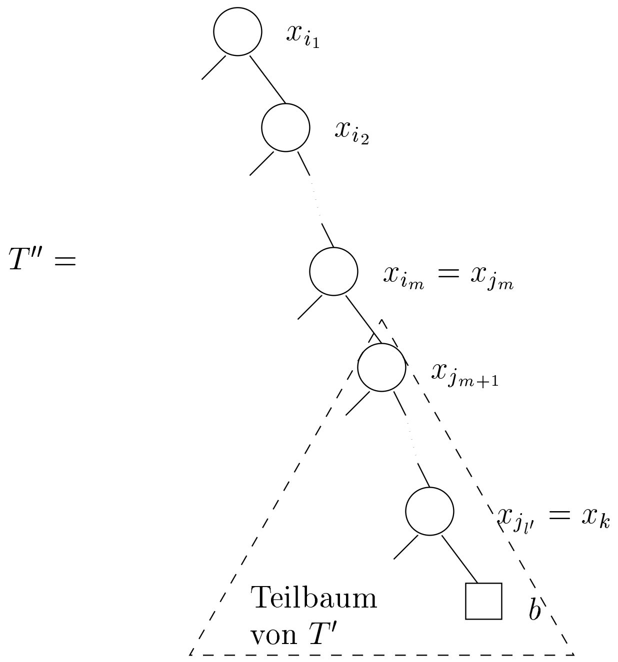


Abbildung 8: Der neue Suchbaum T''.

aussetzung (in der symmetrischen Fassung) angewandt werden, und für ihre Wurzeln folgt $j_2 \leq i_2$.

Falls $j_2 < i_2$ läßt sich ebenso $j_3 \leq i_3$ folgern usw. Da $l' < l$ und $i_l = k = j_{l'}$ muß es einen Index m geben, so daß $i_m = j_m$ ist. Es sei jetzt T'' der Baum, der entsteht, wenn in T der rechte Unterbaum, der in x_{i_m} verwurzelt ist, durch den entsprechenden Unterbaum von T' ersetzt wird,vgl. Abbildung 8. Dann ist T'' wieder ein Suchbaum, der die gleiche Wurzel wie T hat und für die gewichtete Pfadlänge gilt: $p(T'') = p(T')$ für jeden Wert von α_{k+1}.

Damit ist gezeigt, daß für wachsendes α_{k+1} die Wurzel eines geeigneten Suchbaumes höchstens größer wird, und der Satz ist bewiesen. □

Anwenden des Satzes auf alle Unterbäume und Ausnutzen der links-rechts-Symmetrie liefert das Korollar, mit dessen Hilfe die Laufzeit des Konstruktionsalgorithmus entscheidend verbessert werden kann.

Korollar 1.7 *Es gibt immer optimale Suchbäume, für deren Wurzeln* $R_{i,j-1} \leq R_{i,j}$ *und* $R_{i,j} \leq R_{i+1,j}$ *gilt,* $1 \leq i < j \leq n$. □

Damit kann im Konstruktionsalgorithmus die Zeile (*) ersetzt werden durch:
Wähle $m \in \{R_{i,j-1}, \ldots, R_{i+1,j}\}$ so, daß $p(i,m) + p(m+1,j)$ minimal wird.

Wird bei der Berechnung der $w_{i,j}$ auf schon bekannte Teilsummen zurückgegriffen, ergibt sich für die Gesamtkomplexität durch Summation über die Differenzen $j - i$:

$$\begin{aligned} & \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} R_{i+1,j} - R_{i,j-1} + 1 = \sum_{d=1}^{n-1} \sum_{i=1}^{n-d} R_{i+1,i+d} - R_{i,i+d-1} + 1 \\ = \; & \sum_{d=1}^{n-1} (R_{n-d+1,n} - R_{1,d} + n - d) \leq \sum_{d=1}^{n-1} (2n - 2 - d) \; \in O(n^2). \end{aligned}$$

Das führt zu dem Ergebnis:

Satz 1.8 *Für einen linear geordneten Suchbereich mit einer diskreten Wahrscheinlichkeitsverteilung kann ein optimaler Suchbaum in $O(n^2)$ konstruiert werden.* □

Bemerkung. Für den hier untersuchten Fall, daß die Intervallränder Wahrscheinlichkeit Null tragen, gibt es Algorithmen, die einen optimalen Baum in $O(n \log n)$ konstruieren. Erwähnt seien dabei die Arbeiten von Garsia und Wachs [5] sowie von Hu und Tucker [8]. Für beide Konstruktionen wird eine lineare Ordnung auf den Blättern vorausgesetzt.

2 Optimale Suchbäume für Gitter

In diesem Kapitel wird das Verfahren der eindimensionalen Suche exemplarisch verallgemeinert: Gegeben ist die durch ein Gitter in Zellen aufgeteilte Ebene, und zu einem gegebenen Punkt (x, y) soll die Zelle, in der er liegt, bestimmt werden.

Ein *orthogonales Gitter* (kurz: *Gitter*) im kartesischen Koordinatensystem ist durch senkrechte Unterteilungen in x- und y-Richtung gegeben: $S_x := \{x_0 < \ldots < x_n\}$ und $S_y := \{y_0 < \ldots < y_m\}$. Die *Spalten* s_i sind gegeben durch Intervalle der x-Unterteilung $]x_{i-1}; x_i[$ und die *Zeilen* z_k durch Intervalle der y-Unterteilung $]y_{k-1}; y_k[$. Eine *Zelle* (i, k) des Gitters ist ein Rechteck $s_i \times z_k$. Die Wahrscheinlichkeit einer Zelle (i, k) wird mit α_{ik} bezeichnet, und es sei

$$\sum_{i=1}^{n} \sum_{k=1}^{m} \alpha_{ik} = 1.$$

Ein Beispiel für ein Gitter mit Unterteilungen ist in Abbildung 9 zu sehen. Die Wahrscheinlichkeit, daß ein Punkt auf einer der unterteilenden Geraden liegt, sei wieder Null. Um zu einem Punkt (x, y) die Zelle, in der er liegt, zu bestimmen, sind die Abfragen $x > x_i$ oder $y > y_k$ möglich.

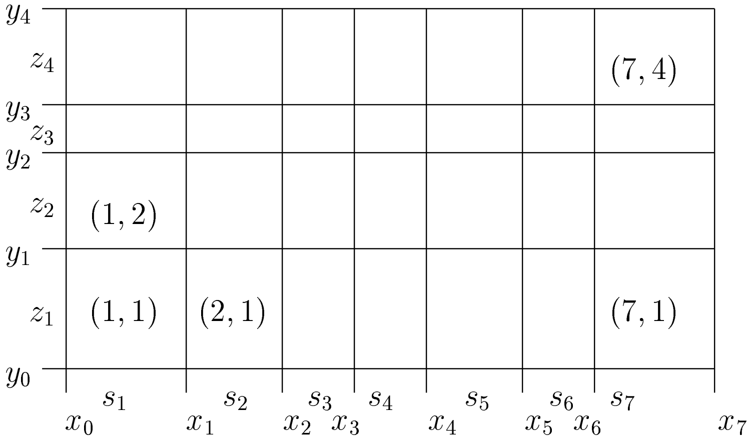


Abbildung 9: Gitter mit Unterteilungen.

2.1 Die Extremfälle

Es werden zunächst mögliche „extreme" Wahrscheinlichkeitsverteilungen auf dem Gitter untersucht, um daraus eine Vermutung über obere und untere Schranken für $p(T)$ zu gewinnen.

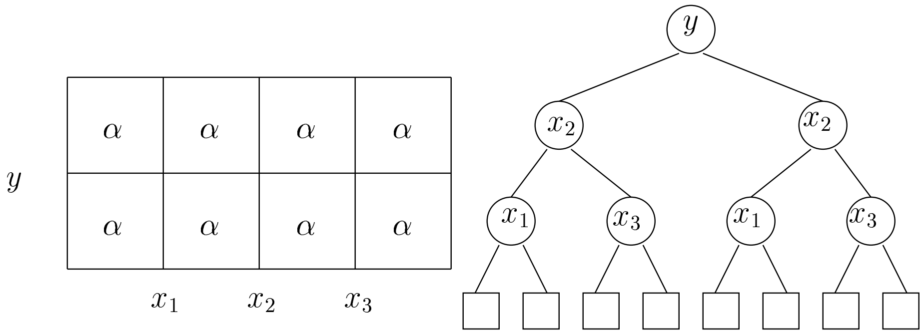


Abbildung 10: Gitter mit Gleichverteilung und ein zugehöriger optimaler Suchbaum.

- Klar ist, daß bei gleicher Wahrscheinlichkeit α für alle Zellen ein balancierter Suchbaum optimal ist, siehe Abbildung 10. Falls die Anzahl $m \cdot n$ der Zellen eine Zweierpotenz ist, folgt

$$p(T) = \log_2 mn = -\sum_{i=1}^{n}\sum_{k=1}^{m} \frac{1}{mn} \log_2 \left(\frac{1}{mn}\right) = H\left(\frac{1}{mn}, \dots, \frac{1}{mn}\right).$$

- Es wird nun der Fall betrachtet, daß eine nicht am Rand liegende Zelle fast die ganze Wahrscheinlichkeit $(1-\varepsilon)$, ε sehr klein, trägt. Alle anderen Zellen haben echt positive, aber verschwindend geringe Wahrscheinlichkeiten α, die der Einfachheit halber gleich groß seien, also $\alpha = \frac{\varepsilon}{mn-1}$. Dieser Fall ist in Abbildung 11 skizziert.

  Um die innere Zelle zu lokalisieren, sind mindestens 4 Abfragen nötig, und für einen optimalen Baum sollten es genau 4 Abfragen sein. Die anderen Zellen werden so abgefragt, daß sich ihr Niveau im Suchbaum um höchstens 1 unterscheidet.

  Es gilt damit $p(T) \longrightarrow 4$ für $\varepsilon \longrightarrow 0$. Weiterhin ist $\lim_{x\to 0} x \log_2 x = \lim_{x\to 1} x \log_2 x = 0$. So ergibt sich $p(T) \longrightarrow H + 4$ für $\varepsilon \longrightarrow 0$, wenn H die Entropie für diese Wahrscheinlichkeitsverteilung ist.

Diese Betrachtungen legen den Schluß nahe, daß für einen optimalen Suchbaum T für ein Gitter mit Wahrscheinlichkeiten α_{ik} gilt

$$H((\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}}) \le p(T) \le H((\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}}) + 4.$$

Später wird sich herausstellen, daß diese Vermutung richtig ist.

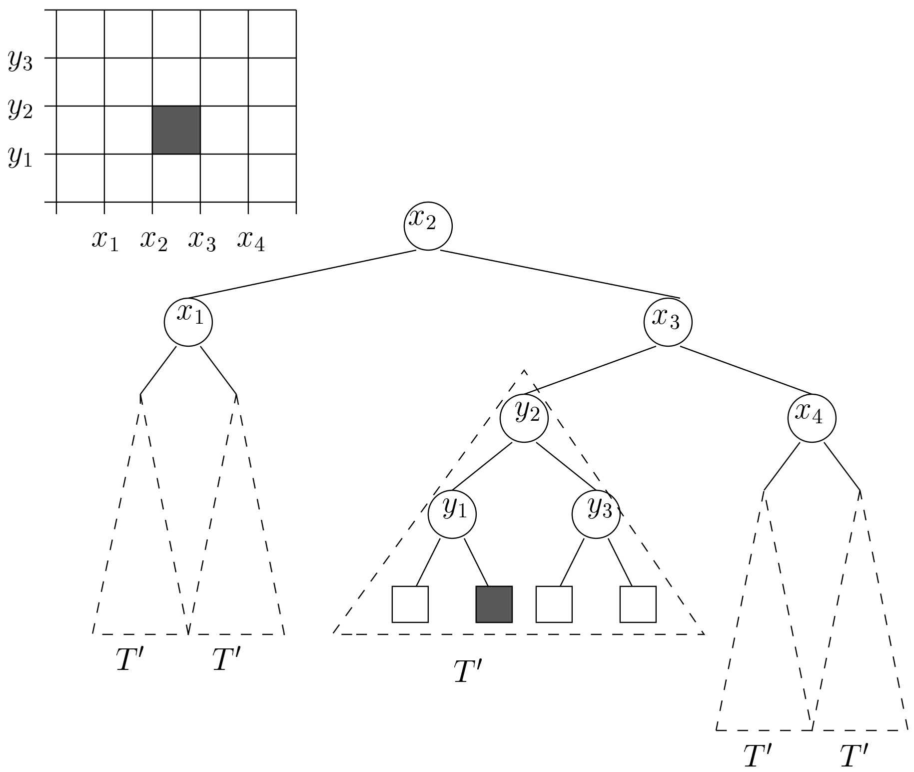


Abbildung 11: Die schraffierte Zelle trägt fast das ganze, die umliegenden verschwindend kleines Gewicht.

2.2 Ein erster Versuch

Die erste Idee, wie zu einem gegebenen gewichteten Gitter ein optimaler Suchbaum konstruiert werden kann, ist die Unterteilung der Suche in x- und y-Richtung. Konkret bedeutet das: Zuerst wird ein optimaler Suchbaum für die Spalten konstruiert, wobei jeder Spalte die Summe der Wahrscheinlichkeiten ihrer Rechtecke zugewiesen wird. An eine feste Spalte wird ein optimaler Suchbaum für die Zeilen in dieser Spalte angehängt.

Es wird gezeigt, daß dieses Verfahren zwar eine recht gute, aber keine optimale Lösung liefert.

Die Wahrscheinlichkeit der Spalte s_i sei $p_i := \sum_{k=1}^{m} \alpha_{ik}$. Wegen $\sum_{i=1}^{n} p_i = 1$ ist das eine Wahrscheinlichkeitsverteilung. An ein Blatt s_i des optimalen Spaltenbaumes T_x für die Verteilung $p_1, \ldots, p_n$ wird ein Baum $T_y(i)$ angehängt, der optimal für die Verteilung $q_1(i), \ldots, q_m(i)$ ist, wobei $q_k(i) := \frac{\alpha_{ik}}{p_i}$ ist. Auch das liefert für jede feste Spalte eine Wahrscheinlichkeitsverteilung.

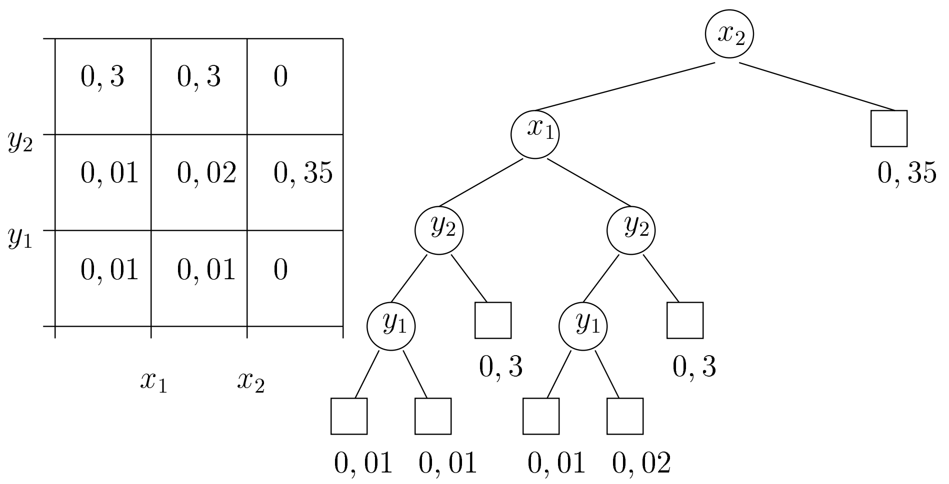


Abbildung 12: Gitter mit dem Baum, der durch das beschriebene Verfahren erzeugt wird mit $p(T) = 2,35$.

Ein Beispiel für einen so konstruierten, jedoch nicht optimalen Baum, ist in Abbildung 12 zu sehen, Abbildung 13 zeigt einen zugehörigen optimalen Suchbaum.

Es sei $l_{T_x}(i)$ das Niveau des Blattes s_i in T_x und $l_{T_y(i)}(k)$ das Niveau des Blattes (i, k) in $T_y(i)$.

Damit wird ein Punkt (x, y) im Gitter folgendermaßen lokalisiert:

- Lokalisiere x in T_x, Ergebnis: $x \in s_i$.
- Lokalisiere y in $T_y(i)$, Ergebnis: $y \in z_k(i)$.
- Insgesamt: $(x, y) \in (i, k)$.

Die Suche nach (x, y) braucht somit $l_{T_x}(i) + l_{T_y(i)}(k)$ Abfragen.

Satz 2.1 *Für einen so konstruierten Suchbaum T zu einer Wahrscheinlichkeitsverteilung $(\alpha_{ik})_{\substack{i=1\dots n\\ k=1\dots m}}$ gilt:*

$$p(T) \leq H((\alpha_{ik})_{\substack{i=1\dots n\\ k=1\dots m}}) + 4.$$

Beweis. Mit den Bezeichnungen von oben gilt

$$p(T) = \sum_{i=1}^{n}\sum_{k=1}^{m} l_T(i,k)\alpha_{ik} = \sum_{i=1}^{n}\sum_{k=1}^{m} \Big(l_{T_x}(i) + l_{T_y(i)}(k)\Big)\,\alpha_{ik} \quad ,$$

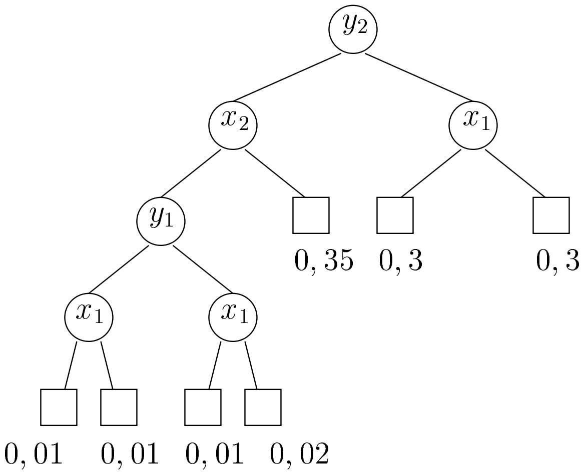


Abbildung 13: Optimaler Suchbaum für das Gitter aus Abbildung 12 mit $p(T) = 2,10$.

und da $\alpha_{ik} = q_k(i)\, p_i$ ist, ergibt sich durch Summation

$$= p(T_x) + \sum_{i=1}^{n} p_i\, p(T_y(i)).$$

Mit $H_x := H(p_1, \ldots, p_n)$ und $H_y(i) := H(q_1(i), \ldots, q_m(i))$ läßt sich das, indem Satz 1.3 auf T_x und $T_y(i)$ angewendet wird, von oben abschätzen durch

$$H_x + 2 + \sum_{i=1}^{n} p_i (H_y(i) + 2) \quad ,$$

und Einsetzen für H_x, $H_y(i)$ und $q_k(i)$ sowie Zusammenfassen liefert

$$= -\sum_{i=1}^{n} p_i \left(\log_2 p_i + \sum_{k=1}^{m} \frac{\alpha_{ik}}{p_i} \log_2 \frac{\alpha_{ik}}{p_i} \right) + 4 = -\sum_{i=1}^{n} p_i \left(\sum_{k=1}^{m} \frac{\alpha_{ik}}{p_i} \log_2 \alpha_{ik} \right) + 4$$

$$= -\sum_{i=1}^{n} \sum_{k=1}^{m} \alpha_{ik} \log_2 \alpha_{ik} + 4 \quad ,$$

womit der Satz bewiesen ist. □

Es werden nun die Kosten für den Aufbau dieser Struktur untersucht:

- Die Berechnung von p_i, $q_k(i)$ ist in $O(mn)$.
- Die Konstruktion von T_x braucht $O(n^2)$, die von $T_y(i)$ $O(m^2)$ Zeit.
- Damit ist die Komplexität insgesamt $O(mn+n^2+nm^2) = O(n^2+nm^2)$.

2.3 Die Pfadlänge eines optimalen Suchbaumes für ein Gitter

In diesem Kapitel werden für den optimalen Suchbaum eines Gitters mit gegebenen Gewichten eine untere- und obere Schranke der gewichteten Pfadlänge bewiesen.

Lemma 2.2 *Für jeden Suchbaum T eines Gitter mit Gewichten $(\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}}$ ist $p(T) \geq H((\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}})$.*

Beweis. Der Beweis des Lemmas verläuft analog zum Beweis der unteren Schranke aus Satz 1.3 mit dem einzigen Unterschied, daß jetzt über alle Paare (i, k) summiert wird. □

Wie in Kapitel 2.1 gesehen, wird diese Schranke für die Gleichverteilung angenommen, die Ungleichung ist also scharf.

Es ist nun gezeigt, daß für *jeden* Suchbaum T $p(T) \geq H((\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}})$ ist und daß es immer einen Suchbaum T gibt, für den $p(T) \leq H((\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}}) + 4$ ist. Damit gilt für einen *optimalen* Suchbaum T_0 :

Satz 2.3

$$H((\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}}) \leq p(T_0) \leq H((\alpha_{ik})_{\substack{i=1\dots n\\k=1\dots m}}) + 4.$$

□

Bemerkung. $p(T)$ ist nach oben auch durch $\lceil \log_2(\#\ \text{Zellen}) \rceil$ beschränkt. In manchen Fällen ist diese Schranke genauer, meistens jedoch, da sie nicht von den Gewichten abhängt, zu allgemein.

Satz 2.3 liefert zwar ein schönes Ergebnis, jedoch hängen sowohl die hier gezeigten Schranken als auch die für den eindimensionalen Fall nur von den Gewichten α ab, nicht jedoch von der Stelle ihres Auftretens im Gitter bzw. im Intervall. Wie in Abbildung 15 zu sehen ist, können bei unterschiedlicher Anordnung der gleichen Gewichte die Pfadlängen der zugehörigen optimalen Suchbäume sehr verschieden sein. Die durch ungünstige Anordnung der Gewichte verursachte Abweichung der Pfadlänge von der unteren Schranke kann für Gitter stärker ausfallen als im eindimensionalen Fall.

Wenn es gelingt, auf sinnvolle Weise die Geometrie in die Abschätzungen mit einzubeziehen, sollte es möglich sein, genauere Schranken anzugeben.

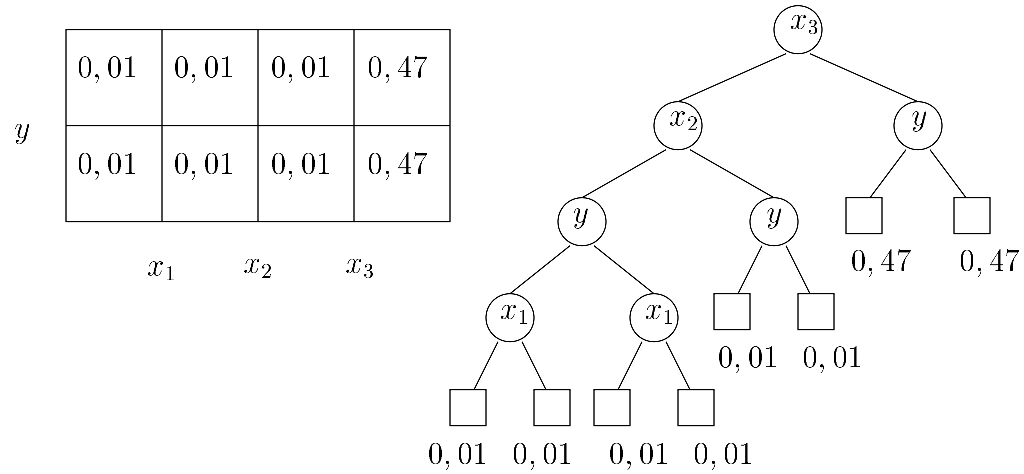


Abbildung 14: Gitter mit optimalem Suchbaum T_1.

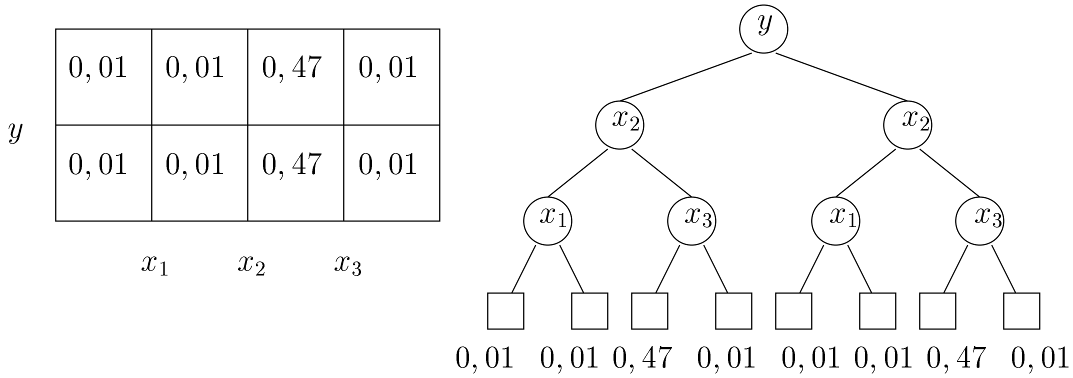


Abbildung 15: Gitter, bei dem die gleichen Gewichte anders angeordnet sind, mit optimalem Suchbaum T_2. Es ist $p(T_1) = 2,10$ und $p(T_2) = 3,00$. Satz 2.3 liefert für diese Verteilung $1,42 \leq p(T_0) \leq 5,42$.

2.4 Die Konstruktion eines optimalen Suchbaumes für ein Gitter

Wie im eindimensionalen Fall wird ein optimaler Suchbaum dynamisch konstruiert. Dazu muß ein optimaler Suchbaum für jedes Teilrechteck des Gitters konstruiert werden, indem für ein größeres Rechteck wieder die Wurzel eines Suchbaumes bestimmt wird, so daß die Summe der gewichteten Pfadlängen des rechten und linken Unterbaumes minimal ist. Es ist jetzt zu beachten, daß ein Vergleich den Suchbereich *entweder* in x- *oder* in y-Richtung einschränkt, also die entsprechenden Teilrechtecke weniger hoch oder weniger breit als das ursprüngliche sind.

Zur Beschreibung eines Teilrechteckes sind leider vier Indizes notwendig: für den rechten, linken, oberen und unteren Rand.

Definition 2.1 *Es sei* $[i,j,k,l]$ *das Rechteck* $]x_{i-1}; x_j[\times]y_{k-1}; y_l[$ *und* $p(i,j,k,l)$ *die gewichtete Pfadlänge,* $w(i,j,k,l)$ *das Gewicht und* $R_{i,j,k,l}$ *die Wurzel des optimalen Suchbaumes* $T_{i,j,k,l}$ *für dieses Rechteck.*

Bemerkung.

1. Für Teilrechtecke, die nur aus einer Zelle (i,k) bestehen, ist wie im eindimensionalen Fall $p(i,i,k,k) = 0$ und $w(i,i,k,k) = \alpha_{ik}$.

2. Für das Gewicht eines Teilrechtecks ist
$$w(i,j,k,l) = w(i,j-1,k,l) + \sum_{h=k}^{l} \alpha_{jh} = w(i,j,k,l-1) + \sum_{b=i}^{j} \alpha_{bl}.$$

3. Falls die Wurzel von $T_{i,j,k,l}$ eine x-Unterteilung ist, gilt
$$\begin{aligned} p(i,j,k,l) &= \min_{i \le b < j} (\ p(i,b,k,l) + p(b+1,j,k,l)\ ) + w(i,j,k,l) \\ &= p(i, R_{i,j,k,l}, k, l) + p(R_{i,j,k,l}+1, j, k, l) + w(i,j,k,l) \end{aligned}$$
und andernfalls
$$\begin{aligned} p(i,j,k,l) &= \min_{k \le h < l} (\ p(i,j,k,h) + p(i,j,h+1,l)\ ) + w(i,j,k,l) \\ &= p(i,j,k,R_{i,j,k,l}) + p(i,j,R_{i,j,k,l}+1,l) + w(i,j,k,l). \end{aligned}$$

Damit läßt sich sofort der Konstruktionsalgorithmus angeben:

- $p(i,i,k,k) := 0$ und $w(i,i,k,k) := \alpha_{ik}$ für $i = 1 \ldots n,\ k = 1 \ldots m$.
- Berechne $w(i,j,k,l)$ für alle $1 \le i < j \le n,\ 1 \le k < l \le m$.

- Für jedes Rechteck mit Breite $b = 1$ bis n
  Für jedes Rechteck mit Höhe $h = 1$ bis m
  Für jede Startspalte $i = 1$ bis $n - b$
  Endspalte $j := i + b$.
  Für jede Startzeile $k = 1$ bis $m - h$
  Endzeile $l := k + h$.

  - Wähle $m(x) \in \{i, \ldots, j-1\}$ so, daß
    $p(x) := p(i, m(x), k, l) + p(m(x) + 1, j, k, l)$ minimal wird.
  - Wähle $m(y) \in \{k, \ldots, l-1\}$ so, daß
    $p(y) := p(i, j, k, m(y)) + p(i, j, m(y) + 1, l)$ minimal wird .
  - Falls $p(x) \leq p(y)$
    $status(i, j, k, l) := x$ und $R_{i,j,k,l} := m(x)$.
    $p(i, j, k, l) := p(i, m(x), k, l) + p(m(x)+1, j, k, l)) + w(i, j, k, l)$.

    Andernfalls
    $status(i, j, k, l) := y$ und $R_{i,j,k,l} := m(y)$.
    $p(i, j, k, l) := p(i, j, k, m(y)) + p(i, j, m(y)+1, l) + w(i, j, k, l)$.

Die Komplexität des Algorithmus ist bestimmt durch den Aufwand für den dritten Schritt, bei dem für jedes Rechteck der Breite b und Höhe h aus $b + h$ möglichen Unterteilungen die beste berechnet werden muß. Es gibt $(n - b)(m - h)$ Rechtecke mit Breite b und Höhe h, also insgesamt

$$\sum_{b=1}^{n} \sum_{h=1}^{m} (n - b)(m - h) \quad \in O(n^2 m^2)$$

Stück. Da für jedes dieser Rechtecke aus höchstens $m + n$ Unterteilungen die beste gesucht werden muß, ergibt sich eine Konstruktionszeit von $O(n^2 m^2 (n + m))$. Damit läßt sich der folgende Satz formulieren.

Satz 2.4 *Für ein gewichtetes $m \times n$-Gitter kann in $O(n^2 m^2 (n + m))$ Zeit mit $O(n^2 m^2)$ Speicherbedarf ein optimaler Suchbaum dynamisch konstruiert werden.* □

2.5 Verallgemeinerung des Gitters

Bei den bisher angestellten Untersuchungen wurde davon ausgegangen, daß das Gitter durch Geraden, die senkrecht zu den Koordinatenachsen verlaufen, gegeben ist. Diese Annahme ist zwar nützlich insofern, als die Darstellung des Problems dadurch etwas übersichtlicher wird, aber nicht notwendig. Tatsächlich reicht es, wenn ein System von zwei gerichteten Parallelenscharen g_i und h_k , $i = 1 \ldots n$; $k = 1 \ldots m$ gegeben ist.

Die Zellen (i,k) dieser Struktur sind Parallelogramme, und $\hat{T}_{i,j,k,l}$ ist hier ein optimaler Suchbaum für den Bereich zwischen $g_{i-1},\ g_j$ und $h_{k-1},\ h_l$. Zum Lokalisieren eines Punktes (x,y) werden die Vergleiche $x \le x_i,\ y \le y_j$ ersetzt durch Fragen *Ist* (x,y) *links von* g_i*?* und *Ist* (x,y) *links von* h_j*?*

Die Geraden sollen so gerichtet sein, daß g_{i-1} links von g_i bzw. h_{j-1} links von h_j steht. Eine Gerade g sei durch zwei ihrer Punkte, beispielsweise $P_g = (0, g(0))$ und $Q_g = (1, g(1))$ gegeben.

Lemma 2.5 *Bei gleichen Wahrscheinlichkeitsverteilungen braucht das Suchen eines Punktes* (x,y) *in einem optimalen Suchbaum* $\hat{T}$ *für zwei Parallelenscharen mit* n *bzw.* m *gerichteten Geraden dieselbe Anzahl von Vergleichen wie in einem optimalen Suchbaum* T *für ein orthogonales* $n \times m$ *Gitter.*

Beweis. Bei gleicher Wahrscheinlichkeitsverteilung haben T und $\hat{T}$ dieselbe Struktur. Es ist also nur zu zeigen, daß die Frage *Ist* (x,y) *links von* g_i*?* sich durch *einen* Vergleich umsetzen läßt.

Das ist kein Problem, denn es gilt allgemein für einen Punkt $R = (x,y)$ und eine Gerade g durch die zwei Punkte $P = (x_1, y_1)$ und $Q = (x_2, y_2)$:

$$\begin{vmatrix} x & y & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix} \begin{cases} > 0 & \Leftrightarrow \quad R \text{ links von } g \\ = 0 & \Leftrightarrow \quad R \text{ liegt auf } g \\ < 0 & \Leftrightarrow \quad R \text{ rechts von } g \end{cases}$$

Ist g durch die speziellen Punkte P_g und Q_g von oben gegeben, gilt damit

$$\begin{vmatrix} x & y & 1 \\ 0 & g(0) & 1 \\ 1 & g(1) & 1 \end{vmatrix} = x(g(0) - g(1)) + y - g(0) \begin{cases} > 0 & \Leftrightarrow \quad R \text{ links von } g \\ = 0 & \Leftrightarrow \quad R \text{ liegt auf } g \\ < 0 & \Leftrightarrow \quad R \text{ rechts von } g \end{cases} .$$

□

3 Ein Suchbaum für gewichtete Regionen in einem Arrangement

Das Ziel dieses Kapitels ist es, einen Punkt P in einem gewichteten Arrangement zu lokalisieren. Dabei ist ein Arrangement eine Menge von gerichteten Geraden in der Ebene, eine Region eines Arrangements ist der Schnitt von Halbebenen, die jeweils durch Geraden begrenzt sind. Zu einem Punkt P soll die Region, in der er liegt, bestimmt werden.

Der Fall, daß die Regionen nicht gewichtet sind, wurde seit den siebziger Jahren ausführlich untersucht. Es sind Konstruktionsalgorithmen für verschiedene Datenstrukturen entwickelt worden, die das Lokalisieren eines Punktes in einer Unterteilung der Ebene ermöglichen. Eines dieser Verfahren für den ungewichteten Fall wird in Kapitel 3.1 vorgestellt. Dann wird erläutert, warum dieses sich nicht einfach auf den gewichteten Fall übertragen läßt.

Anschließend wird eine Möglichkeit vorgestellt, wie ein Punkt möglichst schnell gefunden werden kann, wenn die Regionen Gewichte tragen und die Wahrscheinlichkeit, daß ein Punkt auf einer der Geraden liegt, Null ist. Dazu stehen nur die Fragen *Ist P links oder rechts von der Geraden g?* zur Verfügung.

In einem Arrangement von n Geraden läßt sich jede Region durch einen Vektor $(v_1, \ldots, v_n)$ charakterisieren, wobei $v_i = -$, falls die Region links der Geraden mit der Nummer i liegt, und $v_i = +$ sonst.

Für den Fall, daß das Arrangement aus mehreren Scharen paralleler Geraden besteht, kann eine Region einfacher beschrieben werden. Es ist dann ausreichend, aus jeder Schar die erste Gerade anzugeben, die rechts von der Region liegt, um sie vollständig zu charakterisieren.

Im ersten Schritt wird der Zusammenhang zwischen einem aus d Parallelenscharen bestehenden Arrangement und einem d-dimensionalen Gitter hergestellt, um dann optimale Suchbäume für ein gewichtetes d-dimensionales Gitter näher zu untersuchen. Im folgenden wird geprüft, ob die gewonnenen Ergebnisse schöne Folgerungen für die ursprüngliche Fragestellung zulassen.

3.1 Der ungewichtete Fall

Die Suche nach einem Punkt in der Ebene ist für beliebige geradlinige Unterteilungen $\mathcal{S}$ möglich, Unterteilungen durch Arrangements sind ein Spezialfall davon. Zuerst wird ein Verfahren vorgestellt, mit dem ein Punkt in einer beliebigen geradlinigen Unterteilung der Ebene lokalisiert werden kann. Dabei sind nicht nur Vergleiche zwischen dem Punkt und den Geraden, sondern auch zwischen dem Punkt und bestimmten vertikalen Linien erlaubt. Die Darstellung orientiert sich an [3]. Anschließend wird erklärt was sich daraus für den Fall ergibt, daß die Unterteilung durch ein Arrangement gegeben ist. Schließlich wird noch erläutert, warum dieses Verfahren nicht hilfreich ist, um eine Suche mit gewichteten Regionen zu realisieren.

Gegeben sei eine geradlinige Unterteilung der Ebene $\mathcal{S}$ mit n Geradenstücken, bei der keine zwei Endpunkte von Geradenstücken direkt übereinander liegen sollen. Es folgt, daß keine Region durch eine vertikale Linie begrenzt werden kann.

$\mathcal{S}$ wird folgendermaßen erweitert: Ausgehend von jedem Schnittpunkt werden vertikale Linien nach oben und unten eingefügt, die bis zur ersten Kante von $\mathcal{S}$, die sie treffen, verlaufen. Wenn keine Kante getroffen wird, entsteht ein vertikaler Strahl. Diese Erweiterung wird mit $\mathcal{E}$ bezeichnet. Dadurch entsteht eine neue Unterteilung $\mathcal{S}^* = \mathcal{S} \cup \mathcal{E}$, in der alle Regionen Trapeze oder Dreiecke sind. Die Verfeinerung $\mathcal{S}^*$ von $\mathcal{S}$ heißt Trapezierung. Jede Region R ist dann durch höchstens 4 Angaben eindeutig bestimmt: Die obere und untere begrenzende Linie $top(R)$ und $bottom(R)$ sowie die Punkte $left(R)$ und $right(R)$, die die linke und rechte Begrenzung verursachen.

Der Speicheraufwand für $\mathcal{S}^*$ ist linear in n, denn jedes Geradenstück in $\mathcal{S}$ verursacht 4 Kanten in der Erweiterung $\mathcal{E}$ (für jeden Endpunkt 2 Kanten). Damit ist die Anzahl der Endpunkte der Geradenstücke in $\mathcal{S}^*$ linear in n, ebenso die Anzahl der Trapeze.

Mit der folgenden Datenstruktur, einem gerichteten azyklischen Graphen mit ausgezeichneter Wurzel, ist es möglich, einen Punkt P in $\mathcal{S}^*$ zu lokalisieren:

Die Suchstruktur hat 2 Sorten von inneren Knoten; jeder innere Knoten hat zwei Nachfolger. Für jeden Endpunkt eines Geradenstückes aus $\mathcal{S}$ gibt es einen x-Knoten und für jedes Geradenstück einen y-Knoten. Für jedes Trapez in $\mathcal{S}^*$ gibt es genau ein Blatt.

In den x-Knoten kann getestet werden, ob P rechts oder links von der vertikalen Linie durch den entsprechenden Punkt liegt. In den y-Knoten ist es möglich zu entscheiden, ob P über oder unter dem entsprechenden Geradenstück liegt.

Die Suche beginnt in der Wurzel, geht entlang eines gerichteten Weges und endet in einem Blatt. Je nach dem Ergebnis des Tests in einem inneren Knoten geht die Suche in einem der beiden Nachfolger weiter. Dabei soll gesichert sein, daß ein y-Knoten nur dann erreicht wird, wenn P tatsächlich über oder unter dem entsprechenden Geradenstück liegt. Das Ergebnis der Suche ist das Trapez, in dem P liegt. Wenn jedes Trapez einen Zeiger hat auf die Region von $\mathcal{S}$, zu der es gehört, ist P damit auch in $\mathcal{S}$ lokalisiert. Ein

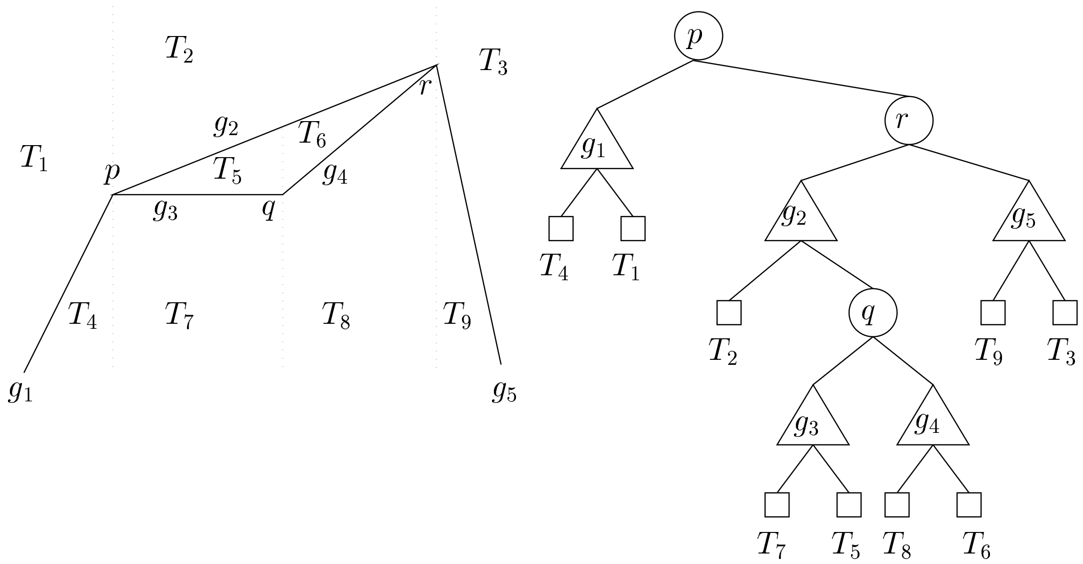


Abbildung 16: Eine trapezierte Unterteilung der Ebene und die Suchstruktur. Die x-Knoten werden als Kreise dargestellt, die y-Knoten als Dreiecke. Die Geradenstücke sind in der Reihenfolge g_2, g_5, g_3, g_4, g_1 eingefügt worden. In diesem Fall entsteht ein Baum mit Wurzel p, die Kanten sind von oben nach unten gerichtet.

Beispiel für eine trapezierte Unterteilung mit einer zugehörigen Suchstruktur ist in Abbildung 16 zu sehen.

Wichtig ist es zu bemerken, daß diese Suchstruktur im allgemeinen kein Baum ist. Es können beispielsweise zwei innere Knoten das selbe Blatt als Nachfolger haben.

Es gibt einen randomisiert inkrementellen Algorithmus, der die gewünschte Struktur in erwartet $O(n \log n)$ Zeit konstruiert, die erwartete Größe der Struktur ist $O(n)$. Dabei werden die einzelnen Geradenstücke nacheinander in zufälliger Reihenfolge eingefügt, nach jedem Einfügen wird die Suchstruktur angepaßt. Für jeden Punkt P beträgt die erwartete Abfragezeit $O(\log n)$. Die Wahrscheinlichkeit dafür, daß eine Einfügereihenfolge der Geradenstücke zu einem hohen Suchaufwand oder zu einer großen Suchstruktur führt, ist klein. Daher kann eine Suchstruktur, in der die Suche nach einem Punkt auch im schlechtesten Fall $O(\log n)$ Zeit benötigt und die im schlechtesten Fall $O(n)$ Speicherplatz braucht, konstruiert werden. Dafür wird einfach der Konstruktionsalgorithmus angewandt, bis er eine solche gewünschte Suchstruktur liefert.

Was ergibt sich damit für ein Arrangement $\mathcal{A}$ von n Geraden? Im schlechtesten Fall schneiden sich die Geraden jeweils paarweise in einem Punkt, und keine 3 Geraden haben einen gemeinsamen Schnittpunkt. Dann zerfällt jede Gerade in n Geradenstücke, und es gibt insgesamt $O(n^2)$ Geradenstücke in $\mathcal{A}$. Damit ergibt sich für diesen Fall eine Suchdauer von $O(\log n)$, und es wird $O(n^2)$ Speicherplatz benötigt. Der Aufbau der Suchstruktur für diesen Fall benötigt erwartet $O(n^2 \log n)$ Zeit.

Es gibt noch verschiedene andere Datenstrukturen für dieses Suchproblem, die das Lokalisieren eines Punktes in $O(\log n)$ Zeit ermöglichen und den gleichen Speicherplatz benötigen. Eine Übersicht gibt der Artikel von Preparata [13].

Die grundlegende Idee all dieser Verfahren ist es, die gegebene Unterteilung der Ebene zu verfeinern. Das führt zu Problemen, wenn das Verfahren auf Unterteilungen mit gewichteten Regionen übertragen werden soll. Beim Einfügen neuer Linien werden Regionen der ursprünglichen Unterteilung zerschnitten. Das Gewicht einer Region müßte also auf die entstehenden Teilregionen „aufgeteilt" werden, und die Gewichte sollten bei der Konstruktion berücksichtigt werden.

Diese Verfahren können in keinem Fall einen optimalen Suchbaum liefern, denn die Blätter der Suchstruktur repräsentieren jeweils nur Teile von Regionen. Durch eine Verfeinerung ist die Suche aufwendiger als nötig.

3.2 Ein Arrangement von d Parallelenscharen und ein d-dimensionales Gitter

Um in einem Arrangement von d Parallelenscharen $g_1, \ldots, g_d$ zu einem Punkt die Region, in der er liegt, zu bestimmen, reicht es aus, einen Suchbaum für ein d-dimensionales Gitter zu erstellen. In diesem Abschnitt wird gezeigt, wie die Suche im Arrangement auf das Gitter übertragen werden kann.

Definition 3.1 *m_i sei die Steigung der Geraden aus der Schar g_i und n_i die Anzahl der Geraden in g_i. Damit ist eine Gerade g_{ij} $(j = 1 \ldots n_i)$ aus g_i durch die Gleichung $g_{ij}(x) = m_i x + b_{ij}$ gegeben. Dabei sollen in einer Schar die Geraden nach aufsteigenden Achsenabschnitten b_{ij} numeriert und so gerichtet sein, daß $g_{i\,j-1}$ links von g_{ij} liegt.*
Eine Region $(i_1, \ldots, i_d)$ des Arrangements wird begrenzt durch $g_{1\,i_1-1}$ und g_{1i_1}, $g_{2\,i_2-1}$ und g_{2i_2}, $\ldots$, $g_{d\,i_d-1}$ und g_{di_d}.
Zu einem Punkt $P = (p_x, p_y)$ bezeichne das Tupel $(i_1, \ldots, i_d)_P$ die Region, in der P liegt.

Ein d-dimensionales Gitter sei durch Scharen von Hyperebenen $h_1, \ldots, h_d$, die senkrecht auf den Koordinatenachsen stehen, gegeben. Um einen Punkt Q im Gitter zu lokalisieren, sind Fragen der Art *Ist Q links oder rechts von der Hyperebene h?* erlaubt.

Definition 3.2 *h_i bezeichne die Schar von Hyperebenen, die senkrecht auf der i-ten Koordinatenachse stehen, n_i sei ihre Anzahl. Damit ist jede Hyperebene h_{ij} $(j = 1 \ldots n_i)$ aus h_i gegeben durch die Gleichung $x_i = \xi_{ij}$, wobei dann ξ_{ij} die Schnittstelle von h_{ij} mit der i-ten Koordinatenachse ist.*
Eine Zelle $(i_1, \ldots, i_d)$ des Gitters wird begrenzt durch $h_{1\,i_1-1}$ und h_{1i_1}, $h_{2\,i_2-1}$ und h_{2i_2},...,$h_{d\,i_d-1}$ und h_{di_d}.
Zu einem Punkt $Q = (q_1, \ldots, q_d)$ bezeichne das Tupel $(i_1, \ldots, i_d)_Q$ die Zelle, in der Q liegt.

Durch diese Wahl der Hyperebenen ist ein Punkt Q links von h_{ij} genau dann, wenn $q_i < \xi_{ij}$

Satz 3.1 *Wenn ein Suchbaum für ein d-dimensionales Gitter gegeben ist, kann durch geeignete Wahl der Hyperebenen damit auch ein Punkt in einem Arrangement von d Parallelenscharen lokalisiert werden.*

Beweis. Für ein gegebenes Arrangement und einen Punkt $P = (p_x, p_y)$ werden die Hyperebenen eines d-dimensionalen Gitters bestimmt durch die Achsenabschnitte der Geraden: Die Hyperebene h_{ij} sei gegeben durch $x_i = b_{ij}$. Gesucht wird im so definierten Gitter nach einem Punkt $Q = (q_1, \ldots, q_d)$, dessen Koordinaten durch $q_i := p_y - m_i p_x$ festgelegt seien. Anschaulich ist q_i der Schnittpunkt einer Geraden durch P parallel zu g_i mit der y-Achse.

Damit liegt P zwischen $g_{i\,j-1}$ und g_{ij} genau dann, wenn $b_{i\,j-1} < q_i < b_{ij}$. Durch die Wahl der Hyperebenen ist also $(i_1, \ldots, i_d)_P = (i_1, \ldots, i_d)_Q$. □

3.3 Ein Suchbaum für ein d-dimensionales gewichtetes Gitter

Im vorhergehenden Kapitel wurde exemplarisch der Spezialfall $d = 2$ untersucht. Darauf aufbauend werden nun Aussagen für beliebige Dimensionen gezeigt. Dabei sei ein d-dimensionales Gitter gegeben wie vorhergehend beschrieben, und jeder Zelle $(i_1, \ldots, i_d)$ sei das Gewicht $\alpha_{i_1 \ldots i_d}$ zugeordnet, die Ränder haben wieder das Gewicht Null. Zu einem gegebenen Punkt soll die Zelle, in der er liegt, bestimmt werden.

3.3.1 Die Pfadlänge eines optimalen Suchbaumes

Für die durchschnittliche Anzahl von Vergleichen, die in einem optimalen Suchbaum zum Lokalisieren eines Punktes benötigt werden, wird hier eine obere und untere Schranke angegeben. Dazu sind die folgenden Bezeichnungen hilfreich.

Definition 3.3 $\mathcal{I}^d := \{(i_1, \ldots, i_d) : \quad i_r \in \{1, \ldots, n_r\},\ r \in \{1, \ldots, d\}\}$ *sei die Menge aller Zellen eines d-dimensionalen Gitters und* $\mathcal{H}^d$ *bezeichne die Entropie für die Verteilung* $(\alpha_i)_{i \in \mathcal{I}^d}$ *auf den Zellen.*

Damit läßt sich die gewichtete Pfadlänge wie folgt abschätzen.

Satz 3.2 *Für ein d-dimensionales Gitter mit gegebener Gewichtung* $(\alpha_i)_{i \in \mathcal{I}^d}$ *der Zellen gibt es einen Suchbaum T, so daß* $p(T) \leq \mathcal{H}^d + 2d$.

Beweis. Induktion über die Dimension d.

- Für die Fälle $d = 1$ und $d = 2$ ist die Behauptung schon bewiesen.
- Es sei vorausgesetzt, daß es für ein $(d-1)$-dimensionales Gitter einen Baum T gibt, der die Bedingung erfüllt, also

$$p(T) = \sum_{\hat{\imath} \in \mathcal{I}^{d-1}} l_T(\hat{\imath}) \alpha_{\hat{\imath}} \leq \mathcal{H}^{d-1} + 2(d-1).$$

- Zu zeigen ist, daß es dann auch für ein d-dimensionales Gitter einen solchen Baum gibt. Dafür werden zuerst einige Definitionen benötigt:

  Definition 3.4 *Die Summe der Gewichte für eine feste letzte Koordinate* $j \in \{1, \ldots, n_d\}$ *sei*

$$p_j = \sum_{\substack{i \in \mathcal{I}^d \\ i_d = j}} \alpha_i = \sum_{\hat{\imath} \in \mathcal{I}^{d-1}} \alpha_{(\hat{\imath}, j)}.$$

  Für die Zellen $\hat{\imath} \in \mathcal{I}^{d-1}$ *werden Wahrscheinlichkeitsverteilungen*

$$q_{\hat{\imath}}(j) := \frac{\alpha_{(\hat{\imath}, j)}}{p_j}$$

  für jede feste letzte Koordinate $j \in \{1, \ldots, n_d\}$ *definiert.*

Konstruktion des Suchbaumes T:

1. Konstruiere einen optimalen Suchbaum T_1 für die letzte Koordinate bezüglich der Gewichte $p_1, \ldots, p_{n_d}$.
2. Hänge an jedes Blatt $(i_d = j)$ von T_1 einen optimalen Suchbaum $T_{d-1}(j)$ für das zugehörige $(d-1)$-dimensionale Gitter mit der entsprechenden Gewichtung $(q_{\hat{i}}(j))_{\hat{i}\in\mathcal{I}^{d-1}}$.

Behauptung. Für den so konstruierten Baum T ist $p(T) \leq \mathcal{H}^d + 2d$.
Beweis. Mit obigen Bezeichnungen ist

$$p(T) = \sum_{i\in\mathcal{I}^d} l_T(i)\alpha_i = \sum_{j=1}^{n_d} \sum_{\hat{i}\in\mathcal{I}^{d-1}} (l_{T_{d-1}(j)}(\hat{i}) + l_{T_1}(j))\alpha_{(\hat{i},j)}.$$

Durch Umformungen analog zum Beweis von Satz 2.1 und Anwenden der Induktionsvoraussetzung auf T_1 und $T_{d-1}(j)$ läßt sich das nach oben abschätzen mit

$$\sum_{j=1}^{n_d} p_j \mathcal{H}^{d-1}(j) + 2(d-1) + \mathcal{H}^1 + 2$$

und durch Einsetzen der entsprechenden Summen für $\mathcal{H}^{d-1}(j)$, $\mathcal{H}^1$ und $q_{\hat{i}}(j)$ sowie Anwenden von Logarithmengesetzen und Zusammenfassen ergibt sich ganz analog zur Rechnung im Beweis von Satz 2.1 Gleichheit mit

$$-\sum_{j=1}^{n_d} \sum_{\hat{i}\in\mathcal{I}^{d-1}} \alpha_{(\hat{i},j)} \log_2 \alpha_{(\hat{i},j)} + 2d = -\sum_{i\in\mathcal{I}^d} \alpha_i \log_2 \alpha_i + 2d = \mathcal{H}^d + 2d.$$

Somit ist ein Verfahren gegeben, das für jede Verteilung einen Suchbaum liefert, der die geforderte Bedingung erfüllt, und der Satz ist bewiesen. □

Korollar 3.3 *Für die gewichtete Pfadlänge eines optimalen Suchbaumes T_0 für ein d-dimensionales gewichtetes Gitter gilt*

$$\mathcal{H}^d \leq p(T_0) \leq \mathcal{H}^d + 2d.$$

Beweis. Die obere Schranke folgt sofort aus Satz 3.2. Der Beweis für die untere Schranke verläuft wie im Beweis von Satz 1.3, wobei hier über alle $i \in \mathcal{I}^d$ summiert werden muß. □

Diese Schranken sind scharf: Wieder wird die untere bei Gleichverteilung erreicht und die obere ist der Grenzwert für den Fall, daß eine Zelle fast die ganze und alle benachbarten echt positive, aber sehr kleine Wahrscheinlichkeiten tragen.

3.3.2 Konstruktion eines optimalen Suchbaumes

Ein optimaler Suchbaum kann auch für ein d-dimensionales gewichtetes Gitter dynamisch konstruiert werden, jedoch nur mit sehr großem Aufwand. Das Verfahren soll hier nur skizziert werden, da die Idee im wesentlichen dieselbe wie für den Fall $d = 2$ ist, nur sind jetzt entsprechend mehr Schleifen notwendig.

- Initialisieren der Pfadlängen und Gewichte für die einzelnen Zellen.
- Berechnen der Gewichte für alle Teilstücke des Gitters, die in jede der d Richtungen durch ihren rechten und linken Rand gegeben sind.
- Berechnen der Wurzeln der optimalen Suchbäume für „größer werdende“ Teilstücke des Gitters, indem für alle möglichen Aufteilungen geprüft wird, welche den Baum mit der minimalen Pfadlänge liefert.

In der Formulierung „größer werdende“ Teilstücke verbergen sich ineinandergeschachtelte Schleifen, die von den einzelnen Zellen ausgehend ein Anwachsen in alle Koordinatenrichtungen realisieren. Der folgende Satz läßt erkennen, daß dieses Verfahren recht aufwendig ist:

Satz 3.4 *Die dynamische Konstruktion eines optimalen Suchbaumes für ein d-dimensionales gewichtetes Gitter mit n Unterteilungen in jeder Koordinatenrichtung benötigt $O(n^{2d})$ Speicherplatz und $O(dn^{2d+1})$ Zeit.*

Beweis. Für die Fälle $d = 1$ und $d = 2$ ist die Behauptung in Satz 1.4 bzw. Satz 2.4 gezeigt, die Argumentation für den allgemeinen Fall läuft analog: Für ein Teilstück des Gitters mit Breite b_i bezüglich der i-ten Dimension $(i = 1 \ldots d)$ gibt es $\prod_{i=1}^{d}(n - b_i)$ Möglichkeiten. Damit werden insgesamt

$$\sum_{b_1=1}^{n} \cdots \sum_{b_d=1}^{n} \prod_{i=1}^{d}(n - b_i) \quad \in O(n^{2d})$$

Teilstücke des d-dimensionalen Gitters betrachtet. Um die Wurzel eines optimalen Suchbaumes für eines der Teilstücke zu bestimmen, muß aus jeweils höchstens nd Möglichkeiten die beste gesucht werden. Es ergibt sich die Zeitkomplexität $O(dn^{2d+1})$. □

3.4 Folgerungen für die Suche im Arrangement

Die Suche in einem gewichteten Arrangement von d Parallelenscharen kann auf die Suche in einem gewichteten d-dimensionalen Gitter zurückgeführt

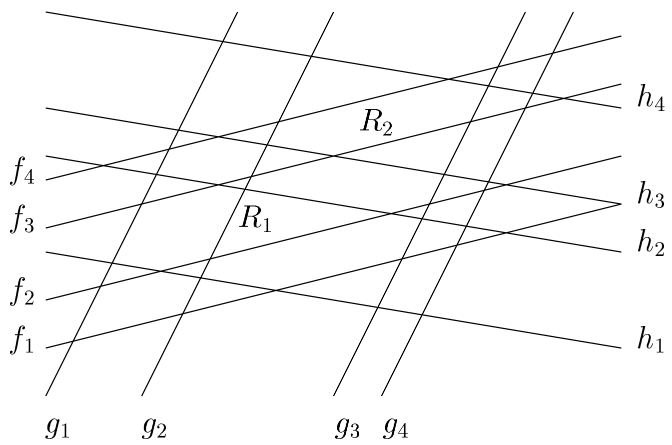


Abbildung 17: 3 Parallelenscharen f, g und h. Region R_2 wird von 6 Geraden begrenzt, Region R_1 nur von 3 Geraden. Die mögliche Region $(3, 1, 4)$ tritt hier nicht auf.

werden. Dabei werden zu den Geraden des Arrangements Hyperebenen des Gitters assoziiert, wie es im Teil 3.2 erklärt wurde. Eine zu einer Region des Arrangements korrespondierende d-dimensionale Zelle des Gitters erhält das Gewicht der Region. Viele Zellen des Gitters jedoch entsprechen keiner Region des Arrangements, siehe zum Beispiel Abbildung 17. Diese bekommen das Gewicht Null. Damit haben die Schranken aus Korollar 3.3 auch für einen optimalen Suchbaum für die Suche im Arrangement aus d Parallelenscharen Gültigkeit. Die obere Schranke ist scharf. Sie ist der Grenzwert für ein Arrangement, in dem es eine Region gibt, die von $2d$ Geraden begrenzt wird und fast das ganze Gewicht trägt und deren Nachbarregionen alle echt positive, aber sehr kleine Gewichte haben. Die untere Schranke könnte für ein Arrangement mit der Gleichverteilung angenommen werden, für das ein balancierter Suchbaum konstruiert werden kann. Die Frage, ob das immer möglich ist, ist Thema der Kapitel 4 und 5.
Damit gilt der Satz:

Satz 3.5 *Für die gewichtete Pfadlänge $p(T)$ eines optimalen Suchbaumes T für ein gewichtetes Arrangement von d Parallelenscharen ist*

$$\mathcal{H}^d \leq p(T) \leq \mathcal{H}^d + 2d,$$

und die obere Schranke ist scharf. □

Wie im Fall $d = 2$ stellt sich auch hier die Frage, ob es nicht möglich ist, durch Einbeziehen der geometrischen Struktur diese Schranken zu verbessern.

In einem Arrangement von d Parallelenscharen werden fast keine Regionen tatsächlich durch Geraden aus jeder der Scharen begrenzt, häufig reichen weniger Angaben aus, um die Region eindeutig zu benennen.

Die über dem d-dimensionalen Gitter erzeugte Suchstruktur erfüllt zwar den gewünschten Zweck, die Konstruktion ist jedoch komplexer als das ursprüngliche Problem: Ein d-dimensionales Gitter, das jede Koordinatenrichtung durch n Hyperebenen unterteilt, hat n^d Zellen. Das sind um Größenordnungen mehr als es Regionen in einem Arrangement von d Parallelenscharen mit jeweils n Geraden gibt, wie der folgende Satz zeigt:

Satz 3.6 *Für die Anzahl der Regionen $\mathcal{Z}_d(n)$ in einem Arrangement von d Parallelenscharen mit jeweils n Geraden gilt*

$$\mathcal{Z}_d(n) \leq \frac{d(d-1)}{2} n^2 + dn + 1 \quad \text{,also} \quad \mathcal{Z}_d(n) \in O((nd)^2).$$

Beweis. Induktion über Anzahl der Parallelenscharen d.

- Durch *eine* Schar g_1 von n parallelen Geraden wird die Ebene in $n+1$ Streifen geteilt, also $\mathcal{Z}_1(n) = 1 \cdot n + 1$.
- Kommt eine weitere Schar g_2 dazu, so schneidet jede Gerade aus g_2 alle Geraden von g_1. Bei jedem Schnitt durch eine Region kommt genau eine neue Region hinzu, und damit ist $\mathcal{Z}_2(n) = \mathcal{Z}_1(n) + n\mathcal{Z}_1(n) = (n+1)^2 = n^2 + 2n + 1$.
- Angenommen die Behauptung gilt für die Parallelenscharen $g_1, \ldots, g_{d-1}$. Werden nun Geraden einer neuen Schar g_d eingefügt, so schneidet jede der neuen Geraden alle $(d-1)n$ schon vorhandenen Geraden. Dabei wird nach jeder Schnittstelle eine vorhandene Region zerteilt in zwei neue Regionen, und vor dem ersten Schnitt wird eine unbeschränkte Region zerteilt. Es kann natürlich passieren, daß eine Gerade aus g_n durch eine *Schnittstelle* von k schon vorhandenen Geraden geht. Dann entsteht durch den Schnitt der neuen Geraden mit k anderen lediglich *eine* Region mehr. Somit können beim Einfügen einer Geraden aus der neuen Schar höchstens $(d-1)n+1$ Regionen hinzukommen, und damit ist insgesamt

  $$\mathcal{Z}_d(n) \leq \mathcal{Z}_{d-1}(n) + n((d-1)n+1) \quad .$$

  Das ist nach der Voraussetzung

  $$\leq \frac{(d-1)(d-2)}{2} n^2 + (d-1)n + 1 + n((d-1)n+1) = \frac{d(d-1)}{2} n^2 + dn + 1,$$

  womit der Satz bewiesen ist. □

Versucht man also, das Problem der Konstruktion eines optimalen Suchbaumes für ein Arrangement über den optimalen Suchbaum eines zugehörigen gewichteten d-dimensionalen Gitters zu lösen, so ist es sehr komplex. Zum einen ist der Speicherbedarf hoch, und zum anderen braucht die Konstruktion viel Zeit: Beides wächst exponentiell mit der Anzahl d der Parallelenscharen.

In einem Arrangement von d Parallelenscharen mit jeweils n Geraden gibt es nicht alle Regionen und Teilstücke, die formal beschrieben werden können. Auf den ersten Blick erscheint es daher so, als könnte durch dynamisches Programmieren direkt in der Ebene ein optimaler Suchbaum deutlich schneller konstruiert werden, als es mit dem „Umweg" über ein d-dimensionales Gitter möglich ist. Die Idee dabei ist, nur Teilstücke zu betrachten, die tatsächlich auftreten. Mögliche Kandidaten für eine geschickte Aufteilung eines Teilstückes sind die Geraden, die es durchqueren. Wie der folgende Satz zeigt, kann die Konstruktion dadurch nicht wesentlich beschleunigt werden.

Satz 3.7 *Ein Arrangement von d Parallelenscharen mit jeweils n Geraden kann mehr als $\left(\frac{n}{2}\right)^{2d}$ verschiedene Teilstücke haben.*

Beweis. Ein Teilstück eines Arrangements von d Parallelenscharen wird durch einen Vektor $(l_1, r_1, \ldots, l_d, r_d)$ beschrieben, wobei l_i den linken und r_i den rechten Rand bezüglich der Schar g_i angibt.

Nun wird folgendes Arrangement betrachtet: Jede der d Scharen wird in zwei Bündel mit jeweils $\frac{n}{2}$ Geraden aufgeteilt. Die Bündel werden so angeordnet, daß ein regelmäßiges $2d$-gon entsteht, siehe Abbildung 18.

Für jeden Vektoreintrag l_i und r_i kann aus $\frac{n}{2}$ Möglichkeiten frei gewählt werden. Das so bezeichnete Teilstück existiert in dem beschriebenen Arrangement. Für zwei verschiedene Vektoren sind auch die dadurch beschriebenen Teilstücke unterschiedlich. Damit gibt es $\left(\frac{n}{2}\right)^{2d}$ Teilstücke, die von $2d$ Geraden begrenzt werden.

Dazu kommen nun noch Teilstücke, die von weniger Geraden begrenzt werden, diese sind jedoch nicht so einfach zu zählen. □

Die Anzahl der Teilstücke eines Arrangements ist also exponentiell in d. Damit kann die Komplexität der Konstruktion nicht wesentlich gesenkt werden, wenn nur Teilstücke des Arrangements betrachtet werden, die tatsächlich vorkommen. Somit ist es eine vernünftige Lösung, einen optimalen Suchbaum für ein Arrangement über den optimalen Suchbaum eines d-dimensionalen Gitters zu konstruieren.

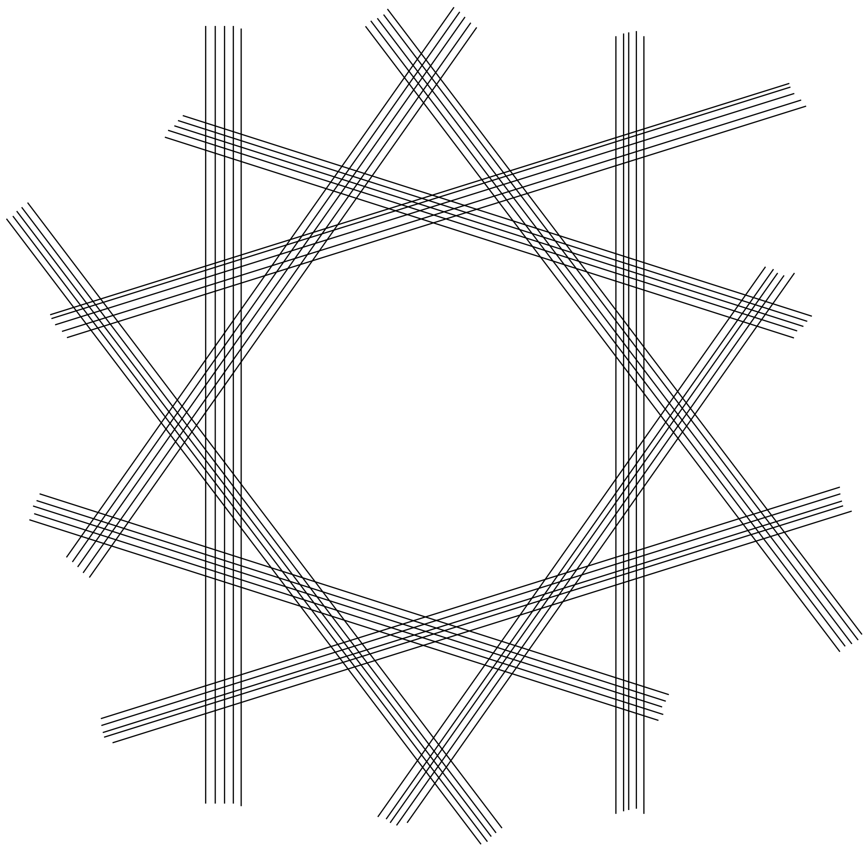


Abbildung 18: Ein Arrangement von 5 Parallelenscharen mit jeweils 10 Geraden. Jede Schar ist in 2 Bündel zu 5 Geraden aufgeteilt, die Bündel bilden ein regelmäßiges Polygon.

4 δ-zentrale Geraden in einem Arrangement

In einem Suchbaum T zu einem Arrangement $\mathcal{A}$, der das Auffinden der Region zu einem gegebenen Punkt P ermöglicht, sind Fragen der Art *Liegt P links oder rechts von der Geraden g?* erlaubt. Jeder Unterbaum von T entspricht also dem Schnitt von Halbräumen, die jeweils durch Geraden des Arrangements $\mathcal{A}$ begrenzt sind.

Definition 4.1 *Ein* partielles Arrangement *eines gegebenen Arrangements $\mathcal{A}$ ist der Schnitt von Halbräumen, die durch Geraden von $\mathcal{A}$ begrenzt werden.*

Damit ist jeder Unterbaum eine Suchstruktur für ein partielles Arrangement. Ein Suchbaum ist gut, wenn jeder Unterbaum möglichst balanciert ist. Einen solchen Suchbaum gibt es dann, wenn sich in jedem partiellen Arrangement von $\mathcal{A}$ jeweils eine Gerade h befindet, so daß links und rechts von h ungefähr gleichviele Regionen liegen.

Definition 4.2 *Eine Gerade h in einem Arrangement $\mathcal{A}$ mit r Regionen heißt* Δ-balancierend *($0 < \Delta \leq \frac{1}{2}$), wenn auf jeder Seite von h mindestens Δr Regionen liegen.*

Falls es in jedem partiellen Arrangement eine Δ-balancierende Gerade mit $\Delta \approx \frac{1}{2}$ gibt, kann ein gut balancierter Baum folgendermaßen konstruiert werden: Für jeden Unterbaum wird die Δ-balancierende Gerade h des zugehörigen partiellen Arrangements gesucht und in die Wurzel die Abfrage *Ist der Punkt rechts oder links von h?* geschrieben. Damit kann garantiert werden, daß im linken und rechten Unterbaum jeweils mindestens der Teil Δ der zum partiellen Arrangement dieses Unterbaumes gehörenden Regionen als Blätter auftaucht. Ein so konstruierter Suchbaum für ein Arrangement mit r Regionen hat höchstens die Tiefe $\frac{\log_2 r}{-\log_2(1-\Delta)} \in O(\log r)$.

In diesem Kapitel wird für Arrangements untersucht, in welchem Verhältnis die am besten geeignete Gerade die Regionen im schlechtesten Fall aufteilt. Dabei werden zuerst die Schnittpunkte zwischen Geraden des Arrangements rechts und links von h gezählt. Anschließend wird der Zusammenhang zwischen der Anzahl der Schnittpunkte und der Anzahl der Regionen auf einer Seite von h dargestellt.

Im folgenden Kapitel werden die hier angestellten Überlegungen verallgemeinert, um so zu Ergebnissen für partielle Arrangements zu kommen.

Definition 4.3 *Eine Gerade h in einem Arrangement $\mathcal{A}$ mit m_h Schnittpunkten, die nicht auf h liegen, heißt* δ-zentral *($0 \leq \delta \leq \frac{1}{2}$), wenn auf jeder Seite von h mindestens δm_h Schnittpunkte liegen.*

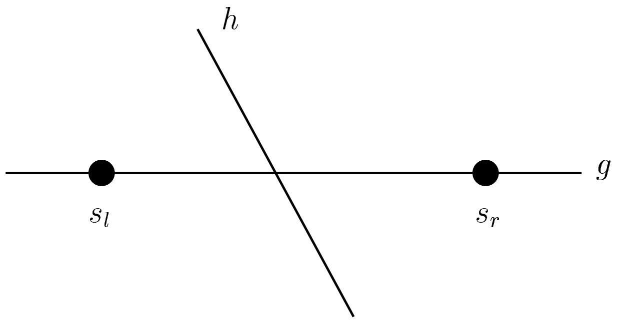


Abbildung 19: Eine gute Kombination (g, h, s_l, s_r).

Eine Gerade heißt zentral, *wenn sie δ-zentral ist und $\delta > 0$, oder wenn weder rechts noch links Schnittpunkte liegen.*

Für eine gute Aufteilung ist es also wünschenswert, daß es immer eine Gerade h gibt, die die Anzahl der Schnittpunkte rechts und links ungefähr halbiert. Es ist das Ziel, ein δ zu bestimmt, so daß die Existenz einer δ-zentralen Geraden garantiert werden kann.

Bemerkung. In einem Arrangement mit $n \leq 3$ Geraden hat jede Gerade eine Seite, auf der *kein* Schnittpunkt liegt, dort gibt es also keine δ-zentrale Gerade mit $\delta > 0$. In solch einem kleinen Arrangement werden jedoch höchstens 3 Fragen zum Lokalisieren eines Punktes benötigt, so daß es keine Einschränkung ist, wenn im folgenden nur Arrangements mit $n \geq 4$ Geraden betrachtet werden.

4.1 Ein Spezialfall

Zuerst wird der Fall untersucht, daß sich je 2 Geraden des Arrangements in einem Punkt schneiden. Weiterhin sollen niemals 3 Geraden einen gemeinsamen Schnittpunkt haben.

Ist ein solches Arrangement mit $n \geq 4$ Geraden $g_1, \ldots, g_n$ gegeben, dann liegen auf jeder Geraden $n-1$ Schnittpunkte, und insgesamt gibt es $\binom{n}{2}$ Schnittpunkte. Damit ist zu jeder Geraden h die Anzahl m_h der Schnittpunkte, die nicht auf h liegen,

$$m_h = \binom{n}{2} - (n-1) = \frac{(n-1)(n-2)}{2}.$$

Definition 4.4 *Ein Tupel (g, h, s_l, s_r) von Geraden g, h und Schnittpunkten s_l, s_r auf g heißt* gute Kombination, *wenn g von h so geschnitten wird, daß s_l links und s_r rechts von h liegt (siehe Abbildung 19).*

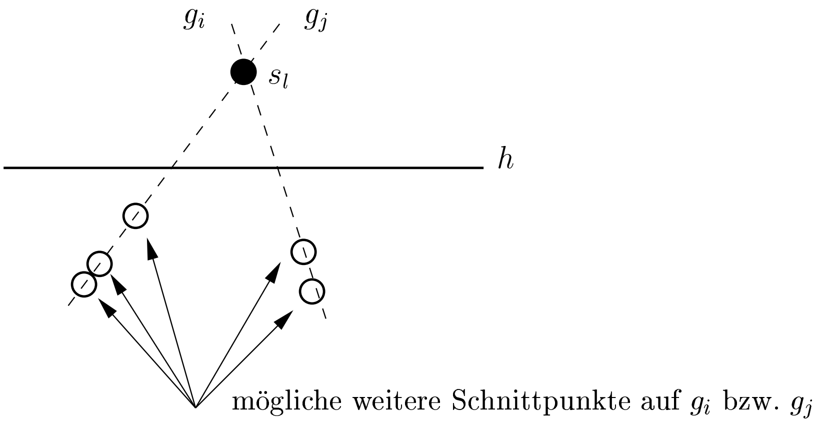


Abbildung 20: Wie oft wird ein Schnittpunkt s_l in guter Kombination mit $\tilde{h}$ gezählt?

Auf jeder Geraden g_i liegen $n-1$ Schnittpunkte, und jeweils 3 Schnittpunkte bilden eine gute Kombination mit g_i: Die beiden äußeren Punkte sind s_l bzw. s_r und h ist die Gerade, die durch den mittleren der Schnittpunkte geht. Damit gibt es zu jeder Geraden g_i genau $\binom{n-1}{3}$ gute Kombinationen (g_i, h, s_l, s_r), bei n Geraden also $n\binom{n-1}{3}$ gute Kombinationen.

Da die Geraden h in diesen Kombinationen auch nur aus $\{g_1, \ldots, g_n\}$ sein können, muß es eine Gerade $\tilde{h}$ geben, die in *mindestens* $\binom{n-1}{3}$ guten Kombinationen vorkommt.

Wieviele Punkte liegen nun auf einer Seite von $\tilde{h}$? Mindestens so viele, wie durch die guten Kombinationen mit $\tilde{h}$ gegeben sind. Ein Punkt s_l (und analoge Überlegungen gelten natürlich auch für s_r) kann allerdings mehrfach in guter Kombination mit $\tilde{h}$ vorkommen. s_l ist Schnittpunkt von 2 Geraden g_i und g_j, kann also in guten Kombinationen der Form $(g_i, h, s_l, \cdot)$ und $(g_j, h, s_l, \cdot)$ vorkommen, siehe Abbildung 20. Für den zweiten Punkt s_r der guten Kombination gibt es in jedem der beiden Fälle höchstens noch $n-3$ Möglichkeiten, denn auf einer Geraden liegen ja insgesamt nur $n-1$ Schnittpunkte. Somit kann ein Schnittpunkt s_l in guter Kombination mit $\tilde{h}$ höchstens $2(n-3)$ Mal gezählt werden. Auf jeder Seite von $\tilde{h}$ liegen also mindestens

$$\frac{1}{2(n-3)}\binom{n-1}{3} = \frac{(n-1)(n-2)(n-3)}{2(n-3)6} = \frac{1}{6}\frac{(n-1)(n-2)}{2} = \frac{1}{6}m_{\tilde{h}}$$

Schnittpunkte. Das führt zu dem folgenden Satz.

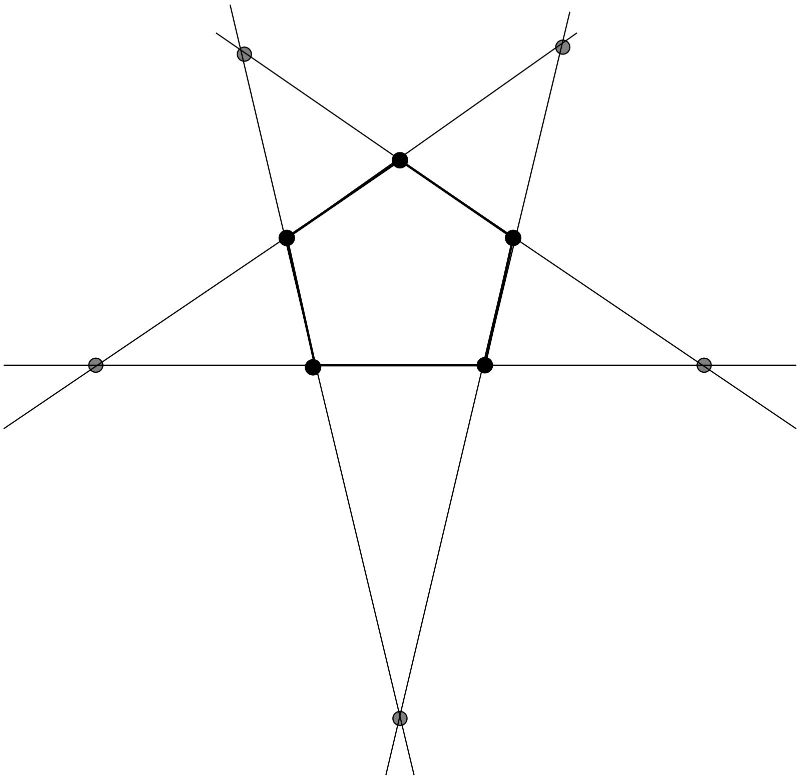


Abbildung 21: Beim regelmäßigen 5-gon ist jede Gerade $\frac{1}{6}$-zentral.

Satz 4.1 *In einem Arrangement mit $n \geq 4$ Geraden, in dem sich je zwei Geraden in einem Punkt schneiden und niemals 3 Geraden einen gemeinsamen Schnittpunkt haben, gibt es immer eine $\frac{1}{6}$-zentrale Gerade h.* $\square$

Wie in Abbildung 21 zu erkennen ist, ist $\delta = \frac{1}{6}$ sogar die bestmögliche Abschätzung: Jede der Geraden teilt die Schnittpunkte im Verhältnis $1 : 5$ auf. Für den betrachteten Spezialfall ist in einem regelmäßigen $(2l + 1)$-gon die Aufteilung der Schnittpunkte auf die Seiten der Geraden schlechtestmöglich. Für großes l ist dabei, wie das folgende Beispiel zeigt, jede Gerade annähernd $\frac{1}{4}$-zentral. Die Differenz zu dem bei der Abschätzung erhaltenen Wert $\frac{1}{6}$ für δ erklärt sich dabei folgendermaßen:

In der Abschätzung wird damit gerechnet, daß ein Schnittpunkt s in guter Kombination mit der betrachteten Gerade $\tilde{h}$ und der Geraden g, auf der er liegt, höchstens $(n-3)$-mal vorkommt. s wird genau dann $(n-3)$-mal gezählt, wenn s der äußerste Schnittpunkt auf g ist und der Schnittpunkt von g und $\tilde{h}$ direkt neben s liegt. Diese Konstellation kann aber nicht für viele Schnittpunkte gelten, denn auf einer Seite von $\tilde{h}$ sind nur $n-1$ Punkte äußere Schnittpunkte auf einer Geraden g, und $\tilde{h}$ wird auch nicht jedes g an der zweiten Stelle kreuzen. Somit wird ein Schnittpunkt durchschnittlich weniger als $2(n-3)$-mal gezählt, was die Abweichung erklärt.

Beispiel. Schnittpunkte bei einem regelmäßigen $(2l+1)$-gon, $l \geq 2$.
Wegen der Symmetrie ist das Verhältnis der Schnittpunkte links und rechts für alle Geraden gleich. Dreht man das $(2l+1)$-gon so, daß die obere Gerade h horizontal verläuft, so liegen oberhalb von h nur Schnittpunkte von Geraden der linken mit solchen der rechten Seite des Polygons. Die erste Gerade der linken Seite schneidet oberhalb von h genau $l-1$ Geraden von rechts, die nächste $l-2$ usw., so daß oberhalb von h insgesamt $\sum_{i=1}^{l-1} i = \frac{(l-1)l}{2}$ Schnittpunkte liegen.

Unterhalb von h liegen dann die restlichen, und das sind

$$\underbrace{\binom{2l+1}{2}}_{\text{alle Schnittpunkte}} \quad - \quad \underbrace{\frac{l(l-1)}{2}}_{\text{Schnittpunkte oberhalb}} \quad - \quad \underbrace{2l}_{\text{Schnittpunkte auf } h}$$

Stück. Damit verteilen sich die Schnittpunkte im Verhältnis

$$\frac{l(l-1)}{2} \quad : \quad \frac{3l^2-l}{2} \quad ,$$

und es ergibt sich

$$\delta = \frac{l-1}{4l-2} \quad ,$$

so daß für großes l $\quad \delta \approx \frac{1}{4}$ ist.

4.2 Der allgemeine Fall

Es werden nun Arrangements betrachtet, in denen Geraden auch parallel zueinander verlaufen dürfen. Weiterhin sollen niemals 3 Geraden einen gemeinsamen Schnittpunkt haben. Es stellt sich die Frage, für welches δ in diesem Fall die Existenz einer δ-zentralen Geraden garantiert werden kann.

Bei der Untersuchung wird sich zeigen, daß eine gute Abschätzung für δ in Abhängigkeit von der Anzahl der Geraden in der größten Parallelenschar möglich ist. Die Abschätzung gliedert sich in 2 Teile. Je nachdem, wieviele Geraden die größte Parallelenschar enthält, ist das Ergebnis des einen oder des anderen Falles besser.

4.2.1 Die größte Parallelenschar hat $k \geq \lambda n$ Geraden

Gibt es viele sich paarweise nicht schneidende Geraden, so ist es eine gute Lösung, die mittlere Gerade h_m der größten Parallelenschar $\mathcal{P}$ zu betrachten.

Für diese läßt sich leicht abschätzen, wie viele Schnittpunkte mindestens auf einer Seite liegen. Im folgenden sei $n \geq 4$ die Gesamtzahl von Geraden, k die Anzahl von Geraden in $\mathcal{P}$ und $\lambda \in]0;1]$. Für den Fall $k \geq \lambda n$ läßt sich die Anzahl von Schnittpunkten auf einer Seite von h_m nun folgendermaßen abschätzen:

Jede der k Geraden aus $\mathcal{P}$ wird von allen $n-k$ restlichen Geraden geschnitten. Auf einer Seite von h_m liegen (bei ungeradem k genau, bei geradem k mindestens) $\lfloor \frac{k-1}{2} \rfloor$ Geraden aus $\mathcal{P}$. Im schlechtesten Fall schneiden sich alle $n-k$ restlichen Geraden paarweise auf derselben Seite von h_m. Auf der Seite von h_m mit weniger Schnittpunkten gibt es dann mindestens

$$\left\lfloor \frac{k-1}{2} \right\rfloor (n-k) \geq \frac{k-2}{2}(n-k) \geq \frac{n-k}{2}(\lambda n - 2)$$

Schnittpunkte, und auf der anderen Seite liegen höchstens

$$\left\lceil \frac{k-1}{2} \right\rceil (n-k) + \binom{n-k}{2} \leq \frac{k}{2}(n-k) + \frac{n-k}{2}(n-k-1) = \frac{n-k}{2}(n-1)$$

Schnittpunkte. Das heißt, die Schnittpunkte verteilen sich im schlechtesten Fall im Verhältnis

$$\lambda n - 2 \quad : \quad n-1$$

auf beide Seiten von h_m. Damit ergibt sich der nächste Satz.

Satz 4.2 *In einem Arrangement mit $n \geq 4$ Geraden, in dem die größte Parallelenschar $k \geq \lambda n$ Geraden besitzt und sich keine 3 Geraden in einem Punkt schneiden, gibt es eine δ-zentrale Gerade h_m mit $\delta = \frac{\lambda n - 2}{(1+\lambda)n-3}$.* $\square$

Bemerkung.

1. Für $\lambda n \leq 2$ ist dieses Ergebnis sinnlos; es ist nur für Arrangements mit $k \geq 3$ parallelen Geraden interessant. Diese Abschätzung ist gut, falls es verhältnismäßig viele parallele Geraden gibt. Die Fälle mit $k < 3$ werden von der Rechnung in 4.2.2 erfaßt.

2. Die so gewählte Gerade h_m ist nicht notwendig diejenige, welche die Schnittpunkte am besten aufteilt, siehe Abbildung 22.

3. Für „große“ Arrangements ist $\delta \approx \frac{\lambda}{1+\lambda}$.

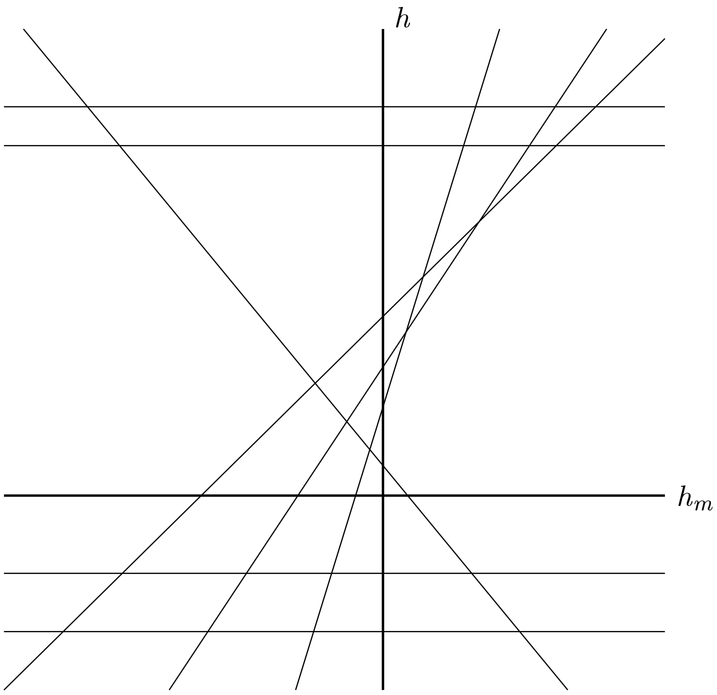


Abbildung 22: Alle Schnittpunkte zwischen den Geraden, die nicht zu $\mathcal{P}$ gehören, liegen auf einer Seite von h_m. h_m teilt die Schnittpunkte im Verhältnis 10:20, h hingegen im Verhältnis 12:14 auf.

4.2.2 Die größte Parallelenschar hat $k < \lambda n$ Geraden

Im folgenden bezeichne $s(g_i)$ die Anzahl der Schnittpunkte auf der Geraden g_i. Es ist dann $s(g_i) \leq n - 1$ für alle $i = 1 \ldots n$, und die Gesamtzahl aller Schnittpunkte ist $\frac{1}{2}\sum_{i=1}^{n} s(g_i)$. Damit gibt es zu jeder Geraden g_i genau $\binom{s(g_i)}{3}$ gute Kombinationen, insgesamt also $\sum_{i=1}^{n}\binom{s(g_i)}{3}$ Stück. Daraus folgt wie im zuerst untersuchten Spezialfall die Existenz einer Geraden $\tilde{h}$, die an mindestens $\frac{1}{n}\sum_{i=1}^{n}\binom{s(g_i)}{3}$ guten Kombinationen beteiligt ist.

γ_i sei die Anzahl guter Kombinationen von $\tilde{h}$ mit g_i. Dann ist

$$\sum_{i=1}^{n}\gamma_i \geq \frac{1}{n}\sum_{i=1}^{n}\binom{s(g_i)}{3}.$$

Wieviele Schnittpunkte liegen nun auf einer Seite von $\tilde{h}$? Ein Schnittpunkt s auf einer Geraden g_i kann in guter Kombination mit $\tilde{h}$ und g_i höchstens $(s(g_i) - 2)$-mal gezählt werden, denn soviele Möglichkeiten gibt es maximal für den zweiten Schnittpunkt in der Kombination.

s ist der Schnittpunkt von 2 Geraden, liegt also auf einer weiteren Geraden g_j und kann auch bei guten Kombinationen von $\tilde{h}$ und g_j gezählt worden sein. Mit diesen Beobachtungen ergibt sich, daß auf einer Seite von $\tilde{h}$ mindestens

$$\frac{1}{2}\sum_{i=1}^{n}\frac{\gamma_i}{s(g_i)-2}$$

Schnittpunkte liegen. Wegen $s(g_i) \leq n-1$ ist das

$$\geq \frac{1}{2}\sum_{i=1}^{n}\frac{\gamma_i}{n-3} = \frac{1}{2(n-3)}\ \sum_{i=1}^{n}\gamma_i \quad ,$$

was nach der obigen Überlegung

$$\geq \frac{1}{2(n-3)}\ \frac{1}{n}\sum_{i=1}^{n}\binom{s(g_i)}{3} = \frac{1}{6n(n-3)}\ \frac{1}{2}\sum_{i=1}^{n}s(g_i)\,(s(g_i)-1)\,(s(g_i)-2)$$

ist. Da die größte Schar paralleler Geraden $k < \lambda n$ Geraden hat, liegen auf jeder Geraden mindestens $(1-\lambda)n$ Schnittpunkte. Wird $s(g_i)$ damit abgeschätzt, ist der Ausdruck

$$\begin{aligned} &\geq \quad \frac{1}{6n(n-3)}\left((1-\lambda)n-1\right)\left((1-\lambda)n-2\right)\ \frac{1}{2}\sum_{i=1}^{n}s(g_i) \\ &= \quad \frac{(1-\lambda)^2n^2-3(1-\lambda)n+2}{6n^2-18n}\ \frac{1}{2}\sum_{i=1}^{n}s(g_i) \quad . \end{aligned}$$

Da $\frac{1}{2}\sum_{i=1}^{n}s(g_i)$ die Gesamtzahl aller Schnittpunkte ist, läßt sich damit der folgende Satz formulieren.

Satz 4.3 *In einem Arrangement mit $n \geq 4$ Geraden, in dem die größte Parallelenschar $k < \lambda n$ Geraden besitzt und sich keine 3 Geraden in einem Punkt schneiden, gibt es eine δ-zentrale Gerade h mit*

$$\delta = \frac{(1-\lambda)^2n^2-3(1-\lambda)n+2}{6n^2-18n} \quad .$$

□

Bemerkung. Für $\lambda n \leq 1$ ist das Ergebnis sinnlos, weil dabei „Scharen“ mit 0 Geraden betrachtet würden. Für „große“ Arrangements ist $\delta \approx \frac{(1-\lambda)^2}{6}$.

4.3 Zusammenfassendes Ergebnis

Die Abschätzung für den ersten Fall 4.2.1 ist nur dann gut, wenn es eine große Parallelenschar gibt, und die für den zweiten Fall 4.2.2 ist um so besser, je kleiner die größte Parallelenschar ist. Wie aus den Tabellen 2 und 3 auf Seite 58 ersichtlich ist, verbessert sich das Ergebnis in beiden Fällen mit wachsender Größe des Arrangements. Es gilt:

Satz 4.4 *In einem Arrangement mit $n \geq 4$ Geraden, in dem die größte Parallelenschar λn Geraden enthält und sich keine 3 Geraden in einem Punkt schneiden, gibt es immer eine δ-zentrale Gerade mit $\delta = \max\{\delta_1, \delta_2\}$, wobei*

$$\delta_1 = \frac{\lambda n - 2}{(1+\lambda)n - 3} \qquad \textit{und} \qquad \delta_2 = \frac{(1-\lambda)^2 n^2 - 3(1-\lambda)n + 2}{6n^2 - 18n} \quad .$$

Beweis. Das folgt sofort aus Satz 4.2 und Satz 4.3. □

Die folgende Tabelle gibt einen Überblick für einige Beispielwerte.

n	12	20	30	50	100	500	1000
$\tilde{\delta}$	0,086	0,101	0,109	0,114	0,119	0,122	0,123

Tabelle 1: Werte für $\tilde{\delta} = \min_{\lambda \in]0,1]} \delta$ in Abhängigkeit von n.

Bemerkung. Bei Arrangements mit $n \leq 6$ lassen sich teilweise bessere Werte für δ bestimmen, als Satz 4.4 liefert. Für den Fall, daß sich alle Geraden schneiden, ist schon in Satz 4.1 gezeigt worden, daß es eine $\frac{1}{6}$-zentrale Gerade gibt. Nun werden Arrangements mit $n \leq 6$ Geraden und $|\mathcal{P}| > 1$ betrachtet, wobei $\mathcal{P}$ eine größte Parallelenschar bezeichnet.

- $n = 4$: Es kann passieren, daß es gar keine zentrale Gerade gibt. Das ist dann der Fall, wenn jeweils zwei Geraden parallel zueinander liegen. Dabei hat jede Gerade eine Seite, auf der kein Schnittpunkt liegt.

- $n = 5$: Wenn $\mathcal{P}$ 2 Geraden enthält, gibt es insgesamt höchstens $\binom{5}{2} - 1 = 9$ Schnittpunkte, und auf jeder Geraden liegen mindestens 3 Schnittpunkte. Zu einer beliebigen Gerade g gibt es also höchstens $9 - 3 = 6$ Schnittpunkte, die nicht auf g liegen. In einem Arrangement mit 5 Geraden und $|\mathcal{P}| = 2$ gibt es immer eine Gerade h, bei der auf jeder Seite mindestens 1 Schnittpunkt liegt. Damit ist h $\frac{1}{6}$-zentral.

  Falls $|\mathcal{P}| = 3$, ist $\lambda = \frac{3}{5}$ und es ergibt sich $\delta_1 = \frac{1}{5}$. Bei größeren Werten für λ verbessert sich der Wert von δ_1.

Somit gibt es in einem Arrangement von 5 Geraden immer eine $\frac{1}{6}$-zentrale Gerade.

- $n = 6$: Falls $|\mathcal{P}| = 2$, gibt es höchstens $\binom{6}{2} - 1 = 14$ Schnittpunkte, und auf jeder Geraden liegen mindestens 4 Schnittpunkte. Für eine beliebige Gerade g gibt es somit höchstens 10 Schnittpunkte, die nicht auf g liegen. Außerdem gibt es immer eine Gerade h, die auf jeder Seite mindestens einen Schnittpunkt hat. Damit ist h $\frac{1}{10}$-zentral.

  Für $|\mathcal{P}| = 3$ ist $\lambda = \frac{1}{2}$ und damit $\delta_1 = \frac{1}{6}$. Für größere Parallelenscharen wird der Wert von δ_1 besser.

  Daraus folgt, daß es in einem Arrangement von 6 Geraden immer eine $\frac{1}{10}$-zentrale Gerade gibt.

Ab $n = 7$ lassen sich auf so offensichtliche Art und Weise keine besseren Werte für δ bestimmen, als Satz 4.4 liefert. Es ergeben sich die folgenden Werte:

- $n = 7$:

  $|\mathcal{P}| = 2 \quad \Rightarrow \quad \lambda = \frac{2}{7} \quad \Rightarrow \quad \delta_2 = \frac{1}{14}.$

  $|\mathcal{P}| = 3 \quad \Rightarrow \quad \lambda = \frac{3}{7} \quad \Rightarrow \quad \delta_1 = \frac{1}{7}.$

  Für mehr parallele Geraden verbessert sich der Wert von δ_1.

- $n = 8$:

  $|\mathcal{P}| = 2 \quad \Rightarrow \quad \lambda = \frac{1}{4} \quad \Rightarrow \quad \delta_2 = \frac{1}{12}.$

  $|\mathcal{P}| = 3 \quad \Rightarrow \quad \lambda = \frac{3}{8} \quad \Rightarrow \quad \delta_1 = \frac{1}{8}.$

  Für mehr parallele Geraden verbessert sich der Wert von δ_1.

- $n = 9$:

  $|\mathcal{P}| = 2 \quad \Rightarrow \quad \lambda = \frac{2}{9} \quad \Rightarrow \quad \delta_2 = \frac{5}{54} \approx 0,093.$

  $|\mathcal{P}| = 3 \quad \Rightarrow \quad \lambda = \frac{1}{3} \quad \Rightarrow \quad \delta_1 = \frac{1}{9}.$

  Für mehr parallele Geraden verbessert sich der Wert von δ_1.

Für wachsendes n werden die Werte von δ_1 und δ_2 besser. Damit ergibt sich die schlechteste Abschätzung für den Fall $n = 7$ und $|\mathcal{P}| = 2$. Dabei ist $\delta = \frac{1}{14}$. Es ergibt sich der folgende Satz.

Satz 4.5 *In einem Arrangement mit $n \geq 5$ Geraden, in dem sich keine 3 Geraden in einem Punkt schneiden, gibt es immer eine δ-zentrale Gerade mit $\delta \geq \frac{1}{14} \approx 0,071$.* □

In Abbildung 23 wurde mit *maple* die Abhängigkeit δ von λ für beide Fälle bei „großen" Arrangements skizziert. Der Schnittpunkt beider Kurven in diesem Bereich liegt bei $(0,1404 \mid 0,1231)$, das heißt bis $\lambda = 0,1404$ ist δ_2 besser, für größeres λ liefert δ_1 bessere Werte. Wird immer der jeweils günstigere Fall gewählt, so ergibt sich als schlechtestes Ergebnis $\delta = 0,1231$.

Satz 4.6 *In einem „großen" Arrangement, in dem sich keine 3 Geraden in einem Punkt schneiden, gibt es eine δ-zentrale Gerade mit $\delta \geq 0,1231$.* □

Bemerkung. Wie Tabelle 1 zeigt, gibt es beispielsweise für $n = 100$ eine Gerade, die auf jeder Seite mindestens 11,9 % aller Schnittpunkte hat. $\delta = 0,1231$ wird für $n \approx 2000$ erreicht.

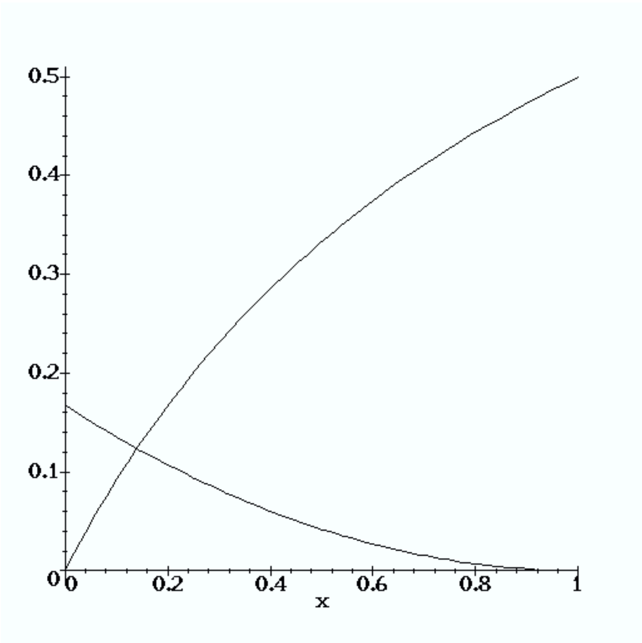


Abbildung 23: Die ansteigende Kurve zeigt für „große" Arrangements das Verhalten von δ_1 in Abhängigkeit von λ, die abfallende das von δ_2.

4.4 Δ-balancierende Geraden in einem Arrangement

Es ist jetzt geklärt, für welche δ die Existenz einer δ-zentralen Geraden garantiert werden kann. Die ursprüngliche Fragestellung ist jedoch, ob es eine Gerade gibt, die die Anzahl der *Regionen* möglichst „gerecht" aufteilt.

Der folgende Satz zeigt, daß die Regionen in einem Arrangement mindestens so gut aufgeteilt werden wie die Schnittpunkte.

n	$\lambda=\frac{3}{4}$	$\lambda=\frac{1}{2}$	$\lambda=\frac{1}{4}$	$\lambda=\frac{1}{5}$	$\lambda=\frac{1}{6}$	$\lambda=\frac{1}{8}$	$\lambda=\frac{1}{9}$	$\lambda=\frac{1}{10}$
6	0,333	0,167	/	/	/	/	/	/
12	0,389	0,267	0,083	0,035	/	/	/	/
20	0,406	0,296	0,136	0,095	0,066	0,026	0,012	/
30	0,414	0,310	0,159	0,121	0,094	0,057	0,044	0,033
50	0,420	0,319	0,176	0,140	0,114	0,080	0,068	0,058
100	0,424	0,327	0,189	0,154	0,129	0,096	0,084	0,075
1000	0,428	0,333	0,199	0,165	0,142	0,110	0,099	0,089
2000	0,428	0,333	0,199	0,166	0,142	0,110	0,099	0,090
$\frac{\lambda}{1+\lambda}$	0,429	0,333	0,200	0,167	0,143	0,111	0,100	0,091

Tabelle 2: Werte von δ_1 in Abhängigkeit von λ und n, also für den Fall, daß die größte Parallelenschar $k \geq \lambda n$ Geraden enthält.

n	$\lambda=\frac{3}{4}$	$\lambda=\frac{1}{2}$	$\lambda=\frac{1}{4}$	$\lambda=\frac{1}{5}$	$\lambda=\frac{1}{6}$	$\lambda=\frac{1}{8}$	$\lambda=\frac{1}{9}$	$\lambda=\frac{1}{10}$
6	/	0,019	0,081	0,099	0,111	/	/	/
12	0,003	0,031	0,086	0,101	0,111	0,125	0,129	0,133
20	0,006	0,035	0,089	0,103	0,113	0,125	0,130	0,133
30	0,007	0,037	0,091	0,104	0,114	0,126	0,130	0,134
50	0,009	0,039	0,092	0,105	0,114	0,127	0,131	0,134
100	0,009	0,040	0,093	0,106	0,115	0,127	0,131	0,135
1000	0,010	0,042	0,094	0,107	0,116	0,128	0,132	0,135
2000	0,010	0,042	0,094	0,107	0,116	0,128	0,132	0,135
$\frac{(1-\lambda)^2}{6}$	0,010	0,042	0,094	0,107	0,116	0,128	0,132	0,135

Tabelle 3: Werte von δ_2 in Abhängigkeit von λ und n, also für den Fall, daß die größte Parallelenschar $k < \lambda n$ Geraden enthält.

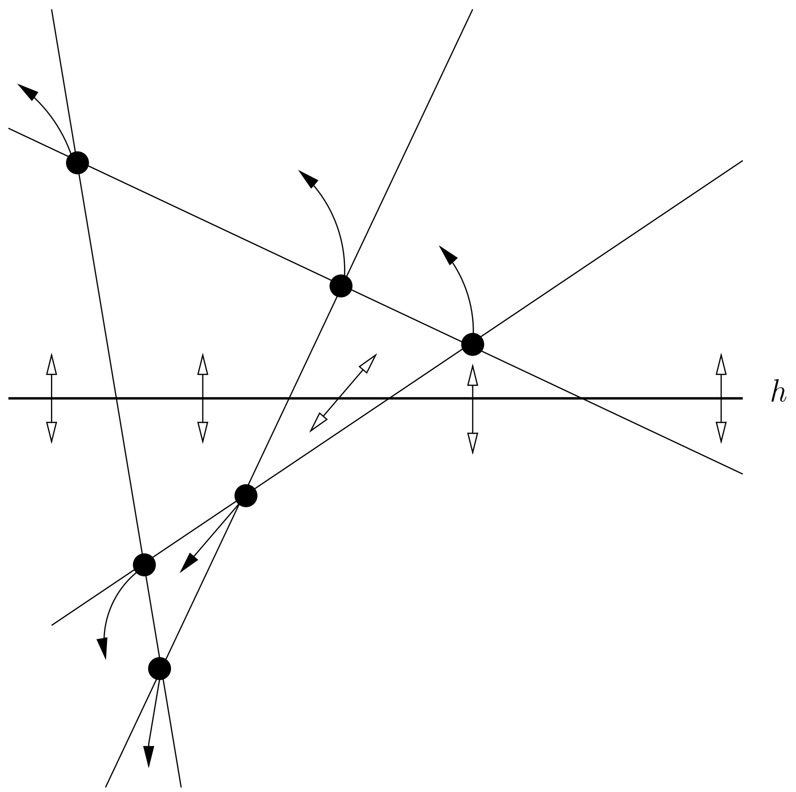


Abbildung 24: Regionen die nicht von h begrenzt sind, werden mit Schnittpunkten identifiziert. Von h begrenzte Regionen treten paarweise auf.

Satz 4.7 *Eine δ-zentrale Gerade h in einem Arrangement $\mathcal{A}$ mit $n \geq 4$ Geraden, in dem sich keine 3 Geraden in einem Punkt schneiden, ist auch δ-balancierend.*

Beweis. h sei eine δ-zentrale Gerade im Arrangement $\mathcal{A}$, und $\mathcal{A}$ sei so gedreht, daß h waagerecht verläuft. Falls es dann weitere waagerechte Geraden gibt, werden diese geringfügig verschoben; dadurch ändert sich nichts an der Anzahl der Schnittpunkte und Regionen. Dann ist oberhalb von h der tiefste Punkt einer Region, die nicht von h selber begrenzt wird, der Schnittpunkt von zwei Geraden des Arrangements. Analog ist unterhalb von h der höchste Punkt jeder solchen Region Schnittpunkt von zwei Geraden des Arrangements. Jede nicht von h begrenzte Region kann also mit einem Schnittpunkt identifiziert werden. Die von h begrenzten Regionen, die mit keinem Schnittpunkt korrespondieren, treten über und unter h gepaart auf; es gibt also auf jeder Seite von h gleich viele davon (siehe Abbildung 24).

Seien m und m' die Anzahl der Schnittpunkte sowie r und r' die Anzahl der Regionen oberhalb bzw. unterhalb von h, und $m \leq m'$. Da h δ-zentral ist, ist $m \geq \delta(m + m') = \delta m_h$. Außerdem bezeichne l die Anzahl der von h begrenzten Regionen über bzw. unter h.

Da $\delta \leq \frac{1}{2}$ ist, gilt $\delta m_h + l \geq \delta(m_h + 2l)$, und somit ist

$$\Delta = \frac{r}{r + r'} = \frac{m + l}{m + m' + 2l} \geq \frac{\delta m_h + l}{m_h + 2l} \geq \delta \quad .$$

h ist also δ-balancierend. □

Korollar 4.8 *In einem Arrangement mit $n \geq 5$ Geraden, in dem sich keine 3 Geraden in einem Punkt schneiden, gibt es immer eine Δ-balancierende Gerade mit $\Delta \geq 0,071$.*

Beweis. Das folgt sofort aus Satz 4.5 und Satz 4.7. □

5 Partielle Arrangements

In diesem Kapitel wird versucht, die bei der Untersuchung von Arrangements verwendete Vorgehensweise zu verallgemeinern. Das Ziel ist es, damit Aussagen über die Existenz zentraler und balancierender Geraden in partiellen Arrangements zu gewinnen.

Da nur Ausschnitte eines Arrangements betrachtet werden, sind die „Geraden“ jetzt eigentlich nur noch *Geradensegmente*, der einfacheren Lesbarkeit halber soll jedoch weiter von Geraden gesprochen werden. Das Problem für partielle Arrangements ist, daß in einem Ausschnitt die Parallelität von Geraden keine transitive Eigenschaft ist. Wenn $f \| g$ bedeutet, f und g haben keinen gemeinsamen Schnittpunkt, so kann es drei Geraden f, g, h innerhalb eines partiellen Arrangements geben, so daß $f \| g$ und $g \| h$, aber f und h schneiden sich. Ein Beispiel dafür ist in Abbildung 25 zu sehen. Es hat jedoch zuerst den Anschein, als könnte (mit einem Verlust an Güte) ein Δ gefunden werden, für das die Existenz einer Δ-balancierenden Geraden auch in einem partiellen Arrangement gesichert werden kann.

Definition 5.1 *Eine* Parallelenschar $\mathcal{P}$ *in einem partiellen Arrangement* $\mathcal{A}^*$ *sei eine Menge sich ausdrücklich paarweise nicht schneidender Geraden. Zu einer gegebenen Gerade* g *sei die* Pseudoparallelenschar $\mathcal{Q}_g$ *die Menge aller Geraden, die* g *nicht schneiden, untereinander aber Schnittpunkte haben dürfen.*

5.1 Von δ-zentralen zu Δ-balancierenden Geraden in partiellen Arrangements

Der Übergang von δ-zentralen zu Δ-balancierenden Geraden wird für partielle Arrangements etwas schwieriger, denn es kann nun auch Regionen geben, die mit keinem Schnittpunkt korrespondieren, und trotzdem *nicht* von der δ-zentralen Geraden h begrenzt sind (siehe Abbildung 26).

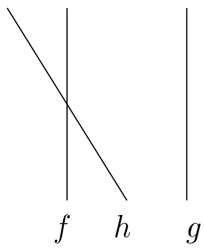


Abbildung 25: $f \| g$ und $g \| h$, aber f und h schneiden sich.

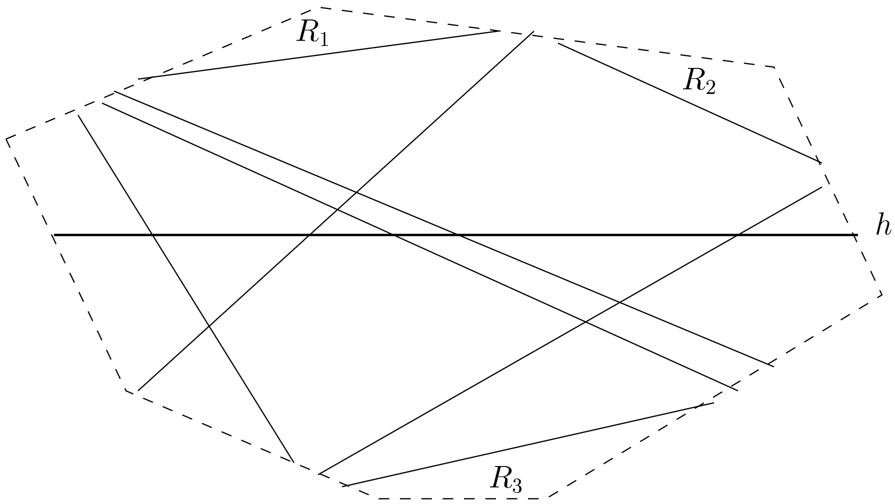


Abbildung 26: R_1, R_2 und R_3 korrespondieren mit keinem Schnittpunkt und werden nicht von h begrenzt.

Wenn es in einem partiellen Arrangement eine δ-zentrale Gerade h gibt, hilft der folgende Satz, auf das Verhältnis, in dem die Regionen aufgeteilt werden, zu schließen.

Satz 5.1 *In einem partiellen Arrangement mit n Geraden und m Schnittpunkten, in dem sich keine 3 Geraden in einem Punkt schneiden, ist die Anzahl der Regionen $r = m + n + 1$.*

Bemerkung. Dieser Satz läßt sich z.B. mit vollständiger Induktion über n beweisen. Im hier durchgeführten Beweis wird ein ähnliches Argument wie im Beweis von Satz 4.7 verwendet.

Beweis. Ein beliebiges partielles Arrangement $\mathcal{A}^*$ läßt sich (ggf. durch geringfügiges Verschieben) so darstellen, daß es keine waagerechte Gerade gibt, die Zahl der Regionen oder Schnittpunkte ändert sich durch eine solche Verschiebung nicht. Außerdem können unbeschränkte Regionen von $\mathcal{A}^*$ so berandet werden, daß keine Schnittpunkte verloren gehen, die Konvexität erhalten wird und die Zahl der Regionen gleich bleibt. Damit ist der tiefste Punkt einer Region entweder der Schnittpunkt von 2 Geraden aus $\mathcal{A}^*$ oder die Stelle, an der das untere Ende einer Geraden auf den Rand von $\mathcal{A}^*$ trifft. Es werden nun wie folgt Regionen mit Schnittpunkten in Beziehung gesetzt, zur Veranschaulichung dient Abbildung 27.

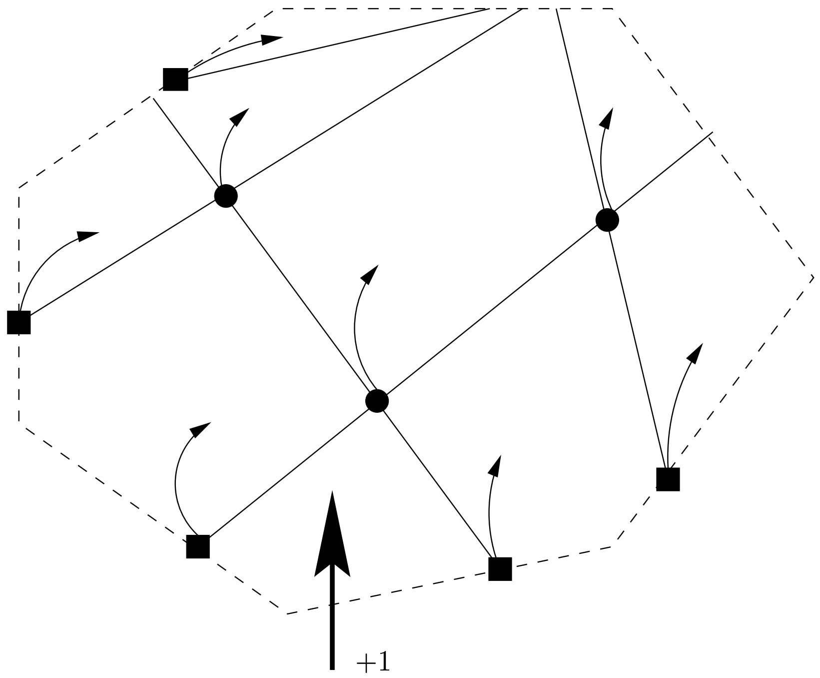


Abbildung 27: Skizze zum Beweis von Satz 5.1

Jeder Geraden g wird die Region, deren tiefster Punkt der Schnittpunkt von g mit dem Rand von $\mathcal{A}^*$ ist, zugeordnet. Da keine Gerade waagerecht verläuft, wird auf diese Weise jeder Geraden genau *eine* am Rand liegende Region zugeordnet, nämlich die an ihrem unteren Ende. Weiterhin wird jedem Schnittpunkt von zwei Geraden die Region zugeordnet, deren tiefster Punkt er ist. Damit wird jede Region bis auf die am tiefsten liegende mit einer Geraden oder einem Schnittpunkt identifiziert, und es ist $r = m+n+1$. □

Mit Hilfe dieser Beobachtung kann jetzt für eine δ-zentrale Gerade h auch das Verhältnis, in dem die Regionen durch sie aufgeteilt werden, bestimmt werden. Im folgenden sei mit m die Anzahl der Schnittpunkte, mit r die Anzahl der Regionen und mit n die Anzahl der Geraden auf einer Seite von h, mit m', r' und n' entsprechend die Anzahl der Schnittpunkte, Regionen und Geraden auf der anderen Seite von h bezeichnet, dabei sei $m \leq m'$.

Nach Voraussetzung ist h so gewählt, daß $m \geq \delta(m + m')$ ist, also

$$m + m' \leq \frac{m}{\delta}.$$

Je nach Beschaffenheit des partiellen Arrangements gibt es verschiedene Möglichkeiten, eine möglichst gute Abschätzung für $\Delta = \frac{r}{r+r'}$ zu finden.

Die Anzahl der Geraden rechts und links von h ist ausgewogen

Ausgewogen soll hier heißen, daß $n' = \gamma n$ und $\gamma \approx 1$. Dann ist

$$\Delta = \frac{m+n+1}{m+m'+n+n'+2} = \frac{m+n+1}{m+m'+(1+\gamma)n+2} \quad .$$

Da nach Voraussetzung $m + m' \leq \frac{m}{\delta}$ gilt, ist das

$$\geq \frac{m+n+1}{\frac{m}{\delta}+(1+\gamma)n+2}$$

1. Fall: $\gamma > 1$, also $n' > n$. Dann läßt sich der Ausdruck schreiben als

$$\frac{m+n+1}{\frac{1+\gamma}{\delta}\left(\frac{m}{1+\gamma}+\delta n+\frac{2\delta}{1+\gamma}\right)} > \frac{m+n+1}{\frac{1+\gamma}{\delta}(m+n+1)} = \frac{\delta}{1+\gamma} \quad .$$

2. Fall: $\gamma \leq 1$, also $n' \leq n$. Damit läßt sich der Ausdruck zusammenfassen zu

$$\frac{m+n+1}{\frac{2}{\delta}\left(\frac{m}{2}+\frac{1+\gamma}{2}\delta n+\delta\right)} \geq \frac{m+n+1}{\frac{2}{\delta}(m+n+1)} = \frac{\delta}{2} \quad .$$

Das heißt, Δ verschlechtert sich gegenüber δ ungefähr um den Faktor 2.

Es gibt im Verhältnis zur Anzahl der Geraden genug Schnittpunkte

Genug soll hier heißen, daß auf jeder Geraden im Durchschnitt β Schnittpunkte liegen, also insbesondere $\beta(n+n') = 2(m+m')$, und $\beta \geq 1$.

Es ist

$$\Delta = \frac{m+n+1}{m+m'+n+n'+2} = \frac{m+n+1}{m+m'+\frac{2}{\beta}(m+m')+2} = \frac{m+n+1}{\frac{\beta+2}{\beta}(m+m')+2} \quad ,$$

und nach Wahl von h ist das

$$\geq \frac{m+1}{\frac{\beta+2}{\beta}\frac{1}{\delta}m+2} \quad .$$

1. Fall: $\beta \geq 2$, also $\frac{\beta+2}{\beta} \leq 2$. Dann läßt sich der Ausdruck abschätzen durch

$$\frac{m+1}{\frac{2}{\delta}\left(\frac{\beta+2}{\beta}\frac{1}{2}m+\delta\right)} \geq \frac{m+1}{\frac{2}{\delta}(m+1)} = \frac{\delta}{2} \quad .$$

2. Fall: $\beta < 2$, also $\frac{\beta+2}{\beta} > 2$. Damit läßt sich der Term folgendermaßen abschätzen

$$\frac{m+1}{\frac{\beta+2}{\beta}\frac{1}{\delta}\left(m+2\delta\frac{\beta}{\beta+2}\right)} \geq \frac{m+1}{\frac{\beta+2}{\beta}\frac{1}{\delta}(m+1)} = \frac{\beta\delta}{\beta+2} \quad ,$$

und es ist für $\beta = 1$ beispielsweise $\Delta \geq \frac{\delta}{3}$. Für kleines β, also partielle Arrangements mit sehr wenigen Schnittpunkten, wird diese Abschätzung jedoch sehr schlecht.

5.2 Nur kleine Pseudoparallelenscharen

Im Kapitel 4.2.2 wurde gezeigt, daß in einem Arrangement, in dem jede Parallelenschar weniger als λn Geraden hat, eine $\frac{(1-\lambda)^2n^2-3(1-\lambda)n+2}{6n^2-18n}$-zentrale Gerade existiert. Die Überlegungen, die zu diesem Satz führten, können nicht ohne Einschränkungen auf partielle Arrangements übertragen werden. In der Abschätzung wird argumentiert: Wenn jede Parallelenschar weniger als λn Geraden hat, müssen auf jeder Geraden mindestens $(1-\lambda)n$ Schnittpunkte liegen. Das ist für partielle Arrangements nicht notwendig richtig. Wenn jedoch der Fall betrachtet wird, daß alle *Pseudoparallelenscharen* (vgl. Definition 5.1) weniger als λn Geraden haben, liegen tatsächlich auf jeder Geraden mindestens $(1-\lambda)n$ Schnittpunkte, und es ergibt sich der folgende Satz.

Satz 5.2 *In einem partiellen Arrangement, in dem alle Pseudoparallelenscharen weniger als λn Geraden haben und sich niemals 3 Geraden in einem Punkt schneiden, gibt es eine δ-zentrale Gerade h mit $\delta = \frac{(1-\lambda)^2n^2-3(1-\lambda)n+2}{6n^2-18n}$.*

□

5.3 Eine günstige Parallelenschar

Die Probleme, die bei der Untersuchung weiterer Fälle auftraten, führten zu einem Spezialfall, für den noch eine Aussage möglich ist.

5.3.1 Die Idee: Eine große Parallelenschar

In der Abschätzung, die zu Satz 4.2 führt, wird das folgende Argument benutzt: Wenn die Gerade g eine Gerade aus der Parallelenschar $\mathcal{P}$ schneidet, so schneidet sie *alle* Geraden von $\mathcal{P}$. Dies ist, wie aus Abbildung 25 ersichtlich wird, nun aber nicht mehr richtig: Es ist $\mathcal{P} = \{f, g\}$, h schneidet f, aber $h \| g$.

Eine andere Idee für diesen Fall, die zwar mißlingt, aber eine gute Anregung liefert, ist die folgende:

Gegeben sei ein partielles Arrangement $\mathcal{A}^*$ mit n Geraden, das eine Parallelenschar $\mathcal{P}$ mit $|\mathcal{P}| = \lambda n$ Geraden enthält, wobei λ nahe 1 sei.

Es werden nun zwei Sorten von Schnittpunkten in $\mathcal{A}^*$ unterschieden: Ein Schnittpunkt ist *günstig*, falls eine Gerade aus $\mathcal{P}$ beteiligt ist und *ungünstig*, falls beide beteiligten Geraden nicht aus $\mathcal{P}$ sind. Mit m_g werde die Anzahl der günstigen, mit m_u die der ungünstigen Schnittpunkte bezeichnet.

Es gibt $(1-\lambda)n$ Geraden, die nicht in $\mathcal{P}$ liegen, daher ist

$$m_u \leq \binom{(1-\lambda)n}{2} = \frac{1}{2}((1-\lambda)^2n^2 - (1-\lambda)n) \quad .$$

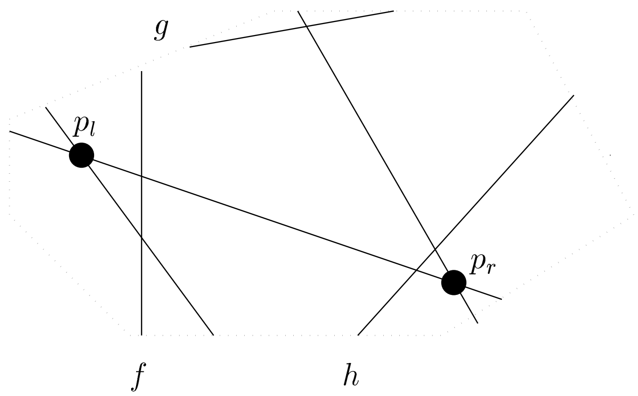


Abbildung 28: p_l und p_r liegen auf derselben Seite von g

Außerdem gilt

$$m_g \leq (1-\lambda)n\;\lambda n = (1-\lambda)\lambda n^2 \quad .$$

Die Geraden aus $\mathcal{P}$ partitionieren $\mathcal{A}^*$. Nun wird die Projektion von $\mathcal{A}^*$ auf die horizontale Achse betrachtet. Für jeden Schnittpunkt zwischen zwei Geraden wird das Gewicht 1 an die entsprechende Stelle auf der Achse aufgetragen. Anschließend wird die Stelle x bestimmt, die die Gewichte halbiert. Es werden dann die beiden Geraden h_l und h_r aus $\mathcal{P}$, die zu x benachbart sind, betrachtet. Da die Parallelenschar groß ist gibt es wenig ungünstige Schnittpunkte. Es können somit zwischen x und h_l bzw. x und h_r nicht viele Schnittpunkte liegen, so daß diese Geraden die Schnittpunkte ebenfalls „gerecht“ aufteilen.

Daß diese Herangehensweise falsch ist, macht folgendes Beispiel deutlich: Gegeben seien drei „parallele“ Geraden f, g und h, so daß f links von g und g links von h liegt. Dann kann es zwei Schnittpunkte p_l links von f und p_r rechts von h geben, so daß beide auf derselben Seite von g sind (siehe Abbildung 28). In diesem Fall wird x zwischen f und h liegen, und damit ist g eine zu x benachbarte Gerade. Das bedeutet also, daß eine neben x liegende Gerade die Schnittpunkte beliebig schlecht aufteilen kann.

5.3.2 Was sich retten läßt: Eine große günstige Parallelenschar

Um Fälle wie den eben behandelten auszuschließen, muß mehr gefordert werden, als nur eine große Parallelenschar. Das Problem an der Situation aus Abbildung 28 ist, daß g beide Endpunkte zwischen f und h hat. Die Argumentation kann zum Ziel führen, wenn eine große *günstige* Schar $\mathcal{P}^*$ betrachtet wird.

Definition 5.2 *Eine Parallelenschar $\mathcal{P}^* = \{h_1, \dots, h_k\}$ in einem partiellen Arrangement $\mathcal{A}^*$, bei der h_i links von h_{i+1} liegt, heißt* günstig, *falls gilt: Es gibt ein zusammenhängendes Randstück von $\mathcal{A}^*$, auf dem jede Gerade aus $\mathcal{P}^*$ genau einmal auftrifft.*

Anschaulich heißt das: In einer günstigen Schar werden Geraden vermieden, die zwischen nahe zusammenliegenden Randstücken verlaufen, die Geraden von $\mathcal{P}^*$ sind relativ „lang“. Man erhält so eine Schar beispielsweise, indem ein Schnitt durch $\mathcal{A}^*$ gelegt wird. Alle Geraden, die durchgeschnitten werden und in $\mathcal{A}^*$ parallel sind, sind (in der Reihenfolge von links nach rechts) in $\mathcal{P}^*$.

Für eine günstige Schar ist dann richtig: Liegt der Schnittpunkt p links von h_i, so liegt er auch links von h_{i+1}, und liegt er rechts von h_{i+1}, so auch rechts von h_i.

Gegeben sei ein partielles Arrangement $\mathcal{A}^*$ mit einer günstigen Parallelenschar $\mathcal{P}^*$ mit $|\mathcal{P}^*| = k = \lambda n$. Der Idee aus dem vorigen Abschnitt folgend, wird wieder $\mathcal{A}^*$ auf eine geeignete Achse projiziert und jeder Schnittpunkt von 2 Geraden durch ein Gewicht an der entsprechenden Stelle markiert. Eine geeignete Achse verläuft beispielsweise parallel zu dem Schnitt, durch den die günstige Schar gewonnen wurde. Die günstigen Schnittpunkte liegen somit auf den h_i, die ungünstigen dazwischen. Dann wird die Stelle x so gewählt, daß auf beiden Seiten von x gleichviele (± 1) Gewichte liegen, mit h_l und h_r werden wieder die Geraden aus $\mathcal{P}^*$ bezeichnet, die links bzw. rechts von x liegen. Weiter seien l und r die Anzahlen der Schnittpunkte zwischen x und h_l bzw. x und h_r.

Es kann nun passieren, daß alle ungünstigen Schnittpunkte zwischen x und h_l oder x und h_r liegen, aber auf jeden Fall ist $l \leq m_u$ und $r \leq m_u$. Es kann weiter die Anzahl der Schnittpunkte $s(h_l)$ auf h_l recht groß sein, ebenso auf h_r. Es ist aber sicher $s(h_l) \leq (1-\lambda)n$ und $s(h_r) \leq (1-\lambda)n$. Damit gilt:

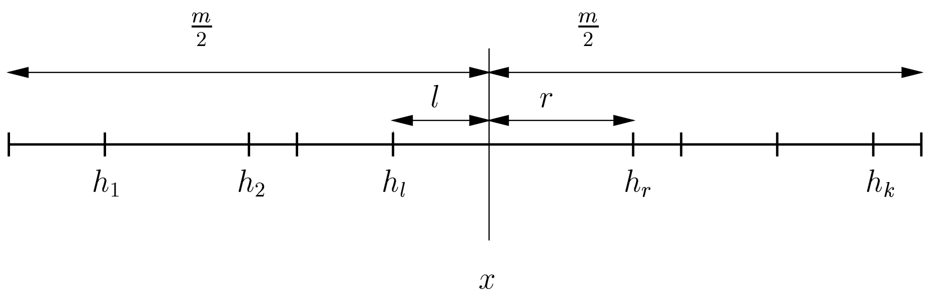


Abbildung 29: Aufteilen der Schnittpunkte

h_l teilt die Schnittpunkte $\frac{m}{2} - l - s(h_l) \quad : \quad \frac{m}{2} + l$ auf, und h_r teilt die Schnittpunkte $\frac{m}{2} + r \quad : \quad \frac{m}{2} - r - s(h_r)$ auf, siehe auch Abbildung 29.

Somit ist also

$$\delta_l = \frac{\frac{m}{2} - l - s(h_l)}{m - s(h_l)} \qquad \text{und} \qquad \delta_r = \frac{\frac{m}{2} - r - s(h_r)}{m - s(h_r)} \quad .$$

Dann gibt es eine δ-zentrale Gerade mit

$$\delta \geq \max\{\delta_l, \delta_r\},$$

wobei sich wegen $l \leq m_u \leq \frac{1}{2}((1-\lambda)^2 n^2 - (1-\lambda)n)$ und $s(h_l) \leq (1-\lambda)n$ abschätzen läßt:

$$\delta_l = \frac{\frac{m}{2} - l - s(h_l)}{m - s(h_l)} \geq \frac{\frac{m}{2} - \frac{1}{2}((1-\lambda)^2 n^2 - (1-\lambda)n) - (1-\lambda)n}{m + s(h_l)}$$

$$\geq \frac{\frac{m}{2} - (1-\lambda)^2 n^2 + (1-\lambda)n) - (1-\lambda)n}{m + (1-\lambda)n} = \frac{\frac{m}{2} - (1-\lambda)^2 n^2}{m + (1-\lambda)n} \quad .$$

Eine analoge Abschätzung läßt sich natürlich auch für δ_r durchführen. Um ein sinnvolles Ergebnis zu erhalten, darf es nicht zu wenig Schnittpunkte geben. Der Ausdruck ist positiv, wenn $m \geq 2(1-\lambda)^2 n^2$ ist. Es darf dabei nicht vergessen werden, daß

$$m = m_u + m_g \leq \frac{1}{2}((1-\lambda^2)n^2 - (1-\lambda)n)$$

ist, so daß insgesamt $2(1-\lambda)^2 n^2 \leq m \leq \frac{1}{2}((1-\lambda^2)n^2 - (1-\lambda)n)$ gelten muß. Ein solches m existiert, wenn $0 \leq (1-\lambda^2)n^2 - (1-\lambda)n - 4(1-\lambda)^2 n^2$ ist. Das ist gleichbedeutend mit $1 \leq n(5\lambda - 3)$. Die Abschätzung gilt somit nur für $\lambda > \frac{3}{5}$ und $n \geq \frac{1}{5\lambda - 3}$.

Satz 5.3 *In einem partiellen Arrangement $\mathcal{A}^*$ mit n Geraden und einer günstigen Parallelenschar $\mathcal{P}^*$ mit $|\mathcal{P}^*| = \lambda n$, in dem sich keine 3 Geraden in einem Punkt schneiden und gilt: $\lambda > \frac{3}{5}$, $n \geq \frac{1}{5\lambda-3}$ und $2(1-\lambda)^2 n^2 \leq m$, gibt es eine δ-zentrale Gerade h mit $\delta = \frac{\frac{m}{2} - (1-\lambda)^2 n^2}{m + (1-\lambda)n}$.*

Bemerkung. Es läßt sich eine bessere Gerade finden, wenn beispielsweise zusätzlich gefordert wird, daß es nur sehr wenige ungünstige Schnittpunkte gibt und daß die günstigen Schnittpunkte gleichmäßig auf die Geraden von $\mathcal{P}^*$ verteilt sind.

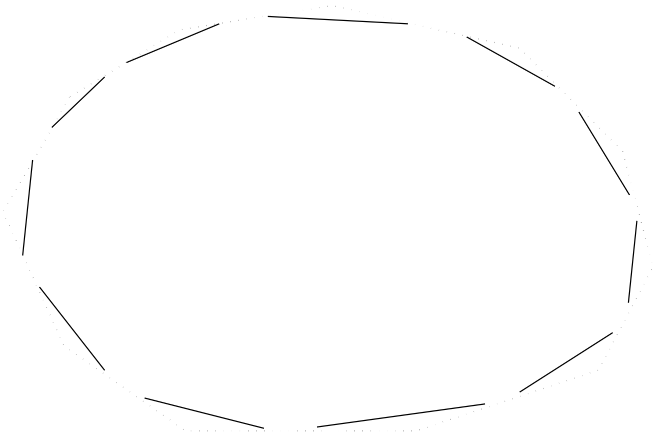


Abbildung 30: Jede Gerade ist zentral, aber keine balancierend

5.4 Die Problemfälle

Wie gesehen läßt sich für die Fälle, daß alle Pseudoparallelenscharen klein sind oder daß es eine große günstige Parallelenschar mit der richtigen Anzahl von Schnittpunkten gibt, eine δ-zentrale Gerade finden. Falls weiterhin genügend viele Schnittpunkte existieren, kann damit auch garantiert werden, daß es eine Δ-balancierende Gerade gibt.

Das folgende Beispiel zeigt, daß der Übergang von δ-zentraler zu Δ-balancierender Gerade für ein partielles Arrangement mit wenigen Schnittpunkten nicht möglich ist.

5.4.1 Erster Problemfall: Zu wenig Schnittpunkte

Beispiel. Ein partielles Arrangement, in dem jede Gerade zentral, aber keine balancierend ist:
Gegeben sei ein partielles Arrangement $\mathcal{A}^*$, dessen Grenze ein konvexes Polygon mit n Ecken ist. Die Geraden verlaufen nun jeweils zwischen zwei benachbarten Seiten des Polygons so, daß sich keine zwei Geraden schneiden (siehe Abbildung 30). Damit ist in $\mathcal{A}^*$ jede Gerade zentral (sie teilt die Schnittpunkte im Verhältnis $0 : 0$ auf), aber keine ist Δ-balancierend, denn jede Gerade hat auf einer Seite nur eine, auf der anderen alle restlichen Regionen.

An diesem Beispiel sieht man auch, daß es für die Existenz einer Δ-balancierenden Geraden nicht ausreicht, eine große Parallelenschar zu haben, denn hier gibt es keine Δ-balancierende Gerade, trotzdem *alle* Geraden paarweise parallel zueinander sind.

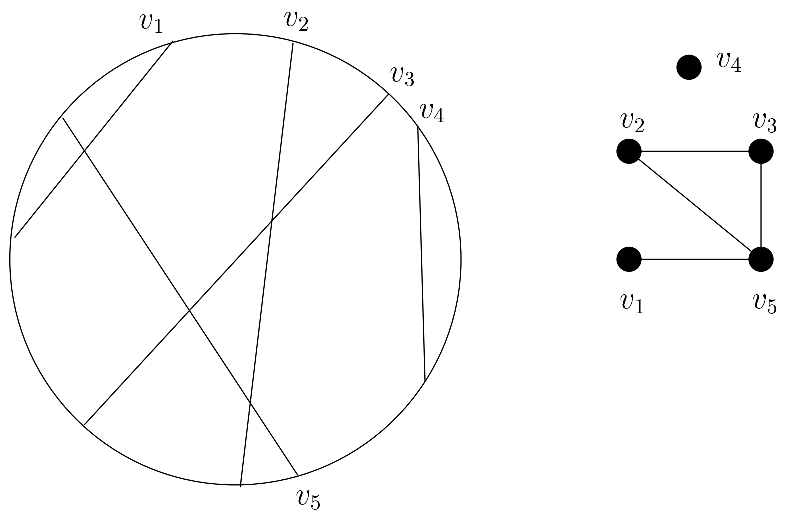


Abbildung 31: Ein Kreis mit Sehnen und der zugehörige Kreisgraph

Für den Fall, daß die einzelnen Parallelenscharen klein sind, es aber große Pseudoparallelenscharen und verhältnismäßig viele Schnittpunkte gibt, konnte kein δ gefunden werden, für das eine δ-zentrale Gerade sicher existiert. Daß eine solche Konstellation auftreten kann, liegt wiederum daran, daß Parallelität in partiellen Arrangements keine transitive Eigenschaft ist.

Die folgenden Überlegungen, mit Hilfe derer das Problem in einem anderen Kontext betrachtet werden kann, führen zu einem partiellen Arrangement, in dem es keine δ-zentrale Gerade gibt.

5.4.2 Kreisgraphen

Definition 5.3 *Gegeben sei ein Kreis mit Sehnen $v_1, \ldots, v_n$. Der* Kreisgraph $G = (V, E)$ *ist gegeben durch* $V = \{v_1, \ldots, v_n\}$ *und* $v_i v_j \in E \quad \Leftrightarrow \quad v_i$ *und v_j schneiden sich.*

Die Ecken von G sind also die Kreissehnen, und zwei Ecken sind verbunden, wenn die entsprechenden Sehnen sich schneiden. Ein Beispiel für einen Kreis mit Sehnen und den dazugehörigen Kreisgraphen ist in Abbildung 31 zu sehen.

Der Zusammenhang zwischen Kreisgraphen und partiellen Arrangements ist der folgende: Wenn es einen Kreis mit Sehnen gibt, so daß jede Sehne weder zentral noch balancierend ist, dann gibt es auch ein entsprechendes partielles Arrangement $\mathcal{A}^*$. Dazu muß nur ein gleichmäßiges Polygon mit ausreichend vielen Ecken betrachtet werden, die Kreissehnen sind dann genau die Geraden von $\mathcal{A}^*$.

Eine Pseudoparallelenschar $\mathcal{Q}_{v_i}$ in $\mathcal{A}^*$ ist die Menge aller Geraden, die v_i nicht schneiden. Das entspricht in G genau den Ecken, die nicht mit v_i verbunden sind, also $V \setminus N(v_i)$. Eine Parallelenschar in $\mathcal{A}^*$ entspricht in G einer Menge von Ecken, die paarweise nicht verbunden sind, also einer unabhängigen Menge.

Damit kommt man der Frage nach einem partiellen Arrangement mit kleinen Parallelenscharen und großen Pseudoparallelenscharen näher, indem ein Kreisgraph G mit einer kleinen Unabhängigkeitszahl $\alpha(G)$ und einem kleinen Grad $d(v)$ für jede Ecke v betrachtet wird. Das kleine $\alpha(G)$ entspricht kleinen Parallelenscharen. Wenn $d(v)$ für jedes v klein ist, ist $n - d(v) = |V \setminus N(v)|$ groß, das heißt, in $\mathcal{A}^*$ sind die Pseusoparallelenscharen groß.

Ein Beispiel eines solchen Kreisgraphen ist ein Graph G, der aus mehreren Zusammenhangskomponenten besteht, wobei jede der Komponenten eine Clique (also ein vollständiger Untergraph) ist. In diesem Graphen ist $\alpha(G)$ die Anzahl der Cliquen und $d(v)$ die Anzahl der Ecken in der entsprechenden Clique.

Näheres über Kreisgraphen findet sich beispielsweise in [7], die Erkennung von Kreisgraphen ist Thema von [4].

5.4.3 Zweiter Problemfall

Mit diesen Überlegungen kann nun ein Beispiel für ein partielles Arrangement konstruiert werden, in dem alle Geraden sowohl die Schnittpunkte als auch die Regionen beliebig schlecht aufteilen.

Beispiel. Ein partielles Arrangement mit n Geraden, in dem keine Gerade δ-zentral oder Δ-balancierend ist:

Ausgangspunkt der Konstruktion ist ein Kreisgraph G mit n Ecken, der aus $n^{\frac{2}{3}}$ Cliquen zu je $n^{\frac{1}{3}}$ Ecken besteht, wobei die Cliquen untereinander nicht verbunden sind. Für diesen Graphen ist $\alpha(G) = n^{\frac{2}{3}}$ und $|V \setminus N(v)| = n - d(v) = n - n^{\frac{1}{3}}$.

Das entspricht einem partiellen Arrangement, in dem die n Geraden aufgeteilt sind in $n^{\frac{2}{3}}$ Gruppen zu je $n^{\frac{1}{3}}$ Geraden. Die Geraden einer Gruppe schneiden sich untereinander paarweise, Geraden aus zwei verschiedenen Gruppen schneiden sich nicht. Ein solches partielles Arrangement kann beispielsweise so aussehen wie in Abbildung 32 skizziert. Ein dicker Balken steht jeweils für eine Gruppe sich paarweise schneidender Geraden.

Wegen der im vorigen Abschnitt erläuterten Beziehung zwischen dem partiellen Arrangement und dem Kreisgraphen sind in jeder Pseudoparal-

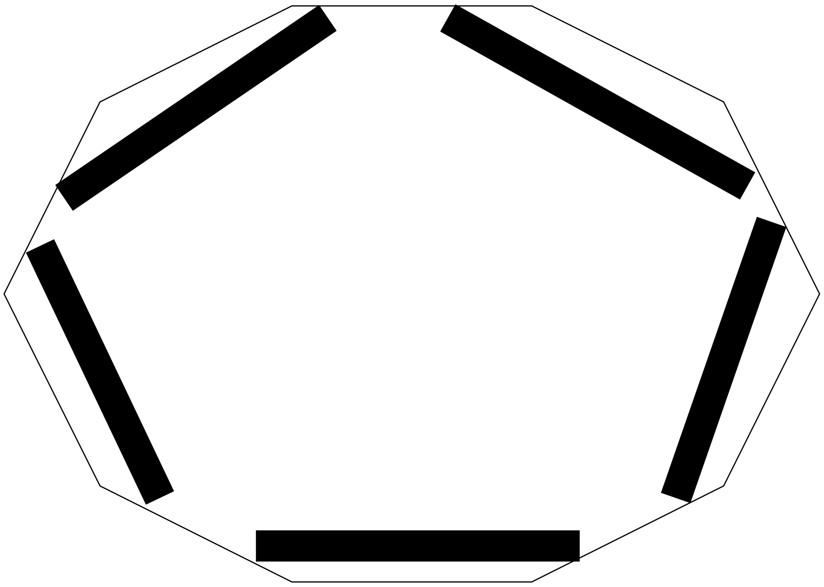


Abbildung 32: Ein $2n$-gon mit n Gruppen sich paarweise schneidender Geraden

lelenschar $n - n^{\frac{1}{3}}$ Geraden, eine Parallelenschar kann jedoch höchstens $n^{\frac{2}{3}}$ Geraden umfassen.

Innerhalb einer Gruppe gibt es $O(n^{\frac{2}{3}})$ Schnittpunkte, damit insgesamt $O(n^{\frac{4}{3}})$ Stück.

Es wird nun eine beliebige Gerade $h \in \mathcal{A}^*$ betrachtet. $\mathcal{A}^*$ wird durch h geteilt, und beide Teile sind wieder partielle Arrangements, für die nach Satz 5.1 $r = m + n + 1$ ist.

Nach Konstruktion können auf der äußeren Seite einer beliebigen Geraden h nur Geraden aus derselben Gruppe wie h und Schnittpunkte zwischen solchen Geraden liegen, also $O(n^{\frac{2}{3}})$ Schnittpunkte und $O(n^{\frac{1}{3}})$ Geraden. Damit liegen auf der äußeren Seite $O(n^{\frac{2}{3}})$ Regionen.

Auf der inneren Seite von h liegen alle Schnittpunkte zwischen Geraden aus anderen Gruppen, und das sind $O(n^{\frac{4}{3}})$ Stück (also quadratisch so viele wie auf der äußeren Seite), sowie $O(n)$ Geraden. Auf der inneren Seite von h liegen dann $O(n^{\frac{4}{3}})$ Regionen, und das sind *quadratisch* so viele wie auf der äußeren Seite. Also gibt es in diesem partiellen Arrangement weder eine δ-zentrale noch eine Δ-balancierende Gerade.

Damit zerschlägt sich die Hoffnung, in jedem partiellen Arrangement eine Δ-balancierende Gerade zu finden, und der Versuch, auf diese Weise einen guten Suchbaum für die Suche in einem Arrangement zu erhalten, scheitert.

6 Fazit und Ausblick

Durch Verallgemeinerung der Theorie für die eindimensionale Suche konnte gezeigt werden, daß für ein d-dimensionales gewichtetes Gitter die Pfadlänge $p(T_0)$ eines optimalen Suchbaumes T_0 die Ungleichung

$$\mathcal{H} \leq p(T_0) \leq \mathcal{H} + 2d$$

erfüllt, wobei $\mathcal{H}$ die Entropie der Wahrscheinlichkeitsverteilung bezeichnet. Weiter wurde gezeigt, daß die Konstruktion eines solchen Suchbaumes mit dynamischem Programmieren $O(dn^{2d+1})$ Zeit benötigt.

Dabei bleibt jedoch noch offen, ob die Wurzeln der Teilbäume möglicherweise gewisse Monotonieeigenschaften erfüllen. Mit deren Hilfe könnte es möglich sein, die Zeitkomplexität der Konstruktion ähnlich wie in Kapitel 1.7 zu senken.

Es ist weiterhin bewiesen worden, daß für die Pfadlänge eines optimalen Suchbaumes eines Arrangements von d Parallelenscharen mit gewichteten Regionen dieselben Schranken gelten wie im Fall des d-dimensionalen Gitters.

Eine weiterführende Fragestellung ist, inwieweit sich diese Schranken verbessern lassen, wenn für das Arrangement bekannt ist, welche der möglichen Regionen nicht auftreten.

Bei der Untersuchung von Arrangements kam ich zu Ergebnissen, die von der Größe des Arrangements abhängen. Ich konnte zeigen, daß es in jedem Arrangement von $n \geq 5$ Geraden eine $0,071$-zentrale Gerade gibt. Für wachsendes n kann die Existenz einer annähernd $0,123$-zentralen Geraden garantiert werden. Außerdem wurde gezeigt, daß δ-zentrale Geraden in Arrangements auch δ-balancierend sind.

Darauf aufbauend könnten nun Strategien zum Auffinden dieser Geraden entwickelt werden.

Eine sich anschließende Frage ist, ob und wenn ja wie sich die Konzepte auf den d-dimensionalen Raum erweitern lassen. Gesucht sind dann zentrale und balancierende Hyperebenen.

Da in partiellen Arrangements Parallelität keine transitive Eigenschaft ist, war es in diesem Fall schwieriger, zentrale und balancierende Geraden zu erhalten. Ich konnte zeigen, daß es in partiellen Arrangements ohne große Pseudoparallelenscharen immer eine δ-zentrale Gerade gibt, wobei δ von der Mächtigkeit der größten Pseudoparallelenschar abhängt.

Für den Fall, daß es eine große günstige Parallelenschar mit vielen Schnittpunkten gibt, konnte ebenfalls ein δ bestimmt werden.

Außerdem habe ich gezeigt, daß eine δ-zentrale Gerade in einem partiellen Arrangement mit mindestens so vielen Schnittpunkten wie Geraden Δ-balancierend ist, wobei Δ höchstens um den Faktor 3 schlechter als δ ist.

Das Beispiel für ein partielles Arrangement ohne balancierende Gerade ist so konstruiert, daß die „innere" Region sehr viele begrenzende Geraden hat. Möglicherweise gibt es bessere Ergebnisse, wenn nur partielle Arrangements betrachtet werden, in denen dieser Fall ausgeschlossen ist.

Weiter könnte untersucht werden, wie ein Arrangement beschaffen sein muß, damit es in den partiellen Arrangements balancierende Geraden gibt.

Auf den ersten Blick mag das Schlußergebnis, daß es partielle Arrangements ohne balancierende Gerade gibt, das Resultat der Arbeit etwas unbefriedigend erscheinen lassen.

Wirft man jedoch einen zweiten und dritten Blick auf die in der Arbeit verwendeten Methoden und die erzielten Ergebnisse, so stellt sich dieser Eindruck als unbegründet heraus.

Es wurde ein Verfahren angegeben, wie ein optimaler Suchbaum für die Suche in einem gewichteten Arrangement konstruiert werden kann, und für die gewichtete Pfadlänge ist eine scharfe obere Schranke bewiesen worden. Für Arrangements wurden gute Abschätzungen für δ angegeben, so daß die Existenz einer δ-balancierenden Geraden gesichert werden kann. Die bei der Betrachtung von partiellen Arrangements herausgearbeiteten Probleme bieten vielfältige Ansatzpunkte für weitere Untersuchungen.

Literatur

[1] Martin Aigner: *Diskrete Mathematik.* Vieweg Verlag, 1996.

[2] Martin Aigner: *Combinatorial Search.* Wiley-Teubner series in computer science, 1988.

[3] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf: *Computational Geometry.* Springer-Verlag, 1997.

[4] Csaba P. Gabor, Kenneth J. Supowit and Wen-Lian Hsu: *Recognizing circle graphs in polynomial time.* Journal Of The ACM 36, 435-473, 1989.

[5] A. M. Garsia and M. L. Wachs: *A new algorithm for minimum cost binary trees.* SIAM Journal On Computing 6, 622-642, 1977

[6] E. N. Gilbert and E. F. Moore: *Variable-length binary encodings.* The Bell System Technical Journal 38, 933-967, 1959.

[7] M. Golumbic: *Algorithmic graph theory and perfect graphs.* Academic Press (New York), 1980.

[8] T. C. Hu and A. C. Tucker: *Optimum computer search trees.* SIAM Journal On Applied Mathematics 21, 514-532, 1971.

[9] David A. Hufman: *A method for the construction of minimum-redundancy codes.* Proceedings Of The I.R.E. 40, 1098-1101, 1952.

[10] Kenneth E. Iverson: *A programming language.* Wiley, 1962.

[11] D. E. Knuth: *Optimum binary Search Trees.* Acta Informatica 1, 14-25, 1971.

[12] Kurt Mehlhorn: *Sorting and Searching.* Springer-Verlag, 1984.

[13] Franco P. Preparata: *Planar point location revisited.* International Journal Of Foundations Of Computer Science 1, 71-86, 1990.