Scheduling im $SINR_G$-Modell

Christof Doll

Karlsruher Institut für Technologie (KIT), 76128 Karlsruhe, Deutschland

1 Einführung

In dieser Ausarbeitung stellen wir die Probleme One-Slot-Scheduling und Scheduling im Modell $SINR_G$ vor. Dieses Modell ist physikalisch motiviert und geht davon aus, dass Sensorknoten in der euklidischen Ebene verteilt sind. Sendende Sensorknoten üben Interferenz aufeinander aus. Bei den beiden genannten Problemen geht es darum, gewünschte Funkverbindungen so zu planen, dass die dabei auftretende Interferenz nicht eine erfolgreiche Übertragung von Daten scheitern lässt. Bei One-Slot-Scheduling ist nach der maximalen Anzahl gleichzeitig realisierbarer Verbindungen gefragt; bei Scheduling nach einem Zeitplan, der alle Verbindungen auf mehrere Zeitintervalle verteilt. Wir zeigen, dass One-Slot-Scheduling $\mathcal{NP}$-vollständig ist und stellen Approximationsalgorithmen zum Lösen der beiden genannten Probleme vor. Der für One-Slot-Scheduling vorgestellte Algorithmus hat eine konstante Apporximationsrate, die Approximationsrate für Scheduling ist logarithmisch.

2 Das Modell $SINR_G$

$SINR_G$ ist ein physikalisch motiviertes Modell, das zur Beschreibung und Simulation von Sensornetzwerken benutzt wird. Dabei repräsentieren Punkte in der euklidischen Ebene die einzelnen Sensorknoten. Es wird davon ausgegangen, dass ein Sensorknoten zu einem Zeitpunkt nur eine Funkverbindung eingehen kann. Er kann also entweder zu einem anderen Sensorknoten Nachrichten senden oder von einem anderen Sensorknoten Signale empfangen, aber nicht beides zu gleich. Hingegen ist es zulässig, dass sich mehrere Sensorknoten in einem Punkt der Ebene befinden. Mehrere sequentielle Funkverbindungen eines realen Sensorknotens können also durch mehrere virtuelle Sensorknoten an der selben Stelle realisiert werden, von denen jeder nur eine Verbindung eingeht. Die Funkverbindungen, die realisiert werden sollen, werden als ein Paar von einem Sender s und einem Empfänger r angegeben. Wir notieren diese als $L = \{l_1, \dots l_n\}$, wobei $l_i = (s_i, r_i)$. Wie bereits angemerkt gilt $s_i, r_i \in \mathbb{R}^2$. In diesem Modell werden die Signale radial ausgesendet. Dabei sende alle Sensorknoten mit der selben Sendeleistung P. Die Signalstärke nimmt mit wachsendem Abstand d zum Sender ab und berechnet sich durch P/d^α. $\alpha > 2$ ist dabei der sogenannte Pfadverlust-Exponent. Je größer der Pfadverlust-Exponent, desto stärker ist die Abnahme der Signalstärke. Die Signalstärke P_{ji} des vom Senders s_j an den Empfänger r_i gesendeten Signals beträgt also $P_{ji} := P/d_{ji}^\alpha$. Dabei bezeichnen wir mit $d_{vw} := d(s_v, r_w)$ den euklidischen Abstand zwischen dem Sender s_v und dem Empfänger r_w. Das $_G$ in $SINR_G$ repräsentiert die Tatsache, dass die Geometrie der Sender und Empfänger die Signalstärke bei den Empfängern impliziert.

Gleichzeitig realisierte Funkverbindungen können Interferenz aufeinander ausüben. Ob eine Funkverbindung erfolgreich realisiert werden kann, hängt von der der Signalstärke

sowie von der Interferenz ab, die von anderen Sendern ausgeht. Dazu wird das Verhältnis von Signalstärke zu Interferenz plus Rauschen (Signal-to-Interference-plus-Noise-Ratio), kurz $SINR_G$ betrachtet.

$$SINR_G(r_i) := \frac{P_{ii}}{\sum_{j \neq i} P_{ji} + N}$$

In dieser Arbeit gehen wir aus Gründen der Übersichtlichkeit und Einfachheit davon aus, dass das Rauschen $N = 0$ ist. Sämtliche Argumente sind jedoch für $N \neq 0$ gültig. Eine Verbindung wird genau dann erfolgreich umgesetzt, wenn $SINR_G(r_i) \geq \beta$. Dabei ist $\beta > 1$ ein von der Hardware abhängiger Schwellwert.

3 Die Scheduling-Probleme

In diesem Kapitel werden wir zwei Probleme definieren. Diese sind SCHEDULING und ONE-SLOT-SCHEDULING. Beide Probleme basieren auf einer Menge $L = \{l_1, \dots l_n\}$ von zu realisierenden Funkverbindungen. Das Problem ONE-SLOT-SCHEDULING ist ein Maximierungsproblem, bei dem das Ziel ist, eine möglichst große Menge an Funkverbindungen gleichzeitig zu realisieren. SCHEDULING hingegen ist ein Minimierungsproblem. Die Funkverbindung in L werden dabei so in mehrere Zeitabschnitte eingeteilt, dass alle Übertragungen innerhalb eines Zeitabschnittes gleichzeitig stattfinden können. Die Anzahl der benötigten Zeitabschnitte ist zu minimieren. Betrachten wir nun die formalen Definitionen der zwei Probleme:

Definition 1 (One-Slot-Scheduling). *Gegeben sei eine Menge $L = \{l_1, \dots, l_n\}$ von Funkverbindungen. Finde $L' \subseteq L$ für das folgende Eigenschaften gelten:*

- $\forall l \in L' : SINR_G(l) \geq \beta$
- $|L'| \stackrel{!}{=} \max$

Definition 2 (Scheduling). *Gegeben sei eine Menge $L = \{l_1, \dots, l_n\}$ von Funkverbindungen. Finde einen Zeitplan $S = \{S_1, \dots, S_T\}$ für den folgende Eigenschaften zutreffen:*

- $S_1 \cup \dots \cup S_T = L$
- $\forall t \in \{1, \cdots, n\} : \forall l \in S_t : SINR_G(l) \geq \beta$
- $T \stackrel{!}{=} \min$

Das Problem SCHEDULING ist $\mathcal{NP}$-vollständig [1]. Der Beweis dieser Aussage wurde von Goussevskaia et al. 2007 erstmals veröffentlicht. Diesen wollen wir im Rahmen dieser Ausarbeitung vorstellen. Wir zeigen dies, indem wir zuerst beweisen, dass SCHEDULING $\in \mathcal{NP}$ ist. Anschließend reduzieren wir das $\mathcal{NP}$-vollständige Problem PARTITION [2] auf SCHEDULING. Daraus folgt dann die $\mathcal{NP}$-Vollständigkeit von SCHEDULING.

Lemma 1. SCHEDULING $\in \mathcal{NP}$

Beweis Für jeden $l \in L$ lässt sich $SINR_G(l)$ in $\mathcal{O}(n)$-Zeit berechnen, da nur die Interferenz von maximal n zeitgleich geplanten Verbindungen berechnet werden muss. Da $SINR_G(l)$ für jede Verbindung $l \in L$ nur ein Mal berechnet werden muss und $|L| = n$, kann die Validität einer Lösung in $\mathcal{O}(n^2)$ überprüft werden. □

Völker et al. merken zu diesem Beweis an, dass für gewisse Rechnermodelle noch nicht gezeigt ist, dass der Vergleich von der Summe mehrerer Wurzeln mit einer weiteren Zahl in polynomialer Zeit stattfinden kann. Da dies ein Spezialfall der Bestimmung des $SINR_G$ mit anschließendem Vergleich mit dem Schwellenwert β ist, ist somit auch nicht gezeigt, dass SCHEDULING $\in \mathcal{NP}$. [3]

Definition 3 (Partition). *Gegeben sei eine Menge $\mathcal{I} = \{i_1, \ldots, i_n\}$ bestehend aus natürlichen Zahlen. Gesucht sind zwei Mengen $\mathcal{I}_1, \mathcal{I}_2 \subseteq \mathcal{I}$, sodass:*

- $\mathcal{I}_1 \cap \mathcal{I}_2 = \emptyset$
- $\mathcal{I}_1 \cup \mathcal{I}_2 = \mathcal{I}$
- $\sum_{i \in \mathcal{I}_1} i = \sum_{i \in \mathcal{I}_2} i = \frac{1}{2} \sum_{i \in \mathcal{I}} i$

Lemma 2. PARTITION $\propto$ SCHEDULING

Beweis Zu Beginn des Beweises werden wir die Transformation angeben. Im zweiten Schritt erläutern wir, warum diese Transformation in polynomialer Zeit durchgeführt werden kann. Darauf folgend zeigen wir, dass jede Ja-Instanz von PARTITION auf eine Ja-Instanz von SCHEDULING abgebildet wird und selbiges auch für Nein-Instanzen gilt.

Für jedes Element aus $\mathcal{I}$ erzeugen wir eine Funkverbindung. Außerdem fügen wir noch zwei weitere Verbindungen hinzu, die von von der Summe der Elemente in $\mathcal{I}$ abhängen. Wir werden zeigen, dass durch die Geometrie der Sender und Empfänger bedingt nicht alle Verbindung in einem Zeitabschnitt realisiert werden können und ein Zeitplan aus zwei Abschnitten genau dann existiert, wenn die zugehörige PARTITION-Instanz eine Lösung hat.

Für jedes $i_j, j \in \{1, \cdots, n\}$ definieren wir eine Verbindung $l_j = (s_j, r_j)$ mit

$$\begin{aligned} s_j &:= ((\tfrac{P}{i_j})^{\frac{1}{\alpha}}, 0) \\ r_j &:= s_j + (d_{min}, 0). \end{aligned}$$

Auf die Wahl der Konstanten d_{min} gehen später ein. Für das weitere Vorgehen setzen wir $\sigma = \sum_{i \in \mathcal{I}} i$. Nun definieren wir zwei zusätzliche Verbindung $l_{n+1} = (s_{n+1}, r_{n+1})$ und $l_{n+2} = (s_{n+2},\ r_{n+2})$, für die gilt

$$\begin{aligned} r_{n+1} &:= (0,0), \quad & s_{n+1} &:= (0, (\tfrac{2P}{\beta\sigma})^{\frac{1}{\alpha}}) \\ r_{n+2} &:= (0,0), \quad & s_{n+2} &:= (0, -(\tfrac{2P}{\beta\sigma})^{\frac{1}{\alpha}}). \end{aligned}$$

Abbildung 1 skizziert die Transformation einer PARTITION-Instanz auf einer SCHEDULING-Instanz. Es ist offensichtlich, dass diese Transformation in linearer Zeit in n durchgeführt werden kann. Es bleibt also lediglich zu zeigen, dass die PARTITION-Instanz genau dann eine Ja-Instanz ist, wenn ein Zeitplan aus zwei Zeitabschnitten existiert. Um diese Aussage zu beweisen, entwickeln wir zuerst einige Formeln. Die zweite dieser Formeln folgt aus der Symmetrie der Verbindungen l_{n+1} und l_{n+2}.

$$P_{(n+1)(n+1)} = \frac{P}{d^{\alpha}_{(n+1)(n+1)}} = \frac{P}{((2P/(\beta\sigma))^{\frac{1}{\alpha}})^{\alpha}} = \frac{P}{2P/(\beta\sigma)} = \frac{\beta\sigma}{2}$$

$$P_{(n+2)(n+2)} = P_{(n+1)(n+2)} = P_{(n+2)(n+1)} = P_{(n+1)(n+1)} = \frac{\beta\sigma}{2}$$

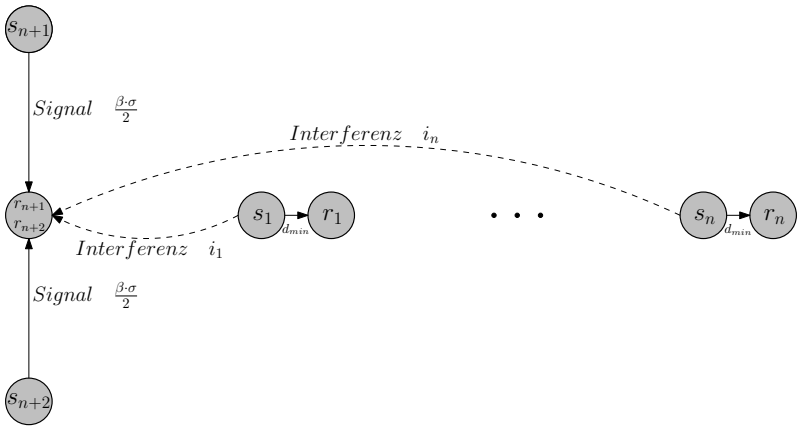


Abbildung 1: Reduktion von PARTITION auf SCHEDULING.

$$SINR(r_{n+2}) = \frac{P_{(n+2)(n+2)}}{\sum_{j\neq n+2, j\neq n+1} P_{j(n+2)} + P_{(n+1)(n+2)}} = \frac{\beta\sigma/2}{\sum_{j\neq n+2, j\neq n+1} P_{j(n+2)} + \beta\sigma/2} \leq 1$$

Da $\beta > 1$, können also l_{n+1} und l_{n+2} nicht gleichzeig realisiert werden. Daraus folgt, dass jeder gültige Zeitplan mindestens aus zwei Zeitabschnitten besteht. Des Weiteren wählen wir d_{min} so klein, dass keine Verbindung $l_i, i \in \{1, \ldots, n\}$ durch die von $\{l_1, \ldots, l_{n+1}\}\backslash\{l_i\}$ ausgehende Interferenz verhindert werden kann. Das heißt, dass alle Verbindungen $l_1, \ldots, l_n$ gleichzeitig realisiert werden könnten. Es bleibt zu untersuchen, wie sich die Interferenz der Verbindungen $l_1, \ldots, l_n$ auf l_{n+1} und l_{n+2} auswirken.

$$P_{i(n+1)} = \frac{P}{d^{\alpha}_{i(n+1)}} = \frac{P}{((P/i_j)^{\frac{1}{\alpha}})^{\alpha}} = \frac{P}{P/i_j} = i_j$$

Die Behauptung ist, dass genau dann wenn die zwei Mengen $\mathcal{I}_1, \mathcal{I}_2$ mit Eigenschaften wie gefordert existieren, ein valider Zeitplan (S_1, S_2) mit zwei Zeitabschnitten existiert. Dieser ist

$$S_1 = \{l_{n+1}\} \cup \{l_j : i_j \in \mathcal{I}_1\}$$
$$S_2 = \{l_{n+2}\} \cup \{l_j : i_j \in \mathcal{I}_2\}.$$

Um zu beweisen, dass dieser Zeitplan eine gültige Lösung für SCHEDULING ist, müssen wir nun noch $SINR_G(r_{n+1})$ und $SINR_G(r_{n+2})$ berechnen.

$$SINR(r_{n+1}) = \frac{P_{(n+1)(n+1)}}{\sum_{j\in S_1, j\neq n+1} P_{j(n+1)}} = \frac{\beta\sigma/2}{\sum_{i_j\in\mathcal{I}_1} i_j} = \frac{\beta\sigma/2}{\sigma/2} \geq \beta$$

$$SINR(r_{n+2}) = \frac{P_{(n+2)(n+2)}}{\sum_{j\in S_2, j\neq n+2} P_{j(n+2)}} = \frac{\beta\sigma/2}{\sum_{i_j\in\mathcal{I}_2} i_j} = \frac{\beta\sigma/2}{\sigma/2} \geq \beta$$

Nun bleibt noch die Umkehrung dieser Implikation zu zeigen. Es ist also zu zeigen, dass kein gültiger Zeitplan mit zwei Zeitabschnitten existiert, wenn sich die Menge $\mathcal{I}$ nicht in zwei gleich schwere Mengen partitionieren lässt. Angenommen ein solcher Zeitplan existiert, dann werden – wie oben gezeigt – l_{n+1} und l_{n+2} in unterschiedlichen Zeitabschnitten

realisiert. Die Verbindungen $l_1, \ldots, l_n$ sind ebenfalls auf diese zwei Zeitabschnitte verteilt. Diese Verteilung korrespondiert der Partition der Menge $\mathcal{I}$ in $\mathcal{I}_1$ und $\mathcal{I}_2$. Außerdem gilt dann $\sum_{i \in \mathcal{I}_1} i > \sigma/2$ oder $\sum_{i \in \mathcal{I}_2} i > \sigma/2$. Somit wertet sich einer der Ausdrücke $SINR(r_{n+1})$ und $SINR(r_{n+2})$ zu einem Wert aus, der kleiner als β ist. Dies widerspricht der Validität des Zeitplanes. Folglich existiert ein Zeitplan mit zwei Zeitabschnitten genau dann wenn die betreffende PARTITION-Instanz eine Ja-Instanz ist. □

4 Algorithmen für One-Slot-Scheduling und Scheduling

Im Jahr 2009 veröffentlichten Goussevskaia et al. Algorithmen zur Approximation der beiden Probleme ONE-SLOT-SCHEDULING und SCHEDULING. Die vorgestellten Algorithmen verfolgen eine Master-Slave-Approximationsstrategie. Dabei wird zuerst ein Algorithmus für ONE-SLOT-SCHEDULING konstruiert, der eine konstante Approximationsrate besitzt. Auf diesem basiert der Algorithmus zur Approximation des SCHEDULING-Problems. [4]

4.1 One-Slot-Scheduling

Wir wollen im Folgenden die Algorithmen von Goussevskaia et al. vorstellen und analysieren. Zuvor müssen wir jedoch noch einige Vorbemerkungen treffen. Wir definieren den Begriff *Affektiertheit*. Die Affektiertheit eines Empfängers $a_S(r_v) := \beta / SINR(r_v)$ gibt wie auch $SINR_G(r_v)$ Aussage darüber, ob eine Verbindung erfolgreich umgesetzt werden kann. Es gilt $a_S(r_v) \leq 1 \Leftrightarrow SINR(r_v) \geq \beta$. Die Affektiertheit eines Empfängers bezeichnen wir auch mit der Affektiert der Verbindung, zu der eben jener Empfänger gehört. Die Verwendung der Affektiertheit vereinfacht den Umgang mit dem Modell, da $a_S(r_v)$ als Summe dargestellt werden kann. Dabei ist jeder einzelner Summand nur von r_v und einem weiteren Sender s_u abhängig. Einen solchen Summanden $RI_u(v) = P_{uv}/P_{vv}$ bezeichnen wir als *relative Interferenz* des Senders s_u. Dieses Konstrukt wird uns insbesondere beim Beweis der Korrektheit von Algorithmus 1 zugutekommen.

Algorithmus 1 : One-Slot-Scheduling

Eingabe : $L = \{l_1, \ldots l_n\}$
Ausgabe : $S \subseteq L$

```
Setze c  \\Konstante abhängig von α und β
Wiederhole
    Kürzeste Verbindung l_v in S einfügen
    Lösche l_u ∈ L : d_uv ≤ c · d_vv
    Lösche l_w ∈ L : a_S(l_w) ≥ 2/3
Bis L = ∅
Gebe S zurück
```

Algorithmus 1 fügt die Verbindungen der Länge nach aufsteigend gierig zur Menge S hinzu. Es werde nun die kürzeste in L enthaltene Verbindung l_v in S aufgenommen. Eine erfolgreiche Übertragung von l_v wird anschließend durch eine Modifikation der Menge L gesichert. Zuerst werden alle Verbindungen gelöscht deren Sender innerhalb des Radius $c \cdot d_{vv}$ des Empfängers r_v liegen. Dabei ist c eine Konstante, die von α und β abhängt. Auf ihre Bestimmung wollen wir in dieser Arbeit nicht weiter eingehen; stattdessen sei auf den

Artikel “Capacity of Arbitrary Wireless Networks” [4] verwiesen. Dann werden alle Verbindungen entfernt, deren Affektiertheit durch die Hinzunahme der von l_v ausgehenden relativen Interferenz auf einen Wert größer oder gleich 2/3 angestiegen ist. Dieser Prozess wird solange durchgeführt bis alle Verbindungen aus L entweder in S aufgenommen oder gelöscht wurden.

Korrektheit des One-Slot-Scheduling-Algorithmus

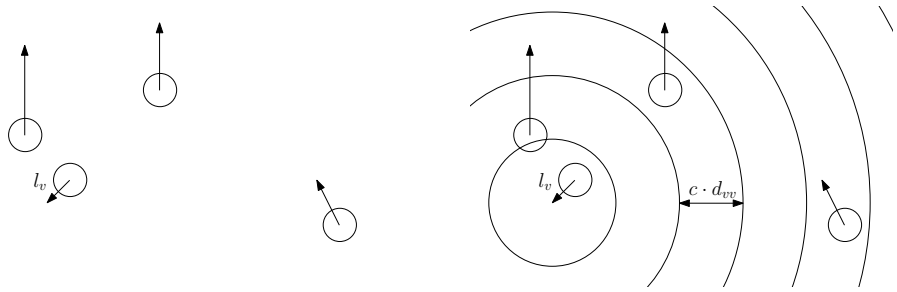
Lemma 3. *Algorithmus 1 berechnet eine korrekte Lösung.*

Beweisskizze Wenn eine Verbindung l_v in S aufgenommen wird, ist ihre Affektiertheit kleiner als 2/3. Wir müssen also zeigen, dass die relative Interferenz der zu einem späteren Zeitpunkt eingefügten Verbindungen in der Summe kleiner oder gleich 1/3 ist. Die Menge dieser Verbindungen bezeichnen wir im Folgenden mit S_v^+. Für diese Verbindungen gilt, dass der Abstand des jeweiligen Senders zum Empfänger größer ist als der Abstand von s_v zu r_v. Durch das erste Eliminationskriterium ist gegeben, dass um jeden Empfänger $r_w \in S_v^+$ eine Kreisscheibe des Radius $c \cdot d_{ww}$ existiert, die außer s_w keinen weiteren Sender enthält. Mithilfe dieser Erkenntnis und der Dreiecksungleichung kann eine untere Schranke für den Abstand je zweier Sender aus S_v^+ aufgestellt werden. Diese ist $d_{vv} \cdot (c-1)$. Kreisscheiben des Radius $d_{vv} \cdot (c-1)/2$ um Sender in S_v^+ schneiden sich also nicht. Abbildung 2a illustriert diese Gegebenheit.

Weiterhin teilen wir die Sender in S_v^+ in mehrere Menge auf. Wir unterteilen dazu die euklidische Ebene in konzentrische Ringe der Breite $c \cdot d_{vv}$ mit dem Mittelpunkt r_v (siehe Abbildung 2b). Der Ring R_k enthält dann alle Sender aus $s_w \in S_v^+$ für die $k \cdot c \cdot d_{vv} \leq d_{wv} \leq (k+1) \cdot c \cdot d_{vv}$ gilt. Ring R_0 enthält aufgrund des ersten Eliminationskriteriums keine Sender. Liegt ein Sender s_w in Ring R_k, so muss die Kreisscheibe um s_w in einem erweiterten Ring R_k enthalten sein. Diesen erweiterten Ring R_k erhalten wir, indem wir die Breite des Ringes in beide Richtungen um $d_{vv} \cdot (c-1)/2$ erweitern. Nun können wir über ein Flächen-Argument die maximale Anzahl der Sender pro Ring bestimmen, indem wir die Fläche des erweiterten Ring R_k durch die Fläche einer Kreisscheibe teilen. Außerdem haben wir bereits eine untere Schranke für den Abstand zu r_v für alle Sender im Ring R_k. Somit lässt sich auch die relative Interferenz, die von allen Sender innerhalb eines Ringes ausgeht, nach oben abschätzen. Summiert man diese Abschätzungen für alle Ring R_k mit k von 1 bis ∞ auf, so kommt man zu dem Ergebnis $a_S(r_v) < 1/3$ bei geeigneter Wahl von c. Für weitere Details verweisen wir auf die Arbeit “Capacity of Arbitrary Wireless Networks” [4].

Approximationsrate des One-Slot-Scheduling-Algorithmus

Nun wollen wir den von Goussevskaia et al. geführten Beweis, dass Algorithmus 1 eine konstante Approximationsrate besitzt, skizzieren. Dazu vergleichen wir die von Algorithmus 1 berechnete Lösung ALG mit einer optimalen Lösung OPT. Dabei bezeichnen wir mit ALG beziehungsweise OPT sowohl die Mengen als auch die Anzahl der Elemente in diesen Mengen. Wir werden zeigen, dass $OPT \leq \rho \cdot ALG$, die Anzahl der optimalerweise gleichzeig realisierbaren Verbindung also maximal um einen konstanten Faktor höher ist als die Anzahl, die Algorithmus 1 berechnet. Dazu werden wir eine obere Schranke für $OPT' = OPT \setminus ALG$ aufstellen.



(a) Disjunkte Kreisscheiben um Sender. (b) Ringe um r_v mit Radius $c \cdot d_{vv}$.

Abbildung 2: Illustration des Beweises von Lemma 3 ($c = 2$).

Zunächst definieren wir zwei Teilmengen von OPT', OPT_1 und OPT_2. OPT_1 beinhalte die in Zeile 4, OPT_2 die in Zeile 5 gelöschten Verbindungen.

Lemma 4. *Sei X eine Lösung und l_v eine Verbindung, die in X realisiert wird. Die Anzahl der Sender in X im Abstand $k \cdot d_{vv},\ k \leq 1$ vom Empfänger r_v ist höchsten k^α. Die Anzahl der Sender in X im Abstand $k \cdot d_{vv},\ k \leq 1$ zu s_v beträgt maximal $(k+1)^\alpha$.*

Beweis Die relative Interferenz von s_u auf r_v lässt sich nach unten abschätzen:

$$RI_u(v) = \frac{P_{uv}}{P_{uv}} = \frac{P/d_{uv}^\alpha}{P/d_{vv}^\alpha} = \frac{d_{vv}^\alpha}{d_{uv}^\alpha} \geq \frac{d_{vv}^\alpha}{k \cdot d_{vv}^\alpha} = \frac{1}{k^\alpha}$$

Da die Affektiertheit von l_v kleiner oder gleich 1 sein muss, kann es also höchstens k^α Sender in diesem Abstand geben. Da außerdem Sender innerhalb des Radius $k \cdot d_{vv}$ von r_v höchstens Abstand $(k+1) \cdot d_{vv}$ von s_v haben, folgt die zweite Aussage. □

Lemma 5. *$OPT_1 \leq \rho_1 \cdot ALG$ mit $\rho_1 = (2c+1)^\alpha$.*

Beweis Es sei X_v die Teilmenge der Verbindungen aus OPT_1, die in der Iteration, in der l_v in ALG aufgenommen wird, aufgrund des ersten Eliminationskriteriums entfernt werden. Beachte, dass l_v selbst nicht in OPT_1 enthalten sein muss. Jede Verbindung l_w in X_v hat mindestens die Länge d_{vv} und der zugehörige Sender ist höchstens im Abstand $c \cdot d_{vv}$ von r_v. Also sind alle Sender in X_v im Radius von $2c \cdot d_{vv} \leq 2c \cdot d_{ww}$ von s_w (siehe Abbildung 3). Da $l_w \in X_v$ können wir nun Lemma 4 auf X_v und l_w anwenden. Es folgt, dass X_v höchstens $(2c+1)^\alpha$ Sender enthält. Für jede Verbindung in ALG werden also höchstens $(2c+1)^\alpha$ Verbindungen eliminiert. Diese sind in OPT_1 enthalten. Folglich erfüllt $\rho_1 = (2c+1)^\alpha$ die zu beweisende Ungleichung. □

Für den Beweis einer oberen Schranke von OPT_2 benutzen wir das sogenannte *blue-dominant centers* Lemma, das wir hier nicht beweisen werden (stattdessen verweisen wir erneut auf "Capacity of Arbitrary Wireless Networks" [4]). Zunächst definieren wir noch die Begriffe "blau-dominant" und "Wächter-Menge".

Definition 4. *Seien $\mathcal{R}$ und $\mathcal{B}$ zwei disjunkte Mengen von Punkten in der euklidischen Ebene. Im Folgenden bezeichnen wir diese als rote und blaue Punkte. Für eine natürliche Zahl q nennen wir einen Punkt $b \in \mathcal{B}$ q-blau-dominant, wenn jeder Kreis $B_\delta(b) = \{p \in \mathbb{R}^2 : d(p,b) \leq \delta\}$ mehr als q-mal so viele blaue Punkte wie rote Punkte enthält.*

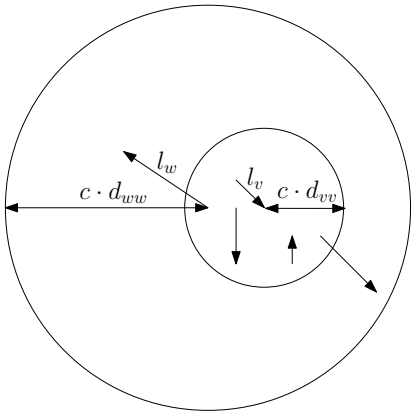


Abbildung 3: Illustration von 5 ($c = 2$).

Definition 5. *Seien $\mathcal{R}$ und $\mathcal{B}$ wie oben definiert. Es sei r ein roter Punkt und $G \subseteq \mathcal{B}$. G ist eine Wächter-Menge von r, wenn für alle $b \in \mathcal{B} \setminus G$ gilt, dass $B_{d(b,r)}(b) \cap G \neq \emptyset$.*

Lemma 6 (Blue-dominant centers Lemma im zweidimensionalen). *Seien $\mathcal{R}$ und $\mathcal{B}$ zwei disjunkte Mengen von Punkten in der euklidischen Ebene und q eine natürliche Zahl. Wenn $|\mathcal{B}| > 5q \cdot |\mathcal{R}|$, dann existiert mindestens ein q-blau-dominanter Punkt in $\mathcal{B}$.*

Lemma 7. *$OPT_2 \leq \rho_2 \cdot ALG$ mit $\rho_2 = 5q$, wobei $q = 3^{\alpha+1}$.*

Beweisskizze Um die Aussage zu beweisen, führen wir einen klassischen Widerspruchsbeweis. Angenommen es gelte $OPT_2 > \rho_2 \cdot ALG$ mit ρ_2 und q wie oben definiert. Wir markieren nun die Sender aus OPT_2 blau und die aus ALG rot. Nach Lemma 6 existiert dann ein q-blau-dominanter Sender s^* in der Menge der Sender aus OPT_2. Wir zeigen nun, dass die zugehörige Verbindung l^* von Algorithmus 1 gewählt wird, was zu einem Widerspruch führt, da s^* somit nicht aus OPT_2 sein kann.

Betrachten wir nun einen roten Sender s_r und die zugehörige Wächter-Menge. Mit $G^*(s_r)$ bezeichnen wir die Menge der Sender aus der Wächter-Menge von s_r, die auf einer Kreisscheibe um s^* vom Radius $d(s^*, s_r)$ liegen. Auf dieser Kreisscheibe liegt mindestens ein roter Sender, nämlich s_r. Wendet man das Blue-dominant centers Lemma auf s^* an, folgt, dass diese Kreisscheibe also mehr als $5q \cdot 1$ blaue Sender enthalten muss. Diese sind bis auf s^* auch in $G^*(s_r)$ enthalten. Es folgt also, dass $|G^*(s_r)| \geq 5q > q$ gelten muss.

Nun können wir eine untere Schranke für die relative Interferenz des roten Senders s_r auf l^* aufstellen. Hierzu verwenden wir Lemma 4 in leicht abgeschwächter Form. Lemma 4 besagt, dass im Radius $k \cdot d(s^*, r^*)$ von s^* höchstens $(k+1)^\alpha$ Sender gleichzeitig senden können. Wir erhöhen diese obere Schranke für die Anzahl der Sender auf $(k+1)^{\alpha+1}$, um in einem späteren Schritt, der in dieser Beweisskizze nicht aufgeführt ist, entsprechend kürzen zu können. Da alle blauen Sender in $G^*(s_r)$ aus OPT_2 stammen und somit in der Optimallösung gleichzeitig aktiv sind, muss Lemma 4 auch hier gelten. Wir suchen also ein k, dass die Ungleichung $q = 3^{\alpha+1} \leq (k+1)^{\alpha+1}$ erfüllt. $k = 2$ ist offensichtlich das kleinste k, das dieser Bedingung genügt. Es gilt also, dass $d(s^*, s_r) \geq 2d(s^*, r^*)$. Somit können wir auch $RI_{s_r}(l^*)$ nach oben abschätzen.

Für die Summe der relativen Interferenzen der blauen Sender aus $G^*(s_r)$ können wir hingegen durch Anwendung der Dreiecksungleichung eine untere Schranke aufstellen. Vergleicht man diese beiden Schranken, so erhält man folgende Ungleichung:

$$\sum_{s_b \in G^*(s_r)} RI_{s_b}(l^*) > 2 \cdot RI_{s_r}(l^*)$$

Für jeden roten Sender s_r gilt also, dass die Sender in seiner Wächter-Menge mindestens doppelt so viel Interferenz auf l^* ausüben wie er selbst. Da diese Gleichung für alle roten Sender gilt und die zugehörigen Wächter-Mengen disjunkt sind, folgt, dass die von allen blauen Sendern ausgehende, auf l^* wirkende Interferenz mindestens doppelt so hoch ist wie die von den roten Sender ausgehende Interferenz. Es gilt also $a_{ALG}(l^*) < 1/2 \cdot a_{OPT_2}(l^*)$. Außerdem ist $a_{OPT_2}(l^*) \leq 1$, da OPT eine gültige Lösung ist. Folglich gilt $a_{ALG}(l^*) \leq 1/2$. Somit ist die Affektiertheit von l^* kleiner als $2/3$ und l^* wurde nicht in Zeile 5 von Algorithmus 1 gelöscht. Dies ist ein Widerspruch dazu, dass s^* ein Sender aus OPT_2 ist.

Lemma 8. *Die Approximationsrate von Algorithmus 1 ist konstant.*

Beweis Wir erhalten dies unmittelbar aus der Kombination von Lemma 5 und 7. Es gilt $OPT \leq OPT' + ALG \leq (1 + \rho_1 + \rho_2) \cdot ALG = \rho \cdot ALG$, wobei $\rho = 1 + \rho_1 + \rho_2$. □

4.2 Scheduling

Wie bereits erwähnt wird eine Master-Slave-Strategie für einen Approximationsalgorithmus für das Problem Scheduling verwendet, das heißt dass der im Folgenden vorgestellte Algorithmus auf dem Algorithmus 1 zurückgreift, um die Ausgabe zu berechnen.

Algorithmus 2 : Scheduling

Eingabe : $L = \{l_1, \ldots l_n\}$
Ausgabe : $(S_1, \ldots, S_T)$, $S_i \subseteq L$

```
1 t := 0
2 Wiederhole
3 |   t := t + 1
4 |   S_t := One-Slot-Scheduling(L)
5 |   L := L \ S_t
6 Bis L = ∅
7 Gebe S zurück
```

Die Korrektheit dieses Algorithmus folgt unmittelbar aus der Korrektheit des Algorithmus 1. Wir müssen uns an dieser Stelle also nur noch Gedanken über die Approximationsrate von Algorithmus 2 machen.

Lemma 9. *Die Approximationsrate von Algorithmus 2 ist in $\mathcal{O}(log(n))$.*

Beweis Zum Beweis dieser Aussage benutzen wir das Konstrukt der Kosteneffektivität, das häufig im Kontext von Set Cover Anwendung findet [5]. Für jede Iteration t des Algorithmus 2 legen wir die Kosten von S_t auf 1 fest. Die Kosteneffektivität von S_t definieren wir als die durchschnittlichen Kosten $p(l_i)$, für die ein Element l_i in S_t in den

Zeitplan aufgenommen wird. Es ist also $p(l_i) = 1/|S_t|$. Man beachte, dass die Kosteneffektivität einer Menge im Laufe des Verfahrens monoton steigend ist, da die Anzahl der Elemente in $|S_t|$ monoton fällt. Die Gesamtkosten des optimalen Zeitplanes sind also $\sum_{i=1}^{n} p(l_i) = \sum_{t=1}^{T} 1 = OPT$.

Nun sei $l_1, \cdots, l_n$ die Reihenfolge, in der die Verbindung von Algorithmus 2 auf den Zeitplan gesetzt werden. Sollten mehrere Verbindung im selben Zeitabschnitt realisiert werden, so können diese in der angegeben Reihenfolge willkürlich sortiert aufgelistet werden.

Da alle Verbindung mit den Kosten OPT realisiert werden können, können auch zu jedem Zeitpunkt in Algorithmus 2 die verbleibenden Verbindung mit Kosten OPT realisiert werden. Zu dem zu untersuchenden Zeitpunkt seien nun die ersten $i-1$ Verbindungen bereits im Plan enthalten. Wir behaupten, dass unter allen Mengen von Verbindungen, die in einem Zeitabschnitt realisiert werden können, eine mit einer Kosteneffektivität von höchstens

$$\frac{OPT}{n-i+1}$$

existiert. Dabei ist $n-i+1$ die Anzahl der noch nicht eingeplanten Verbindungen ist. Angenommen eine solche Menge existiert nicht. Somit gilt für alle S, dass $|S| < OPT/(n-i+1)$. Verwenden wir nun OPT dieser Menge, um weitere Verbindungen auf den Zeitplan zu setzen, so decken wir $OPT \cdot |S| < n-i+1$ der verbleibenden Verbindungen ab. Dies sind jedoch nicht alle verbleibenden Verbindungen und das widerspricht der Tatsache, dass stets alle verbleibenden Verbindungen mit den Kosten OPT realisiert werden können. Folglich existiert eine Menge von Verbindungen mit Kosteneffektivität von höchstens $OPT/(n-i+1)$.

Da nach Lemma 8 die optimale Menge von Verbindung für den nächsten Zeitabschnitt maximal um den Faktor ρ größer ist als die Menge, die Algorithmus 1 wählt, gilt

$$p(l_i) \leq \rho \cdot \frac{OPT}{n-i+1}.$$

Die Gesamtkosten berechnen sich also als

$$\sum_{i=1}^{n} p(l_i) \leq (1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n}) \cdot \rho \cdot OPT = \mathcal{O}(log(n)) \cdot OPT.$$

□

Literatur

1. Goussevskaia, O., Oswald, Y.A., Wattenhofer, R.: Complexity in Geometric SINR. In: MobiHoc'07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing,, ACM (September 2007) 100–109
2. Karp, R.M.: Reducibility Among Combinatorial Problems. In Miller, R.E., Thatcher, J.W., eds.: Complexity of Computer Computations, Plenum Press (1972) 85–103
3. Völker, M., Katz, B., Wagner, D.: On the Complexity of Scheduling with Power Control in Geometric SINR. Technical report, ITI Wagner, Fakultät für Informatik, Universität Karlsruhe (TH) (2009)
4. Goussevskaia, O., Halldórsson, M.M., Wattenhofer, R., Welzl, E.: Capacity of Arbitrary Wireless Networks. In: INFOCOM 2009. (2009)
5. Vazirani, V.V.: Approximation Algorithms. Springer (2004)