Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik I

**Anselm Martin Hoffmeister**

# Dominierende Mengen in Einheitskreisgraphen

**10. November 2007**

Seminararbeit im WS 2007/2008

**Zusammenfassung**

Im Paper [3] von Ambühl, Erlebach, Mihal'ák und Nunkesser wird ein Algorithmus vorgestellt, der in Einheitskreisgraphen in polynomieller Zeit eine Approximation konstanten Faktors findet. Unter Ausnutzung der besonderen geometrischen Eigenschaften kann für dieses NP-harte Problem eine Näherung der optimalen Lösung, die nicht schlechter als das 89-fache des Optimums ist, sowie eine 3-Approximation des bei Funkdatenübertragung interessanten Minimum-Weight Forwarding-Set-Problems angegeben werden.

# Inhaltsverzeichnis

**1 Einführung und Motivation** **2**

**2 Algorithmen für MDS, MWDS, MCDS, MWCDS** **4**

**3 Der Algorithmus** **5**
3.1 Zerlegung in Teilprobleme . . . . . 5
3.2 Lösung des Teilproblems für ein kleines Quadrat . . . . . 5
3.3 Fallunterscheidung für die Approximation des kleinen Quadrats 6
3.4 Annahme Sonderfall: Optimale Lösung enthält keine Löcher . 7
3.5 Annahme Sonderfall: Ein Loch, nicht alle Kreise am Rand . . 7
3.6 Annahme: One-hole-Lösung . . . . . 7
3.7 Algorithmus "Strip Search" . . . . . 8
3.8 Annahme: Multi-hole-Lösung . . . . . 10
3.9 Darstellung in Pseudo-Code . . . . . 12

**4 Gültigkeit des Algorithmus** **16**
4.1 Strip-Search . . . . . 16
4.2 One-Hole-Lösungen . . . . . 17
4.3 Multi-Hole-Lösungen . . . . . 18
4.4 Null-Hole- und entartete One-Hole-Lösungen . . . . . 19
4.5 Zusammenfügen der Teillösungen . . . . . 20

**5 Laufzeit** **21**
5.1 Strip-Search . . . . . 21
5.2 One-Hole/Multi-Hole . . . . . 21
5.3 Zusammenfügen der Teillösungen . . . . . 21

**6 Güte des Algorithmus** **22**

**7 Der Übergang zur verbundenen Knotenmenge** **23**

**8 Fazit** **24**

# 1 Einführung und Motivation

Betrachtet man Funkzellen in Ad-Hoc-Netzwerken in einem einfachen Modell, so nimmt man oft einige Abstrahierungen vor. Sei es, dass alle Sender gleichzeitig Empfänger sind (was oft genug, aber nicht immer zutrifft), dass alle Stationen mit gleicher Stärke senden, und vor allem, dass alle Sender die gleiche Reichweite haben: Abschattungen, Wettereinflüsse und Betonwände werden in Modellen oft ausgeblendet.

Ein Ad-Hoc-Netzwerk zeichnet sich dadurch aus, dass die Stationen nicht nach strategischen Gesichtspunkten für Datenübertragung positioniert werden. Es existiert eine Menge Sender mit einer Aufgabenstellung, Punkt-zu-Punkt-Datenübertragung zwischen zwei beliebigen, nicht notwendig in direkter Nachbarschaft liegenden Sendern oder Flutung (Broadcast) zu ermöglichen[2].

Zur Modellierung kann man sich die Sender als Kreise von Radius 2 in der Ebene vorstellen. Eine bidirektionale Kommunikation oder ein Broadcast ist genau dann möglich, wenn eine Zielstation sich innerhalb des 2-Kreises der Sendestation befindet. In einer leicht variierten Darstellung verwendet man Kreise von Radius 1, Einheitskreise genannt, und ändert das Kriterium der Erreichbarkeit auf die Existenz eines Schnitt- oder Berührpunkts. Man sagt statt "A erreicht B" auch "A *dominiert* B".

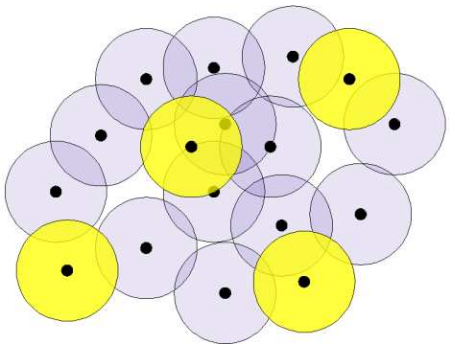

Abbildung 1: Anordnung von Kreisen mit einer dominierenden Menge.

Aus dieser Darstellung lässt sich ein Einheitskreisgraph gewinnen, indem man alle Stationen als Knoten darstellt; zwischen zwei Knoten existiert jeweils eine (ungerichtete) Kante, wenn diese in Reichweite zueinander stehen. Einen Menge Knoten $D \subseteq V$ nennt man eine dominierende Knotenmenge[1], englisch *"dominating set"*, wenn für alle Knoten in $V$ von einem Knoten in $D$ dominiert werden, formal in einem Graphen $G = (V, E), D \subseteq V$:

$$D \text{ dominating set} \Leftrightarrow \forall v \in V \setminus D \quad \exists w \in D : \quad v\bar{w} \in E$$

[1]Die Verwendung der englischen Begriffe scheint auch in deutschen Publikationen üblich zu sein, daher werde ich die deutschen Bezeichnungen bei der ersten Erwähnung jeweils einmal nennen, danach aber meist die englischen Begriffe verwenden.

Ein Knoten $d \in D$ wird *dominator* genannt. Das Problem, zu einem gegebenen Graphen $G$ das kleinste dominating set zu bestimmen – die Anzahl der Knoten in $D$ – heisst "minimum dominating set problem" (MDS) und ist NP-vollständig[1].

In einer Erweiterung des Problems werden Knotengewichte $w(v) \geq 0$ definiert, so dass ein dominating set minimalen Gewichts gefunden werde: MDS ist ein Spezialfall mit $w(v) = 1 \forall v \in V$ des "minimum-weight dominating set problem"[2] (MWDS).

Eine Variante dieser beiden Aufgabenstellungen ergibt sich mit der Forderung, alle Knoten aus $D$ sollen miteinander verbunden sein. Die Probleme "minimum connected dominating set" (MCDS)[3] und "minimum-weight connected dominating set" (MWCDS)[4] sind die offensichtlichen Varianten von MDS und MWDS.

[2]Dominierende Knotenmenge minimalen Gewichts oder Minimale gewichtete Knotenmenge

[3]Minimale verbundene dominierende Knotenmenge

[4]Minimale verbundene gewichtete Knotenmenge

## 2 Algorithmen für MDS, MWDS, MCDS, MWCDS

Das vorliegende Paper [3] beschäftigt sich insbesondere mit gewichteten Einheitskreisgraphen; Erkenntnisse über die Spezialfälle MDS und MCDS können daraus trivial hergeleitet werden.

Ausgehend von der Schwierigkeit der Probleme (Clark et al.[6]: MDS ist NP-hart; Liechtenstein[9]: MCDS ist NP-hart) ist eine in polynomieller Zeit laufende Approximation zu finden, die um nicht mehr als einen konstanten Faktor schlechter ist als die optimale Lösung.

Für die nicht gewichteten Sonderfälle ($w = 1$) existieren bereits derartige Algorithmen (MCDS: Marathe et al[10]). Der Ansatz von Hunt[8] beispielsweise basiert auf der "Shifting Strategy"([4], [8]): Man betrachtet die Ebene streifenweise und berechnet für Streifen oder Streifenausschnitte eine Approximation. Beim Zusammenfassen der Teilergebnisse kann die Güte der Approximation mit einem konstanten Faktor multipliziert werden (vergleiche [7]).

Der Algorithmus von Hunt ist allerdings nicht auf das gewichtete Problem MWDS erweiterbar, da eine der zentralen Annahmen besagt, dass in einem Quadrat der Ebene der Größe $k \times k$ ein optimales dominating set niemals ein Gewicht von mehr als $k^2$ haben kann. Offensichtlich gilt dies nicht, wenn die Knoten des Graphen gewichtet werden.

Auch andere Algorithmen ([5]) für MCDS lassen sich nur im Spezialfall $w = 1$ anwenden. Van Leeuwen ([11]) nutzt in seinem Algorithmus beispielsweise eine Annahme über beschränkte Dichte des zu betrachtenden Graphen in MDS und MCDS. Aussagen über Dichte in der Fläche lassen sich bei gewichteten Knoten wiederum nicht verallgemeinern.

Wang und Li[12] stellten einen Algorithmus für MWDS und MWCDS in Einheitskreisgraphen vor; die Approximationsgüte dieses Algorithmus war allerdigns nicht konstant und im schlechtesten Fall nicht besser als für allgemeine Graphen (die Besonderheit der Einheitskreise kam hier also nicht zum Tragen).

Ambühl, Erlebach, Mihal'ák und Nunkesser präsentieren im vorliegenden Paper erstmals einen Polynomialzeit-Algorithmus konstanter Approximationsgüte.

# 3 Der Algorithmus

Gegeben sei eine Menge von Kreisen $D$ mit Radius 1 in der Ebene mit jeweils einem zugeordneten Gewicht $w_d \geq 0 \ \forall d \in D$.

Der Algorithmus soll eine dominierende Menge $A$ von Kreisen ausgeben, so daß jeder Kreis entweder zu $A$ gehört oder mit mindestens einem Kreis aus $A$ einen gemeinsamen Punkt hat. Gefordert ist eine Approximation an die minimal gewichtete dominierende Menge über $D$.

## 3.1 Zerlegung in Teilprobleme

Der Weg zur Ausgabe einer Approximation besteht aus zwei Schritten.

Im ersten Schritt wird die Ebene in ein Raster von achsenparallelen Quadraten aufgeteilt. Es sollen für den Approximationsalgorithmus nur Quadrate betrachtet werden, in denen ein Kreismittelpunkt liegt.

Der Algorithmus setzt eine Quadratseitenlänge $\mu < 1$ voraus, sei im Folgenden $\mu = 0.999$. Bezeichne $S_{ij}$ das Quadrat zu

$$S_{ij} := \left\{(x, y) \in \mathbb{R}^2 : i\mu \leq x < (i+1)\mu, j\mu \leq y < (j+1)\mu\right\}, i, j \in \mathbb{Z}$$

Falls sich in $S_{ij}$ Kreismittelpunkte befinden (andernfalls wird dieses Quadrat vom Algorithmus ignoriert), bezeichne $D_{ij}$ die Menge der Einheitskreise mit Mittelpunkt in $S_{ij}$. Bezeichne $N(D_{ij})$ die Nachbarschaft von $D_{ij}$ ausserhalb $S_{ij}$:

$$N(D_{ij}) := \{d \in D \setminus D_{ij} : \exists e \in D_{ij} : \bar{de} \leq 2\}$$

Für jedes Quadrat wird separat eine Lösung $L_{ij}$ approximiert, also ein lokales dominating set ausgegeben. Die lokalen Approximationen werden zusammengefasst, die Vereinigung bildet offensichtlich ein dominating set auf $D$.

## 3.2 Lösung des Teilproblems für ein kleines Quadrat

Auf jedem der oben erwähnten Quadrate wird aufzählend ein Wert $w_{max}$ angenommen, so dass nur Kreise mit Gewicht $\leq w_{max}$ im Lösungsversuch vorkommen.

Der Startwert ist offensichtlich das Gewicht des leichtesten Kreisea in $D_{ij} \cup N(D_{ij})$, nacheinander wird dann der jeweils nächstschwerere Kreis herangezogen.

Findet sich für ein $w_{max}$ ein Kreis $d \in D_{ij}$ mit $w_d \leq w_{max}$[5], so wird $L_{ij} := \{d\}$ als Approximation für dieses Quadrat ausgegeben. Aufgrund der Einschränkung $\mu < 1$ dominiert $d$ alle Kreise in $D_{ij}$.

[5]Hier kann auch Gleichheit angenommen werden, "<" kann bei korrekter Ausführung des Algorithmus nicht erfüllt werden

Wenn kein Kreis mit $w_d = w_{max}$ in $D_{ij}$ gefunden wird, so bleiben zwei Möglichkeiten: Entweder kann die Approximation gar keine Lösung angeben, bei der nur Kreise mit $w_d \leq w_{max}$ beteiligt sind, oder alle Kreise, die zur Lösung gehören, befinden sich in $N(D_{ij})$. Dies wird sich im weiteren Verlauf erweisen.

Dazu werden alle Kreise $d \in D_{ij}$ sowie die Kreise $d \in N(D_{ij})$ mit $w_d > w_{max}$ gestrichen und in dieser Runde nicht weiter betrachtet.

Zur weiteren Betrachtung wird das verbleibende Szenario äquivalent modifiziert. Bereits bekannt ist die Tatsache, dass keiner der Kreise in $D_{ij}$ zur Lösung gehört; genau alle diese Kreise müssen aber dominiert werden. Dann ist das verwendete Modell (Einheitskreise; Nachbarschaft gilt für zwei Kreise mit mindestens einem gemeinsamen Punkt) äquivalent zur Vorstellung von Kreisen mit Radius 2 in $N(D_{ij})$, in denen die Kreismittelpunkte aus $D_{ij}$ genau dann liegen, wenn sie dominiert werden.

Zusätzlich wird dieses veränderte Modell skaliert, so dass wieder Einheitskreise betrachtet werden (Skalierung um jeweils Faktor 2 entlang der Einheitsachsen); das Quadrat $S_{ij}$ hat dann eine Seitenlänge von $\delta = \mu/2 < \frac{1}{2}$.

Für die Berechnung einer 2-Approximation des Optimums für dieses Quadrat $S_{ij}$ gibt es mehrere Methoden, deren Erfolg davon abhängt, wie die tatsächlich optimale Lösung aussieht. Da diese optimale Lösung nicht bekannt ist, müssen alle Möglichkeiten durchprobiert werden (zur Anzahl siehe Abschnitt 5).

### 3.3 Fallunterscheidung für die Approximation des kleinen Quadrats

Die optimale Lösung lässt sich in zwei Grundklassen unterteilen. Dazu muss die von den ausgewählten Kreisen abgedeckte Fläche in Bezug auf $S_{ij}$ untersucht werden. Gefragt ist nach der Anzahl der nicht abgedeckten Bereiche. Man stelle sich das Quadrat als Fenster vor und die Kreisflächen als davor hängende Pappscheiben: Betrachtet wird der restliche, durchsichtige Bereich. Zwei wesentliche Fälle sollen unterschieden werden: Es ist genau eine zusammenhängende freie Fläche vorhanden, an deren Rand alle ausgewählten Kreisscheiben in mindestens einem Punkt angrenzen ("One hole"), oder es sind mehrere freie Flächen vorhanden ("multi hole").

Alle anderen Möglichkeiten sind als Abarten zu betrachten: Ist kein Loch vorhanden, so könnte durch Entfernen eines Kreises eine One-Hole-Lösung geschaffen werden (dabei nicht mehr abgedeckte Punkte $p \in D_{ij}$ werden explizit in Kauf genommen und weiter unten behandelt). Ist nur ein Loch vorhanden, an das aber nicht alle Kreise angrenzen (es existiert ein Kreis, der keinen gemeinsamen Punkt mit dem Loch hat), so kann durch Entfernen eines oder zweier Kreise (wieder unter Inkaufnahme dann nicht mehr abgedeckter Punkte) der Multi-Hole-Fall hervorgerufen werden.

Da nicht bekannt ist, in welche Kategorie die optimale Lösung fällt,

werden alle Möglichkeiten durchprobiert. Dies ist für die Laufzeit eine Verschlechterung mit lediglich polynomialem Faktor.

### 3.4 Annahme Sonderfall: Optimale Lösung enthält keine Löcher

Um diesen Fall in eine "One-hole"- oder "Multi-hole"-Lösung umzuwandeln, werden nacheinander jeweils ein Kreis $d$ aus $N(D_{ij})$ und alle von $d$ dominierten Punkte entfernt und das dadurch erhaltene Szenario als "One-hole"- und als "Multi-hole"-Szenario (siehe unten) behandelt. Gleiches wird mit jeder möglichen Paarung, das sind $O(n^2)$ viele, aus zwei Kreisen wiederholt: Jeweils zwei Kreise werden aus $N(D_{ij})$ entfernt, die von diesen dominierten Punkte aus $D_{ij}$ werden entfernt und das erhaltene Szenario wird wieder als "One-hole"- und als "Multi-hole"-Szenario abgearbeitet.

### 3.5 Annahme Sonderfall: Ein Loch, nicht alle Kreise am Rand

Auch in diesem Fall wird abzählend versucht, eine "One-hole" oder "Multi-hole"-Lösung zu erzielen; die entsprechenden Schritte wurden schon im ersten Sonderfall (Abschnitt 3.4) unternommen.

### 3.6 Annahme: One-hole-Lösung

Wir nehmen an, die optimale Lösung für das betrachtete Szenario habe genau eine nicht abgedeckte Fläche, an deren Rändern jeder Kreis anliegt. Dabei sollen die Ränder des Quadrats, falls diese an die Lochfläche angrenzen, als Kreisscheiben mit Gewicht 0 behandelt werden - im Prinzip handelt es sich um Halbebenen, also Kreise mit unendlichem Radius - das ist hier aber nicht relevant.

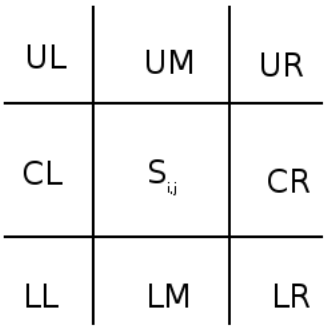


Abbildung 2: Die Aufteilung der Ebene rund um das Quadrat $S_{ij}$.

Die Ebene ausserhalb $S_{ij}$ sei durch die Verlängerung der Quadratseiten in acht Segmente geteilt. Diese sollen im Uhrzeigersinn *UM, UR, CR, LR, LM,*

*LL, CL, UL* heißen (Upper Left / oben links ...), die aus der Vereinigung der Segmente *UL, UM, UR* bestehende Halbebene heiße *UPPER* etc.[6]

Das eine Loch der optimalen Lösung wird von Kreisbögen und *Ecken* begrenzt. Der Algorithmus soll nun die oberste und die unterste Ecke, die von einem Kreis mit Mittelpunkt aus *CL* bzw. *CR* berührt wird, erraten - also durchprobieren. Kandidaten sind jeweils alle Kreise aus den betroffenen Regionen und ein jeweils zweiter Kreis (mit Mittelpunkt aus) *UPPER* bzw. $LOWER \bigcup CR$ bzw. *CL*. Dies sind offensichtlich polynomiell viele in $O(n^8)$ Möglichkeiten.

Die vier Kreise aus *CR, CL* sollen nun entfernt werden, ebenso alle von diesen Kreisen dominierten Punkte aus $D_{ij}$; die verbleibenden Punkte heißen $D'_{ij}$.

Sollten in *CR* oder *CL* keine Kreismittelpunkte liegen, kann ein Teil des Folgenden übersprungen werden.

Die Punkte in $D'_{ij}$ zerfallen in zwei Kategorien: Jene, die im fiktiven Einheitskreis[7] liegen, der durch den oberen linken und unteren linken Eckpunkt geht und dessen Mittelpunkt links des Quadrats liegt (bzw. äquivalent rechts) und alle anderen. Auf erstere wird der "Strip Search"-Algorithmus (Abschnitt 3.7) mit einem vertikalen Streifen und allen verbleibenden Kreisen in *LEFT* und *RIGHT* angewendet; auf letztere ebenfalls der "Strip Search"-Algorithmus, allerdings mit allen verbleibenden Kreisen aus *UPPER* und *LOWER* auf einem horizontalen Streifen.

Im Fall $d_{lu} = d_{ld}$ entfällt die Betrachtung eines Hörnchens, da dies Fläche 0 hat.

Die insgesamt bestmögliche zwischenzeitlich vorliegende Lösung wird zwischengespeichert.

## 3.7 Algorithmus "Strip Search"

Dieser Algorithmus setzt einen Streifen (eine Fläche zwischen zwei parallelen Geraden) voraus, in dem sich eine Menge Punkte $P$ befindet; o.B.d.A. sei dieser Streifen horizontal und durch die Geraden $y_U, y_L$ (konstant) begrenzt. Oberhalb und unterhalb des Streifens befinden sich Einheitskreise in den Mengen $D_U$ und $D_L$.

Gesucht ist eine optimale Abdeckung $D_{opt}$, d.h.

$$D_{opt} \subseteq (U \cup L), U \subseteq D_U, L \subseteq D_L,$$

$$\forall p \in P \exists d \in D_{opt} : d \text{ dominiert } p, \quad \sum_{d \in D_{opt}} w(d) \text{ minimal}$$

[6] Diese Festlegungen entsprechen der intuitiven Anordnung; die englischen Buchstabenkürzel und Bezeichnungen sind aus dem Paper übernommen.

[7] Dabei ist an die Skalierung zu denken: Dieser "Einheitskreis" entspricht einem Kreis mit Radius 2 im ursprünglichen Koordinatenraster. Dieser Kreis ist im allgemeinen nicht Teil der gegebenen Kreismenge

**Algorithmus 3.1** StripSearch

// Gegeben: Punktemenge $P$ sortiert als $p_0 \dots p_{n-1}$
// Gegeben: Kreismengen $U$ und $L$ (inkl. Halbebenen $U$ / $L$

// Initialisiere Tabelle $T_0$
**for all** $d_U \in U, d_L \in L$ **do**
  **if** $(d_U, d_L)$ dominiert $p_0$ **then**
    $T_0(d_U, d_L) := w(d_U) + w(d_L)$
    $U_0(d_U, d_L) := \{d_U\}$
    $L_0(d_U, d_L) := \{d_L\}$
  **else**
    $T_0(d_U, d_L) := \infty$
    $U_0(d_U, d_L) := \emptyset$
    $L_0(d_U, d_L) := \emptyset$
  **end if**
**end for**
// Tabellen für folgende Punkte aufstellen
**for** $i = 1$ **to** $n - 1$ **do**
  **for all** $d_U \in U, d_L \in L$ **do**
    **if** $(d_U, d_L)$ dominiert $p_i$ **then**
      CHOOSE $d'_U, d'_L$ WHERE $\min\{T_{p_{i-1}}(d'_U, d'_L) + \Delta(d_U, d'_U) \times w(d_U) + \Delta(d_L, d'_L) \times w(d_L)\}$
      // $\Delta(a, b) := 0$ wenn $a = b$, 1 sonst.
      $T_i(d_U, d_L) := T_{i-1}(d'_U, d'_L) + \Delta(d_U, d'_U) \cdot w(d_U) + \Delta(d_L, d'_L) \cdot w(d_L)$
      $U_i(d_U, d_L) := U_{i-1} \cup \{d_U\}$
      $L_i(d_U, d_L) := L_{i-1} \cup \{d_L\}$
    **else**
      $T_i(d_U, d_L) := \infty$
      $U_i(d_U, d_L) := \emptyset$
      $L_i(d_U, d_L) := \emptyset$
    **end if**
  **end for**
**end for**
CHOOSE $d_U, d_L$ WHERE $\min T_{n-1}(d_U, d_L)$
**if** $\min(T_{n-1}(d_U, d_L) = \infty$ **then**
  // Abdeckung aller Punkte ist mit den gegebenen Kreisen nicht möglich
**else**
  OUTPUT $T_{n-1}(d_U, d_L)$
  OUTPUT $U_{n-1}(d_U, d_L)$
  OUTPUT $L_{n-1}(d_U, d_L)$
**end if**

Eine Teilmenge $D_x \subseteq (U \cup L)$ heißt erfüllend, wenn sie alle Punkte in $P$ dominiert.

Man nennt einen Kreis "oben aktiv in $D_x$" für eine $x$-Koordinate $x_a$, wenn dieser Kreis unter allen Kreisen aus $D_x \cap U$ den Schnittpunkt $(x_a, y_a)$ mit geringstem Wert $y_a$ hat. Gibt es mehrere Kreise, die den Punkt $(x_a, y_a)$ berühren, so soll der Kreis mit am weitesten links liegendem Mittelpunkt betrachtet werden[8].

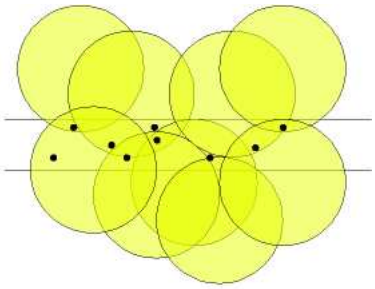

Abbildung 3: Szenario Strip-Search.

Die Halbebenen oberhalb bzw. unterhalb des Streifens sollen als Kugeln unendlichen Radius' und mit Gewicht 0 zu $U$ bzw. $L$ hinzugefügt werden.

Die Punkte $p_i \in P$ seien nach x-Koordinate monoton steigend (von links nach rechts) geordnet.

Nun werden von links nach rechts Kreise gesucht, die alle bis dato besuchten Punkte überdecken. Schrittweise wird die optimale Lösung ermittelt (siehe Algorithmus 3.1).

### 3.8 Annahme: Multi-hole-Lösung

Unter der Annahme, die optimale Lösung sei eine "Multi-hole"-Lösung, müssen ebenfalls noch Schritte durchgeführt werden.

Zuerst erfolgt die Feststellung, dass es ein Kreispaar $d_1, d_2$ gibt, das mindestens einen gemeinsamen Schnittpunkt hat und das $S_{ij}$ in zwei Bereiche mit jeweils mindestens einem "Loch" teilt. Als Sonderfall kann $d_2$ eine der Quadratseiten sein (und somit wie oben erwähnt den Kreis unendlichen Radius' verkörpern, der als Halbebene eine Quadratseite berührt).

Der Algorithmus soll für jede mögliche Kreispaarung

1. ein Kreis $d$ aus UPPER mit der unteren Quadratseite (LOWER, LEFT, RIGHT entsprechend), wobei der Abstand des Mittelpunkts von $d$ zu LOWER echt zwischen $\frac{\sqrt{15}}{4}$ und 1 beträgt

[8] Haben zwei solche Kreise dieselbe $x$-Koordinate des Mittelpunkts, so sind sie identisch (wg. Einheitskreis und $(x_a, y_a)$ auf Kreisrand mit $y_a$ minimal für gegebenes $x_a$)

2. ein Kreis aus UM und ein Kreis aus LOWER (LM+UPPER, CR+LEFT, CL+RIGHT entsprechend) mit einem Abstand der Kreismittelpunkte echt zwischen $\frac{\sqrt{14}}{4}$ und 1.

3. ein Kreis aus UL und ein Kreis aus LR (UR+LL entsprechend) mit der selben Anforderung an den Abstand

die folgenden Betrachtungen durchführen:

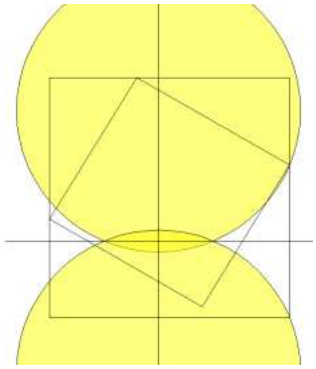

Abbildung 4: Skizze: Lage von $S_{ij}, d_1, d_2$ im rotierten Koordinatensystem.

Zuerst wird für das betrachtete Objekt ein neues Koordinatensystem definiert durch die beiden Kreisscheiben: Die beiden Kreismittelpunkte sollen auf der $y$-Achse zu liegen kommen, beide Schnittpunkte auf der $x$-Achse (für 1. nur die letztere Bedingung). Das minimale, $S_{ij}$ umschliessende achsenparallele Quadrat teile die Ebene analog zu oben in die Bereiche $UL', UM', UPPER'$ etc. Die von $d_1$ und $d_2$ frei gelassene Lochfläche links der $y$-Achse heisse[9] $LH$, analog sei $RH$ die Lochfläche rechts der $y$-Achse.

Ähnlich zum "One-hole"-Fall gibt es einen Bereich $L$ (entsprechend $R$), der in der optimalen Lösung nur von Kreisen aus $CL'$ abgedeckt werden kann. Hier wird eine erneute Fallunterscheidung nötig:

1. Die optimale Lösung beinhaltet keine Kreise aus $CL$ (entsprechend $CR$): Keine Behandlung dieser Kreise ist nötig.

2. Die optimale Lösung beinhaltet exakt einen Kreis aus $CL$ (entsprechend $CR$). Aufzählend wird jeder Kreis aus $CL$ bzw. $CR$ als dieser Kreis angenommen und fortgefahren.

3. Die optimale Lösung beinhaltet zwei oder mehr Kreise aus $CL$ ($CR$).

In Fall 1 und 2 können die nicht von $d_1$, $d_2$ und den Kreisen in $CR/CL$ bereits dominierten Punkte in $S_{ij}$ mit dem "Strip-Search"-Algorithmus (Algorithmus 3.1) bearbeitet werden.

[9] für *left hole*, linkes Loch

Im Fall 3 muss genauso wie im “One-Hole”-Fall der Kreis, auf dem der unterste und der oberste Eckpunkt eines Lochs, die jeweils von einem Kreis aus $CL$ ($CR$) berührt werden, liegen, konstruiert werden - und genauso wie oben wird “Strip-Search” (Algorithmus 3.1) einmal horizontal und einmal vertikal auf $S_{ij}$ angewandt.

### 3.9 Darstellung in Pseudo-Code

Zur Verbesserung der Übersichtlichkeit wird der Algorithmus nicht komplett en block dargestellt, sondern nur in wesentlichen Zügen, die Details finden sich textuell in den vorhergehenden Kapiteln.

Schon anhand der stark verschachtelten Schleifen im Hauptalgorithmus ist ersichtlich, dass die Laufzeit eine hohe polynomielle Komplexität haben wird.

Die Unterfunktionen ONEHOLEERGEBNIS und MULTIHOLEERGEBNIS zeigen ebenfalls mehrere Schleifen auf.

**Algorithmus 3.2** Approximationsalgorithmus für MWDS

// Eingabe: $n$ gewichteten Einheitskreise $d_i \in D$
**for all** $S \in \{S_{ij} := \left\{(x,y) \in \mathbb{R}^2 : i\mu \leq x < (i+1)\mu, j\mu \leq y < (j+1)\mu\right\}\}_{ij}$ **do**
  // Vorausgesetzt $\exists k \in \{1 \dots n\} : d_k \in S_{ij}$
  KOORDINATENTRANSFORMATION // wie im Text erklärt
  // Speichere lokal relevante Kreise:
  $A := d \in D$: $d$ dominiert mindestens einen Punkt in $S_{ij}$
  **for all** $d \in A$, sortiert monoton steigend in $w(d)$ **do**
    Approximation_$S_{ij} := \infty$
    **if** $\exists d_a \in S_{ij}$ mit $w(d_a) = w(d)$ **then**
      // Es kann keine lokal bessere Lösung folgen
      **if** $(w(d_a) <$ Approximation_$S_{ij}$ **then**
        Approximation_$S_{ij} := w(d_a)$
      **end if**
      break;
    **else**
      $B := \{d \in A : w(d) \leq w(d_a)\}$
      $t :=$ONEHOLEERGEBNIS$(B)$
      **if** $t <$ Approximation_$S_{ij}$ **then**
        Approximation_$S_{ij} := t$
      **end if**
      $t :=$MULTIHOLEERGEBNIS$(B)$
      **if** $t <$ Approximation_$S_{ij}$ **then**
        Approximation_$S_{ij} := t$
      **end if**
      **for all** $d_x \in B$ **do**
        $t :=$ONEHOLEERGEBNIS$(B \setminus \{d_x\})$
        **if** $t <$ Approximation_$S_{ij}$ **then**
          Approximation_$S_{ij} := t$
        **end if**
        $t :=$MULTIHOLEERGEBNIS$(B \setminus \{d_x\})$
        **if** $t <$ Approximation_$S_{ij}$ **then**
          Approximation_$S_{ij} := t$
        **end if**
        **for all** $d_y \in (B \setminus \{d_x\}$ **do**
          $t :=$ONEHOLEERGEBNIS$(B \setminus \{d_x, d_y\})$
          **if** $t <$ Approximation_$S_{ij}$ **then**
            Approximation_$S_{ij} := t$
          **end if**
          $t :=$MULTIHOLEERGEBNIS$(B \setminus \{d_x, d_y\})$
          **if** $t <$ Approximation_$S_{ij}$ **then**
            Approximation_$S_{ij} := t$
          **end if**
        **end for**
      **end for**
    **end if**
  **end for**
**end for**

**Algorithmus 3.3** Unterfunktion ONEHOLEERGEBNIS

// ONEHOLEERGEBNIS - Approximation unter Annahme, die optimale Lösung habe genau ein Loch etc.
// Seien UR, UM, UR etc. definiert wie im Text, Kreise gegeben als $d \in N, p \in S_{ij}$
**for all** $d_{l1} \in CL, d_{l2} \in CL, d_{r1} \in CR, d_{r2} \in CR, d_{u1} \in (CL \cup UL \cup UM \cup UR), d_{u2} \in (UL \cup UM \cup UR \cup CR), d_{d1} \in (CL \cup LL \cup LM \cup LR), d_{d2} \in (LL \cup LM \cup LR \cup CR)$ **do**
  $P' := \{p \in S_{ij} : p \notin (d_{l1} \cup d_{l2} \cup d_{r1} \cup d_{r2})\}$
  // Entferne Punkte, die schon von den linken bzw. rechten beiden Kreisen abgedeckt werden
  $d_L$ := Kreis durch die Schnittpunkte $d_{l1}, d_{u1}$ und $d_{l2}, d_{d1}$ mit Mittelpunkt links von $S_{ij}$
  $d_R$ := Kreis durch die Schnittpunkte $d_{r1}, d_{u2}$ und $d_{r2}, d_{d2}$ mit Mittelpunkt rechts von $S_{ij}$
  STRIPSEARCH (vertikal, Punkte in $d_L$ und $d_R$, Kreise in $UL \cup CL \cup LL$ und $UR \cup CR \cup LR$)
  STRIPSEARCH (horizontal, Punkte in $P' \setminus (d_L \cup d_R)$, Kreise in UPPER und LOWER)
**end for**
// Ergebnis ist die beste Zwischenlösung, bei der eine dominierende Menge errechnet wurde

**Algorithmus 3.4** Unterfunktion MULTIHOLEERGEBNIS

```
// MULTIHOLEERGEBNIS - Approximation unter Annahme, die optimale Lösung habe mehr als ein Loch
for all Kreispaare d_1, d_2 wie im Text angegeben do
  Berechne neues Koordinatensystem (Rotation, so dass Kreismittelpunkte von d_1, d_2 auf y-Achse liegen)
  // == Annahme: == keine Kreise aus CL (siehe Text)
  // Annahme: keine Kreise aus CR
  STRIPSEARCH (horizontal)
  // Annahme: ein Kreis aus CR
  for all Kreise d aus CR do
    STRIPSEARCH (horizontal, ohne die durch d dominierten Punkte)
  end for
  // Annahme: zwei oder mehr Kreise aus CR
  for all Kreispaarungen aus CR mit jeweils einem Kreis aus UPPER bzw. LOWER do
    Führe Stripsearch (Horizontal) durch
    Führe Stripsearch (Vertikal) durch
  end for
  // == Annahme: == ein Kreis aus CL (siehe Text)
  for all Kreise e aus CR do
    // Entferne e und alle dominierten Punkte,
    // verfahre weiter wie oben unter 'keine Kreis aus CL'
  end for
  // == Annahme: == mehrere Kreis aus CL (siehe Text)
  for all Kreispaarungen aus CL mit jeweils einem Kreis aus UPPER bzw. LOWER do
    // verfahre analog ONEHOLEERGEBNIS (virtuellen Kreis berechnen etc.)
    // dann Fallunterscheidung CR wie oben
  end for
end for
```

# 4 Gültigkeit des Algorithmus

Zum Nachweis der Gültigkeit sollen die Bauteile des Algorithmus einzeln betrachtet werden. Besonderes Augenmerk ist für die Verwendung von Teilalgorithmen als "Black Box" notwendig, da die Auswahl der Eingabedaten erheblichen Anteil am Funktionieren des Teilalgorithmus haben kann.

## 4.1 Strip-Search

Das Kernstück, die innerste Schleife des MWDS-Algorithmus wird vom "Strip-Search"-Algorithmus gebildet. Die Aufgabe dieser Methode ist es, eine optimale gewichtete Menge von Kreisen aus den (parallelen) Halbebenen $U$ und $L$ auszugeben, so dass alle Punkte im Streifen zwischen den Halbebenen von jeweils mindestens einem Kreis dominiert werden.

Zum Nachweis der Gültigkeit sind zwei Argumente (siehe Abschnitt 3.7) zu nennen:

1. Die betrachteten Kreise $u, d$ sind die jeweils in einem Punkt *aktiven* Kreise, also diejenigen, die bei $x = x_p$ am weitesten in den Streifen hineinragen

2. Bis auf die Halbebenen, die eine Sonderrolle spielen, gilt generell, dass eine Scheibe nur bei benachbarten Punkten in einem zusammenhängenden Bereich *aktiv* sein kann - trivial wegen Betrachtung von Kreisen gleichen Radius'.

Angenommen, der Algorithmus gäbe eine schlechtere Lösung $A$ als alle optimalen Lösungen (sei $Opt$ eine jede optimale Lösung, deren Verschiedenheit von $A$ zu zeigen ist) aus. Dann gibt es zwei Kreise $d_{ROpt}, d_{RA}$ aus UPPER oder LOWER, dessen Mittelpunkt am weitesten rechts unter allen Kreisen liegt, die nicht sowohl in $A$ als auch in $Opt$ liegen, sondern nur in $Opt$ bzw. $A$.

Man entferne nun alle Punkte, die von Kreisen mit Mittelpunkt weiter rechts abgedeckt werden, sowie alle solchen Kreise (die sowohl in $A$ als auch in $Opt$ liegen). Damit erhält man wieder eine optimale Lösung $Opt'$ (wäre diese nicht optimal, könnte durch Hinzufügen der gerade entfernten Kreise eine bessere Lösung als $Opt'$ erzielt werden ¬) sowie eine Lösung $A'$, die der gegebene Algorithmus als Zwischenschritt erzielt haben muss, und die er bei Eingabe nur der noch in $Opt'$ dominierten Punkte auch als beste Lösung fälschlicherweise erzielen würde (dies geht aus dem Algorithmus selbst hervor).

Man kann nun auch $d_{ROpt}$ und alle in $Opt'$ ausschließlich von $d_{ROpt}$ dominierten Punkte entfernen und enthält wieder ein optimal gelöstet Teilproblem mit $Opt''$. Dieses kann vom Algorithmus nicht gefunden worden sein, da ansonsten $Opt'$ gegenüber $A'$ gewählt worden sein müsste.

Da dieses Argument rekursiv anwendbar ist, gibt es eine Lösung mit nur einem Kreis zu einem oder mehreren Knoten, die optimal ist und nicht vom Algorithmus gefunden wird; das ist aber bei der verwendeten Methode auszuschliessen, was zum Widerspruch führt ¬.

Strip Search findet immer eine optimale Lösung für das gestellte Problem.

## 4.2 One-Hole-Lösungen

Der Algorithmus prüft in jedem Quadrat unter der Annahme, es läge eine One-Hole-Lösung vor, bei der alle Kreise am Rand des Lochs anliegen. Hier ist nun eine Fallunterscheidung nötig - das bedeutet, der Algorithmus nimmt jeden der Fälle an und sucht eine optimale Abdeckung, das insgesamt beste Ergebnis wird zwischengespeichert.

Aus Gründen der Übersichtlichkeit soll lediglich die linke Seite ($CL$) hier detailliert betrachtet werden, das Argument gilt aber symmetrisch auch für $CR$.

Die erste Annahme lautet, es gehört kein Kreis mit Mittelpunkt in $CL$ zur optimalen Lösung. Dieser Fall ist einfach: Es gehöre auch kein Kreis in $CR$ zur optimalen Lösung, dann lässt sich die optimale Lösung für das Teilproblem in $S_{ij}$ offensichtlich mit Strip Search finden.

Falls doch mindestens ein Kreis mit Mittelpunkt in $CL$ zu *Opt* gehört, so gibt es unter allen am Rand des Lochs beteiligten einen höchsten und einen niedrigsten Kreis $d_{lu}$ und $d_{ld}$ aus $CL$ (Im Sonderfall kann dies ein und der selbe Kreis sein - offensichtlich macht das in der folgenden Argumentation keinen wesentlichen Unterschied). Der Nachbarkreis nach oben bzw. unten muss aus UPPER oder CR bzw. LOWER oder CR stammen. Der Algorithmus prüft all dieses Varianten (insgesamt beteiligt sind acht Kreise, also $O(n^8)$ viele).

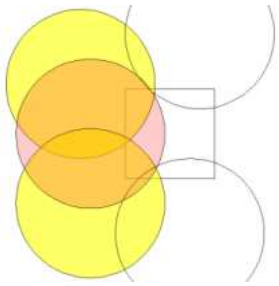

Abbildung 5: Skizze: $d_{lu}$ und $d_{ld}$ (gelb) mit $d_F$ (rosa) und Hörnchen.

Die beiden Kreise $d_{lu}$ und $d_{ld}$ haben mit diesen Nachbarn jeweils einen gemeinsamen Punkt, eine Ecke des Lochs. Man konstruiere nun einen fiktiven Kreis $d_F$, der durch diese beiden Ecken gehe und den Mittelpunkt

links des Quadrats $S_{ij}$ habe. Die in diesem Kreis, nicht aber in $d_{lu}$ und $d_{ld}$ liegenden Punkte (eine Fläche in Hörnchenform) können nur von Kreisen aus LEFT oder RIGHT abgedeckt werden, nicht von Kreisen aus UM oder LM. Demzufolge kann hier ein vertikales Strip Search zum Einsatz kommen. Alle weder von $d_{lu}$ und $d_{ld}$ (und dem Äquivalent auf der rechten Seite) noch in den beiden Hörnchen liegenden Punkte können nur von Kreisen aus UPPER und LOWER dominiert werden. Auch hier kann Strip Search zum Einsatz kommen.

### 4.3 Multi-Hole-Lösungen

Zuerst werden einige Überlegungen zu Multi-Hole-Umgebungen angestellt (siehe Abbildung zu Multi-Hole-Umgebungen zur Illustration).

Aus der Gestaltung von Multi-Hole-Umgebung ist klar, dass jeder Punkt im rechten oder linken $d_1d_2-$Loch der Gleichung

$$|y_p| \;<\; 1 - \sqrt{1-(x_p)^2}$$

genügt. Weiter gilt: Für jeden Punkt $p$ aus dem linken und $q$ aus dem rechten $d_1d_2$-Loch gilt eine $y$-Abstandsbeschränkung:

$$|y_p - y_q| \;\leq\; 1 - \sqrt{1-(x_p - x_q)^2}$$

Beides ist trivial aus der Lage der Einheitskreise herzuleiten.

Zunächst wird gezeigt, dass kein Kreis aus $RIGHT$ $(UR, CR, LR)$ einen Punkt aus dem linken Loch beinhalten kann (die symmetrische Aussage gilt analog).

Dazu wird ein indirekter Beweis geführt. Angenommen, es gäbe einen Kreis $d$ aus $RIGHT$ in $Opt$. Dieser kann das rechte $d_1d_2$-Loch nicht komplett überdecken, weil dies der Annahme einer Multi-Hole-Lösung entgegenstünde. Folglich existieren $p$ im rechten Loch ausserhalb $d$ und $q$ im linken Loch auf dem Kreisbogen von $d$ (auch das linke Loch kann mit selbem Argument nicht vollständig von $d$ überdeckt werden). Ohne Beschränkung der Allgemeinheit gelte $y_q \geq y_p$. Da $d$ den Punkt $p$, nicht aber $q$ enthält, muss der Mittelpunkt $m_d$ unterhalb von $q$ liegen. Weiterhin hat $m_d$ von $p$ genau den Abstand 1, da $p$ auf dem Einheitskreis $d$ liegt. Also liegt $m_d$ auf dem Einheitskreis um $p$. Es existiert ein weiterer Punkt $c$ auf diesem Einheitskreis, der auch von $q$ genau den Abstand 1 hat. Offensichtlich liegt $m_d$ links von $c$ (wegen $q \notin d$). Sei $\alpha$ der Basiswinkel des gleichschenkligen Dreiecks $\triangle qcp$ (in $p$ oder $q$). Dann gilt

$$dist(pq) < \frac{\sqrt{2}}{2} < 1 \quad \Rightarrow \quad \frac{\pi}{3} < \alpha < \frac{\pi}{2}$$

Bezeichne $\beta$ den Winkel zwischen $\bar{pq}$ und der von $q$ nach $-\infty$ verlaufenden senkrechten Halbgeraden. Offensichtlich gilt $0 < \beta < \pi$ und diese Aussagen sind äquivalent:

1. $c$ liegt rechts von $q$
2. $\alpha < \beta$

Dann folgert aus

$$\cos\alpha = \frac{dist(pq)}{2}$$

$$\cos\beta = \frac{y_q - y_p}{dist(pq)}$$

$$(y_q - y_p) \;\leq\; 1 - \sqrt{1 - (x_q - x_p)^2} \quad \Rightarrow$$

$$\Rightarrow 1 - (x_q - x_p)^2 \;\leq\; (1 - (y_q - y_p))^2 \quad \Rightarrow$$

$$\Rightarrow 2(y_q - y_p) \;\leq\; (y_q - y_p)^2 + (x_q - x_p)^2 = dist(pq)^2 \quad \Rightarrow$$

$$\Rightarrow \cos\beta \leq \cos\alpha \quad \Rightarrow \quad \alpha \leq \beta$$

dass $m_d$ links von $q$ liegt. Mit $q \in S_{ij}$ kann $m_d$ aber nicht wie vorausgesetzt in $RIGHT$ liegen, damit wäre der gewünschte Widerspruch herbeigeführt.

Die weitere Argumentation folgt dem Hörnchen-Argument für den One-Hole-Fall und endet in Anwendung des Strip-Search-Algorithmus.

### 4.4 Null-Hole- und entartete One-Hole-Lösungen

Bei der Betrachtung einer gedachten Null-Hole-Lösung habe die optimale Lösung keine Löcher, das Quadrat $S_{ij}$ sei also vollständig von Kreisen $d_k$ aus $Opt$ abgedeckt.

Dann gilt für jeden Kreis $d$ aus $Opt$, dass durch Entfernen dieses Kreises mindestens ein Loch entsteht. Durch Entfernen von $d$ wird also entweder eine One-Hole-, eine Multi-Hole- oder eine entartete One-Hole-Lösung hergestellt.

Im entarteten One-Hole-Fall kann durch Entfernen eines Kreises aus $Opt$ eine Multi-Hole-Lösung hergestellt werden:

Nach Annahme Entartung gibt es mindestens einen Kreis $d$ in $Opt$, der nicht am Rand des einzigen Loches liegt. Es gibt in $S_{ij}$ eine Fläche, die nur von $d$ überdeckt wird (andernfalls wäre $Opt$ nicht optimal, da durch Entfernen von $d$ immer noch eine überdeckende Lösung für alle Punkte existierte).

Durch Entfernen dieses Kreises entsteht also ein zweites Loch, wodurch der Multi-Hole-Fall eintritt.

Der Algorithmus spielt alle Möglichkeiten durch, also auch dieses Entfernen von $d$. Damit werden auch die beiden Sonderfälle korrekt behandelt.

### 4.5 Zusammenfügen der Teillösungen

Offensichtlich muss die Optimale Lösung $Opt_{ij}$ eines Teilquadrats $S_{ij}$ in eine der betrachteten Kategorien fallen. Damit wird sie vom Algorithmus in jedem Fall behandelt.

Wie im Abschnitt 6 (auf Seite Seite 22) behandelt, wird durch das Parzellieren der Ebene eine maximal konstante Verschlechterung herbeigeführt. Der Algorithmus betrachtet alle $S_{ij}$, in denen mindestens ein Mittelpunkt eines Einheitskreises liegt, damit ist er vollständig.

# 5 Laufzeit

Die Laufzeit des Algorithmus ist polynomiell in der Anzahl der Eingabeknoten ($n$). Dies ergibt sich aus der Verschachtelung von Schleifen, die in Polynomzeit ablaufen.

## 5.1 Strip-Search

Die innerste Schleife liegt im Algorithmus STRIPSEARCH (vergl. Algorithmus 3.1), hier laufen zwei verschachtelte Schleifen ($O(n)$ und $O(n^2)$), in der innersten Schleife läuft die Auswahl CHOOSE in quadratischer Zeit.

STRIPSEARCH läuft also in $O(n^5)$.

## 5.2 One-Hole/Multi-Hole

Weiter zu untersuchen ist die grösste Verschachtelungstiefe. Diese wird offensichtlich im Algorithmus MULTIHOLE auftreten – One-Hole muss daher nicht betrachtet werden. Die äusserste Schleife ist quadratisch, darin befindet sich eine $O(n^4)$-Schleife (jeweils vier Kreise bzw. zwei Kreispaarungen werden untersucht). Innerhalb dieser befindet sich eine weitere $O(n^4)$-Schleife, innerhalb derer STRIPSEARCH aufgerufen wird.

MULTIHOLE läuft in $O(n^(2 + 4 + 4 + 5)) = O(n^{15})$.

## 5.3 Zusammenfügen der Teillösungen

In der äussersten Schleife des Algorithmus (siehe Algorithmus 3.2) werden $O(n)$ Quadrate untersucht; diese können in $O(n \log(n))$ Zeit passend sortiert werden. Für jedes Quadrat werden $O(n)$ Durchläufe nötig (aufsteigend im Kreisgewicht). Innerhalb dieser Schleife werden zwei weitere in $n$ lineare Schleifen aufgerufen; in der innersten Schleife erfolgt ein Aufruf von MULTIHOLE.

Die Gesamtlaufzeit des Algorithmus errechnet sich demnach zu $O(n^{19})$ mit Eingabegrösse $n$ als Anzahl der gewichteten Einheitskreise.

# 6 Güte des Algorithmus

Der Approximationsfaktor begründet sich durch den Gütefaktor von 2 für das doppelt ausgeführte Stripsearch und die Tatsache, dass eine Betrachtung von Einzelquadraten nur eine im Produkt konstante Verschlechterung darstellt. Ein jeder Einheitskreis kann Punkte in maximal $O(1/\delta^2)$ Quadraten dominieren. Genauer muss jedes Quadrat, dass einen von einem Einheitskreis $d$ dominierten Punkt enthält, im Radius $1+\sqrt{2}\delta$ um den Mittelpunkt von $d$ vollständig enthalten sein, demzufolge gibt es maximal

$$q_{max} := \left\lfloor \pi \left(1+\sqrt{2}\delta\right)^2 / \delta^2 \right\rfloor = \left\lfloor \frac{\pi}{0.4995^2} \left(1+0.4995\sqrt{2}\right)^2 \right\rfloor = 36$$

solcher Quadrate.

Der Approximationsfaktor für den vorliegenden Algorithmus ist also 72.

# 7 Der Übergang zur verbundenen Knotenmenge

Der Algorithmus in der oben ausgearbeiteten Variante legt eine Approximation für das MWDS-Problem vor. Der Übergang zur verbundenen Knotenmenge setzt selbstverständlich voraus, dass eine Verbindung möglich ist, dass bedeutet, dass für jedes Paar $d_0, d_Z$ von Kreisen eine Folge $1 \dots k$ existiert mit

$$\forall i \in \{1 \dots k\} : d_i \text{ dominiert } d_{i-1}; \quad d_Z \text{ dominiert } d_k$$

Ein Cluster[10] im Einheitskreisgraphen ist gegeben durch einen Kreis $d$ sowie alle Kreise $d_X$ mit $d$ dominiert $d_X$. Jede Zusammenhangskomponente ist über maximal drei Kanten mit einer weiteren Zusammenhangskomponente verbunden (die beiden dazwischen liegenden Knoten bzw. Einheitskreise werden von jeweils einer der beiden Zusammenhangskomponenten dominiert.

Nun kann mit einem Greedy-Algorithmus aus allen separaten Clustern ein zusammenhängender dominierender Einheitskreisgraph gebildet werden. Dieser ist maximal Faktor 17 schlechter als die optimale Lösung.

Dieser Faktor 17 ist begründet darin, dass in einer Distanz 2 von jedem Knoten des Graphen maximal 18 Kreise verschiedern Cluster liegen können: Dies ist anhand maximal dicht gelegter gleich großer Münzen direkt evident.

In Gesamtheit gilt für den Algorithmus und Übergang zur verbundenen Knotenmenge, also das Problem MWCDS, ein Approximationsfaktor von $72 + 17 = 89$.

[10] Zusammenhangskomponente

# 8 Fazit

Der hier vorgestellte Algorithmus ist eher von theoretischem als von praktischem Interesse. Sowohl die hohe Laufzeit (in $O(n^{19})$) als auch der Approximationsfaktor 72 sind für einen Approximationsalgorithmus allgemein gesprochen nicht gering. Andererseits ist dies der erste überhaupt vorgestellte polynomielle Algorithmus für MWDS und MWCDS, insofern ist das Paper von Ambühl et al. in der Theorie interessant und bemerkenswert.

Einige der im vorgestellten Algorithmus verwendeten Methoden, insbesondere Reskalierung und Rotation, aber auch die konsequente Anwendung dynamischer Programmierung, machen diese Techniken anschaulich.

Die Auswirkungen des Papers auf Arbeiten über generelle Kreisgraphen bleiben abzuwarten.

# Literatur


[1] J. Alber. Exact algorithms for np-hard problems on networks: Design, analysis, implementation. 2002.

[2] K. Alzoubi, P.-J. Wan, and O. Frieder. Message-optimal connected dominating sets in mobile ad hoc networks. *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2002)*, pages 157–164, 2002.

[3] C. Ambühl, T. Erlebach, M. Mihal'ák, and M. Nunkesser. Constant-factor approximation for minimum-weight (connected) dominating sets in unit disk graphs. Paper, 2006.

[4] B. S. Baker. Approximation algorithms for np-complete problems on planar graphs. *Journal of the ACM*, pages 153–180, 1994.

[5] X. Cheng, X. Huang, D. Li, W. Wu, and D.-Z. Du. A pllynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks*, pages 42(4):202–208, 2003.

[6] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Matehmatics*, pages 86:165–177, 1990.

[7] M. Franceschetti, M. Cook, and J. Bruck. A geometric theorem for approximate disk covering algorithms. 2001.

[8] H. B. H. III, M. V. Marathe, V. Radhakrishnan, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. Nc-approximation schemes for np- and pspace-hard problems for geometric graphs. *Journal of Algorithms*, pages 26(2):238–274, 1998.

[9] D. Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, pages 11(2):329–343, 1982.

[10] M. V. Marahte, H. Breu, H. B. H. III, S. S. Ravi, and D. J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, pages 25:59–68, 1995.

[11] F. J. van Leeuwen. Approximation algorithms for unit disk graphs. *Proceedings of the 31st International Workshop on Graph-Theoretic Concepts in Computer Science (WG'05), LNCS 3787*, pages 351–361, 2005.

[12] Y. Wang and X.-Y. Li. Distributed low-cost backbone formation for wireless ad hoc networks. *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2005)*, pages 2–13, 2005.