- Bachelorarbeit -

Ein Approximationsalgorithmus für das metrische unkapazitierte Facility Location Problem

vorgelegt von
Florian Uhl

1. Gutachter : Dr. habil. Marco Lübbecke
2. Gutachter : Prof. Dr. Stefan Ulbrich

August 2010

Fachbereich Mathematik
der Technischen Universität Darmstadt

Erklärung zur Bachelorarbeit

Hiermit versichere ich, die vorliegende Bachelorarbeit ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 8.August 2010

(Florian Uhl)

Inhaltsverzeichnis

1. Einleitung

Sowohl private als auch staatliche Unternehmen stehen immer wieder vor der Entscheidung, an welchen Standorten sie neue Versorgungseinrichtungen, sogenannte Facilities, eröffnen sollen. Als Beispiel sei hier die Platzierung von Fabriken, Krankenhäusern oder Energielieferanten zur optimalen Versorgung von Städten erwähnt. Wird die Entscheidung der Unternehmen nur unter Berücksichtigung der Eröffnungskosten bzw. Betriebskosten der Facilities und der Verbindungskosten der Facilities mit den Kunden getroffen, so versucht das Unternehmen eine Instanz des sogenannten Facility Location Problems zu lösen.
Mit diesem Problem beschäftigt sich die Forschung schon seit den frühen 60er-Jahren (siehe [6]). Bisher konnte jedoch noch kein Algorithmus gefunden werden, der dieses Problem in polynomieller Zeit löst. Jain und Vazirani haben einen Algorithmus entwickelt, der für das metrische unkapazitierte Facility Location Problem eine approximative Lösung berechnet (siehe [3]). Dieser Algorithmus wird in der vorliegenden Arbeit vorgestellt und analysiert. Er basiert auf dem Primal-Dual-Verfahren.

2. Primal-Dual-Verfahren

Das Primal-Dual-Verfahren wird aufbauend auf die Darstellung von Vazirani (siehe [7]) erläutert. Zum besseren Verständnis werden vorab die Definition des dualen Problems und der Satz vom schwachen komplementären Schlupf wiederholt.

2.1. Grundlagen der linearen Optimierung

Definition 2.1. *Das zu*

$$min \quad c^T x \quad s.t. \quad Ax \geq b,\ x \geq 0 \quad (P)$$

duale Problem kann geschrieben werden in der Form

$$max \quad b^T y \quad s.t. \quad A^T y \leq c,\ y \geq 0 \quad (D).$$

Satz 2.2. ***(Satz vom schwachen komplementären Schlupf)***
Betrachte das Paar zueinander gehörender dualer Probleme (P) und (D) aus Definition 2.1. Die Vektoren $\tilde{x}$ und $\tilde{y}$ seien zulässig für (P) bzw. (D). Dann sind folgende Aussagen äquivalent:

a.) $\tilde{x}$ ist optimal für (P) und $\tilde{y}$ ist optimal für (D).

b.) Für alle Komponenten der Duallösung $\tilde{y_i}$ gilt:

$$\tilde{y_i} = 0 \ oder \ (A\tilde{x})_i = b_i.$$

Für alle Komponenten der Primallösung $\tilde{x_j}$ gilt:

$$\tilde{x_j} = 0 \ oder \ (A^T\tilde{y})_j = c_j.$$

Beweis: Siehe [5]. □

2.2. Primal-Dual-Verfahren

Die Idee des Primal-Dual-Verfahrens besteht darin, ausgehend von einer dual zulässigen Lösung eine primal zulässige Lösung zu konstruieren, ohne dabei die Bedingungen des Satzes vom schwachen komplementären Schlupf (Satz 2.2) zu verletzen. Für lineare Programme, die bekanntlich in polynomieller Zeit lösbar sind, können diese Bedingungen tatsächlich erfüllt werden. Für ganzzahlige Programme ist dies jedoch nicht zu erwarten.

Definition 2.3. *Sei P ein Optimierungsproblem und A ein Algorithmus zur Lösung von P. Des Weiteren bezeichne $OPT(I)$ den Optimalwert einer Instanz $I \in P$ und $A(I)$ den Wert der Lösung des Algorithmus A.*
*Dann heißt der Algorithmus A ε-**Approximations-Algorithmus**, sofern A polynomielle Laufzeit hat und sofern ein $\varepsilon \geq 1$ existiert, so dass gilt:*

$$\frac{1}{\varepsilon}\, OPT(I) \leq A(I) \leq \varepsilon\, OPT(I) \text{ für alle } I \in P.$$

*Ein ε-Approximations-Algorithmus hat **Approximationsgüte** ε.*

Allerdings eignet sich das Primal-Dual-Verfahren zur Entwicklung von Approximationsalgorithmen. Hierzu relaxiert man mindestens eine der beiden Bedingungen des Satzes vom komplementären Schlupf (Satz 2.2.). Dies würde bei dem Problem aus Definition 2.1. folgendermaßen aussehen:

Relaxierte primale Schlupfbedingungen

Für alle Komponenten der Primallösung $\tilde{x_j}$ gilt:

$$\tilde{x_j} = 0 \text{ oder } \frac{c_j}{\alpha} \leq (A^T \tilde{y})_j \leq c_j. \qquad \text{(RPS)}$$

Relaxierte duale Schlupfbedingungen

Für alle Komponenten der Duallösung $\tilde{y_i}$ gilt:

$$\tilde{y_i} = 0 \text{ oder } b_i \leq (A\tilde{x})_i \leq \beta\, b_i. \qquad \text{(RDS)}$$

Hierbei: $\alpha, \beta \geq 1$ und $\alpha, \beta \in \mathbb{R}$.
Wird im Folgenden vom Relaxierungsfaktor der primalen bzw. dualen Schlupfbedingungen gesprochen, so ist hiermit der Faktor α bzw. β gemeint, mit dem die primalen bzw. dualen Schlupfbedingungen relaxiert wurden.

Die Entwicklung eines Approximationsalgorithmus mit Hilfe des Primal-Dual-Verfahrens kann insbesondere für NP-schwere Probleme interessant sein, da diese bekanntlich nicht in polynomieller Zeit lösbar sind, sofern nicht NP $\subset$ P gilt.

Ein mit Hilfe des Primal-Dual-Verfahrens entwickelter Approximationsalgorithmus zur Lösung eines NP-schweren ganzzahligen linearen Optimierungsproblems würde nun wie folgt aussehen:

Zu Anfang wird die LP-Relaxation zu diesem Problem und das dazu gehörige duale Problem bestimmt. Anschließend startet der Algorithmus mit einer für das Problem unzulässigen ganzzahligen primalen und einer zulässigen dualen Lösung, meist mit $x = 0$ und $y = 0$. Danach wird schrittweise die primale Lösung so verändert, dass sie jederzeit ganzzahlig ist und am Ende eine primal zulässige ganzzahlige Lösung entsteht. Hierbei kann eine Veränderung der primalen Lösung auch immer zu einer Veränderung der dualen Lösung führen, da stets die relaxierten Schlupfbedingungen erfüllt sein müssen und da die duale Lösung immer zulässig sein muss. Wurde eine primal zulässige ganzzahlige Lösung gefunden, so terminiert der Algorithmus.

Wie nah die Lösung eines solchen Approximationsalgorithmus an der Optimallösung des Problems ist, hängt, wie wie wir nun sehen werden, von den Relaxierungsfaktoren der primalen und dualen Schlupfbedingungen ab.

Satz 2.4. *Sei P ein diskretes Optimierungsproblem und A ein polynomieller Algorithmus zur approximativen Lösung von P. Berechnet A eine primal zulässige Lösung und eine für das duale Problem der linearen Relaxation des Optimierungsproblems zulässige Lösung und bezeichnet $A(PrimalI)$ den Wert der primalen Lösung des Algorithmus A für die Instanz $I \in P$ und $A(DualI)$ den Wert der dualen Lösung des Algorithmus A für die Instanz I, dann gilt:*

a.) Ist P ein Minimierungsproblem und existiert ein $\varepsilon \geq 1$, so dass gilt:

$$A(PrimalI) \leq \varepsilon \; A(DualI) \text{ für alle } I \in P, \qquad (*)$$

so ist A ein ε-Approximations-Algorithmus für dieses Problem.

b.) Ist P ein Maximierungsproblem und existiert ein $\varepsilon \geq 1$, so dass gilt:

$$\frac{1}{\varepsilon} \; A(DualI) \leq A(PrimalI) \text{ für alle } I \in P, \qquad (**)$$

so ist A ein ε-Approximations-Algorithmus für dieses Problem.

Beweis: Da A ein polynomieller Algorithmus zur Lösung von P ist, genügt es nach Definition 2.3. in beiden Fällen zu zeigen, dass ein $\varepsilon \geq 1$ existiert, so dass gilt:

$$\frac{1}{\varepsilon} \; OPT(I) \leq A(I) \leq \varepsilon \; OPT(I) \text{ für alle } I \; \in P.$$

a.) Bei P handelt es sich um ein Minimierungsproblem.

Da A eine primal zulässige Lösung für I und eine für die lineare Relaxation von I dual zulässige Lösung berechnet, gilt auf Grund der schwachen Dualität:

$$A(DualI) \leq OPT(I) \leq A(PrimalI) = A(I)$$

Daraus folgt:

$$\frac{1}{\varepsilon}\ OPT(I) \leq \frac{1}{\varepsilon}\ A(PrimalI) \overset{wegen\ (*)}{\leq} \frac{1}{\varepsilon}\ \varepsilon\ A(DualI) = \ A(DualI) \leq A(I)$$

Außerdem gilt:

$$A(I) = A(PrimalI) \overset{wegen\ (*)}{\leq} \varepsilon\ A(DualI) \leq \varepsilon\ OPT(I)$$

Also gilt:

$$\frac{1}{\varepsilon}\ OPT(I) \leq A(I) \leq \varepsilon\ OPT(I)$$

b.) Analog.

□

Theorem 2.5. *Sei A ein Algorithmus, welcher in polynomieller Zeit für das Problem*

$$min \quad c^T x \quad s.t. \quad Ax \geq b,\ x \geq 0,\ x \in \mathbb{Z} \quad (2.5P)$$

eine zulässige Lösung $A(PrimalI)$ berechnet. Des Weiteren berechne A für das duale Problem der linearen Relaxation dieses Problems, also für

$$max \quad b^T y \quad s.t. \quad A^T y \leq c,\ y \geq 0 \quad (2.5D)$$

eine zulässige Lösung $A(DualI)$. Genügen nun die Lösungen $A(PrimalI)$ und $A(DualI)$ den relaxierten Schlupfbedingungen des zugehörigen linearen Problems, so gilt:

A ist ein $\alpha\beta$- Approximations-Algorithmus für dieses Problem.

Beweis: Da $A(PrimalI)$ eine zulässige Lösung ist, gilt: $OPT(I) \leq A(PrimalI) = A(I)$.
Im Folgenden bezeichne $\tilde{x}$ die Lösung von (2.5P) von Algorithmus A und $\tilde{y}$ die Lösung von (2.5D) von Algorithmus A.
Da $\tilde{x}$ eine zulässige Lösung von (2.5P) ist, ist $\tilde{x}$ insbesondere zulässig für:

$$min \quad c^T x \quad s.t. \quad Ax \geq b,\ x \geq 0.$$

Weil die beiden berechneten Lösungen den relaxierten Schlupfbedingungen genügen, gilt:

$$A(PrimalI) = c^T \tilde{x} \overset{(RPS)}{\leq} \alpha\ (A^T \tilde{y})^T \tilde{x} = \alpha \tilde{y}^T A \tilde{x} \overset{(RDS)}{\leq} \alpha\beta \tilde{y}^T b = \alpha\beta b^T \tilde{y} = \alpha\beta A(DualI).$$

Die Behauptung folgt nun aus Satz 2.4. □

3. Das unkapazitierte Facility Location Problem

Beim Facility Location Problem geht es darum, die Versorgung einer bestimmten Menge von Städten möglichst kostengünstig zu gestalten. Die Versorgung der Städte funktioniert hierbei über sogenannte Facilities, die an verschiedenen Standorten errichtet werden können. Die Gesamtkosten des Problems ergeben sich aus den Eröffnungkosten für eine bzw. mehrere Facilities und aus den Verbindungskosten, die für die Verbindung einer bestimmten Stadt mit einer bestimmten Facility anfallen. Die Eröffnungkosten hängen ebenso wie die Verbindungskosten vom jeweiligen Standort ab. Beim Facility Location Problem kann jede Stadt mit jeder möglichen Facility verbunden werden.

Im Folgenden betrachten wir das unkapazitierte Facility Location Problem, bei welchem gewährleistet ist, dass jede Facility alle Städte komplett versorgen kann.

3.1. Modellierung des Problems

Das unkapazitierte Facility Location Problem lässt sich mit Hilfe eines vollständigen bipartiten Graphen $G = (V, E)$ darstellen. Hierbei gilt: $V = F \cup C$ und $E = F \times C$, wobei F die Menge aller möglichen Facilities und C die Menge aller Städte ist. Des Weiteren bezeichne f_i die Eröffnungskosten einer Facility i und c_{ij} die Verbindungskosten der Stadt j mit der Facility i. Das Problem ist nun, eine Menge $I \subseteq F$ von offenen Facilities zu finden, sowie eine Funktion $\varphi : C \to I$, die jeder Stadt eine geöffnete Facility zuordnet, so dass die Gesamtkosten minimal sind.
Für alle $i \in F$ und für alle $j \in C$ gilt: $f_i, c_{ij} \in \mathbb{R}^+$
Die Gesamtkosten dieses Problems sind somit

$$\sum_{i \in I} f_i + \sum_{\varphi(j)=i} c_{ij}.$$

Gelte nun für alle $j \in C$ und $i \in F$:

$$x_{ij} := \begin{cases} 1, & \text{falls} \quad \varphi(j) = i \\ 0, & \text{falls} \quad \varphi(j) \neq i \end{cases},$$

dann gibt diese Binärvariable an, ob die Stadt j mit der Facility i verbunden ist oder nicht.

Sei nun für alle $i \in F$ y_i wie folgt definiert:

$$y_i := \begin{cases} 1, & \text{falls} \quad i \in I \\ 0, & \text{falls} \quad i \notin I \end{cases},$$

dann gibt y_i an, ob die Facility i eröffnet wurde oder nicht.

Folglich können die Gesamtkosten umgeschrieben werden zu:

$$\sum_{i \in F, j \in C} c_{ij} x_{ij} + \sum_{i \in F} f_i y_i.$$

Um nun das unkapazitierte Facility Location Problem als IP aufschreiben zu können, muss mit Hilfe von Nebenbedingungen sichergestellt werden, dass gilt:

- Jede Stadt j ist mit einer Facility i verbunden, d.h. für jede Stadt j existiert eine Facility i mit $x_{ij} = 1$.
- Wurde eine Stadt j mit Facility i verbunden, so gilt: $y_i = 1$.
- x_{ij} und y_i können nur die Werte 0 oder 1 annehmen.

Lemma 3.1. *Gilt*

$$\sum_{i \in F} x_{ij} \geq 1, \forall j \in C.$$

Dann folgt:

Für jede Stadt j existiert eine Facility i mit $x_{ij} > 0$
bzw. für alle $j \in C$ existiert ein $i \in F$, so dass gilt: $x_{ij} > 0$.

Beweis: Angenommen eine Stadt j wäre mit keiner Facility i verbunden, so würde gelten:

$$x_{ij} = 0, \ \forall i \in F.$$

Dies stünde im Widerspruch dazu, dass für alle $j \in C$ gelten muss:

$$\sum_{i \in F} x_{ij} \geq 1.$$

□

Lemma 3.2. *Gilt $y_i - x_{ij} \geq 0$, so gilt: $x_{ij} > 0 \Rightarrow y_i > 0$.*

Beweis: Klar. □

Somit lässt sich das unkapazitierte Facility Location Problem wie folgt darstellen:

$$\begin{array}{lll} min & \sum_{i \in F, j \in C} c_{ij} x_{ij} + \sum_{i \in F} f_i y_i & \\ s.t. & \sum_{i \in F} x_{ij} \geq 1 & , \ j \in C \\ & y_i - x_{ij} \geq 0 & , \forall i \in F, \ j \in C \\ & x_{ij} \in \{0,1\} & , \forall i \in F, \ j \in C \\ & y_i \ \in \{0,1\} & , \forall i \in F. \end{array} \tag{3.1}$$

3.2. Komplexität des Facility Location Problems

Im Folgenden wird gezeigt, dass es sich bei dem unkapazitierten Facility Location Problem um ein NP-schweres Problem handelt.
Hierzu wird als Erstes das Minimum „Weight Set Cover Problem“ betrachtet, bei dem es sich um ein NP-schweres Problem handelt (siehe [4]).

Definition 3.3. *Instanzen des „Minimum Weight Set Cover Problems“ bestehen aus einer endlichen Menge X, einer Menge F von Teilmengen von X und Gewichten c: $F \to \mathbb{R}^+$. Für F gilt zusätzlich, dass die Vereinigung der Elemente S in F die Menge X überdeckt, also*

$$F \subseteq \mathcal{P}(X) : \bigcup_{S \in F} S = X.$$

Ziel ist es nun, eine Menge $R \subseteq F$ mit minimalen Gewicht zu finden, für die gilt:

$$\bigcup_{S \in R} S = X.$$

Satz 3.4. *Bei dem „Minimum Weight Set Cover Problem“ handelt es sich um ein NP-schweres Problem.*

Beweis: Siehe [4]. □

Um das „Minimum Weight Set Cover Problem“ modellieren zu können, wird zunächst eine Hilfsvariable x_S eingeführt. Dazu gelte für alle $S \in F$:

$$x_S := \begin{cases} 1, & \text{falls} \quad S \in R \\ 0, & \text{falls} \quad S \notin R \end{cases}.$$

Mit Hilfe dieser Binärvariablen und Lemma 3.1 folgt, dass sich das „Minimum Weight Set Cover Problem“ darstellen lässt durch:

$$\begin{aligned} &min && \sum_{S \in F} c(S) x_S && && \\ &s.t. && \sum_{S : x \in S} x_S \geq 1, && \forall x \in X && (3.2a) \\ & && x_S \in \{0,1\}, && \forall S \in F. && \end{aligned}$$

Hierbei stellt die erste Nebenbedingung sicher, dass gilt: $\bigcup_{S \in R} S = X,$ wobei gilt:
$x_S = 1 \Leftrightarrow S \in R$.

Theorem 3.5. *Das unkapazitierte Facility Location Problem ist ein NP-schweres Problem.*

Beweis: Um die Behauptung zu zeigen, genügt es, jede Instanz des „Minimum Weight Set Cover Problem“ in eine Instanz des unkapazitierten Facility Location Problems umzuschreiben.
Wäre nämlich das unkapazitierte Facility Location Problem kein NP-schweres Problem, so dürfte demnach auch das „Minimum Weight Set Cover Problem“ kein NP-schweres Problem sein. Dies wäre aber widersprüchlich zu Satz 3.4.

Im Folgenden bezeichne nun F' die Indexmenge von F, d.h. $F' := \{1, \ldots, |F|\}$.
Sei nun: $f_i := c(S_i)$, für alle i in F', wobei S_i die i-te Teilmenge von F bezeichnet.

Dann ist (3.2a) äquivalent zu:

$$\begin{aligned} &min && \sum_{i \in F'} f_i y_i && && \\ &s.t. && \sum_{i : x \in S_i} y_i \geq 1, && \forall x \in X && (3.2b) \\ & && y_i \in \{0,1\}, && \forall i \in F'. && \end{aligned}$$

Sei nun X' die Indexmenge von X und sei x_j das j-te Element von X, dann kann (3.2b) umgeschrieben werden zu:

$$\begin{aligned} &min && \sum_{i \in F'} f_i y_i && && \\ &s.t. && \sum_{i : x_j \in S_i} r_{ij} \geq 1, && \forall j \in X' && (3.2c) \\ & && y_i - r_{ij} \geq 0 && , \forall i \in F',\ j \in X' && \\ & && r_{ij} \in \{0,1\} && , \forall i \in F',\ j \in X' && \\ & && y_i \in \{0,1\}, && \forall i \in F'. && \end{aligned}$$

Dies ist möglich, da die erste Nebenbedingung nach Lemma 3.1. sicherstellt, dass für alle $j \in X'$ ein i existiert, so dass gilt: $r_{ij} = 1$. Durch die zweite Nebenbedingung wird gewährleistet, dass dann für das entsprechende i gelten muss: $y_i = 1$. Die zweite Nebenbedingung stellt somit sicher, dass $r_{ij} = 1$ nur dann gelten kann, wenn S_i zur Überdeckung verwendet wird. Mit der ersten Nebenbedingung wird gewährleistet, dass X komplett überdeckt wird.

Gilt:

$$c_{ij} := \begin{cases} 0, & \text{falls} \quad x_j \in S_i \\ \infty, & \text{falls} \quad x_j \notin S_i \end{cases},$$

dann ist die Lösung des folgenden Programms, sofern sie endlich ist, ebenfalls eine Lösung von (3.2a),

$$
\begin{aligned}
min \quad & \sum_{i \in F', j \in X'} c_{ij} r_{ij} + \sum_{i \in F'} f_i y_i & & \\
s.t. \quad & \sum_{i \in F'} r_{ij} \geq 1 & , \ j \in X' & \qquad (3.2d) \\
& y_i - r_{ij} \geq 0 & , \forall i \in F', \ j \in X' & \\
& r_{ij} \in \{0,1\} & , \forall i \in F', \ j \in X' & \\
& y_i \ \in \{0,1\} & , \forall i \in F' &
\end{aligned}
$$

da durch die Definition von c_{ij} sichergestellt wird, dass gilt: $r_{ij} = 1 \Leftrightarrow x_j \in S_i$.
Ist die Lösung des in (3.2d) beschriebenen Problems nicht endlich, so ist das ursprüngliche „Minimum Weight Set Cover Problem“ nicht lösbar.
Bei Modell (3.2d) handelt es nun um ein unkapazitiertes Facility Location Problem. Also lässt sich jede Instanz des „Minimum Weight Set Cover Problems“ in eine Instanz des unkapazitiertes Facility Location Problems umschreiben, weswegen die Behauptung aus der obigen Begründung folgt. □

Damit wissen wir nun, dass das unkapazitierte Facility Location Problem nicht in polynomieller Zeit lösbar ist, sofern nicht gilt: $NP \subset P$. Folglich stellt sich die Frage, ob nicht mit Hilfe des Primal-Dual-Verfahrens ein Approximationsalgorithmus mit einer niedrigen Approximationsgüte entwickelt werden kann.

3.3. Relaxation

Um das Primal-Dual-Verfahren zur Lösung des unkapazitierten Facility Location Problems anwenden zu können, bedarf es einer Relaxation des ganzzahligen Programms (3.1). Diese erfolgt durch Weglassen der Ganzzahligkeitsbedingung.
Die Nebenbedingungen $x_{ij} \leq 1$ und $y_i \leq 1$ können hierbei ebenfalls weggelassen werden, da

$$
\begin{aligned}
& c_{ij} \geq 0, \ \forall i \in F, \ j \in C \\
& f_i \geq 0, \ \forall i \in F
\end{aligned}
$$

gilt und es somit nicht sinnvoll wäre, $x_{ij} > 1$ bzw. $y_i > 1$ zu wählen.

Damit ergibt sich als LP:

$$
\begin{array}{lll}
min & \sum_{i\in F, j\in C} c_{ij}x_{ij} + \sum_{i\in F} f_i y_i & \\
s.t. & \sum_{i\in F} x_{ij} \geq 1 & , \ j \in C \\
 & y_i - x_{ij} \geq 0 & , \forall i \in F, \ j \in C \\
 & x_{ij} \geq 0 & , \forall i \in F, \ j \in C \\
 & y_i \ \geq 0 & , \forall i \in F.
\end{array} \tag{3.3}
$$

Das duale Programm zu (3.3) ist:

$$
\begin{array}{lll}
max & \sum_{j\in C} \alpha_j & \\
s.t. & \alpha_j - \beta_{ij} \leq c_{ij} & , \forall i \in F, \ j \in C \\
 & \sum_{j\in C} \beta_{ij} \leq f_i & , \forall i \in F \\
 & \alpha_j \geq 0 & , \forall j \in C \\
 & \beta_{ij} \geq 0 & , \forall i \in F, j \in C.
\end{array} \tag{3.4}
$$

Für eine optimale Lösung des linearen Problems (3.3) müsste nun nach dem Satz vom komplementären Schlupf (Satz 2.2) gelten:

$$(x, y) \text{ ist für (3.3) zulässig.} \tag{3.5}$$

$$(\alpha, \beta) \text{ ist für (3.4) zulässig.} \tag{3.6}$$

$$x_{ij} > 0 \Rightarrow \alpha_j - \beta_{ij} = c_{ij}. \tag{3.7}$$

$$y_i > 0 \Rightarrow \sum_{j\in C} \beta_{ij} = f_i. \tag{3.8}$$

$$\alpha_j > 0 \Rightarrow \sum_{i\in F} x_{ij} = 1. \tag{3.9}$$

$$\beta_{ij} > 0 \Rightarrow y_i - x_{ij} = 0. \tag{3.10}$$

3.4. Anschauliche Interpretation der Dualvariablen

Mit Hilfe der Bedingungen (3.5) - (3.10) wird im Folgenden eine anschauliche Interpretation der Dualvariablen hergeleitet:

Wenn eine Facility i eröffnet wurde, muss nach Bedingung (3.8) gelten:

$$\sum_{j \in C} \beta_{ij} = f_i.$$

Somit müssen die β_{ij} zusammen die Eröffnungskosten der Facility i tragen. Damit $\beta_{ij} > 0$ sein kann, muss aber nach Bedingung (3.10) gelten:

$$y_i - x_{ij} = 0.$$

Folglich können die Eröffnungskosten der Facility nur von den β_{ij} getragen werden, bei denen die Stadt j auch tatsächlich mit der Facility i verbunden ist (Dabei ist zu beachten, dass Eröffnungskosten natürlich nur dann auftreten, wenn die Facility i eröffnet wurde, also sofern gilt: y_i=1).
Wegen Bedingung (3.7) muss nun bei diesen Städten gelten:

$$\alpha_j - \beta_{ij} = c_{ij}.$$

Auf Grund der starken Dualität gilt für die Optimallösung (x, y) von (3.3):

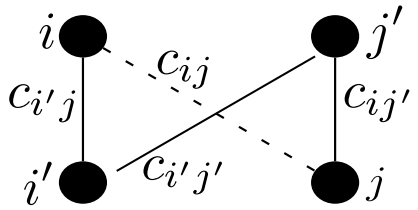
$$\sum_{i \in I} f_i + \sum_{\varphi(j)=i} c_{ij} = \sum_{j \in C} \alpha_j.$$

Daraus folgt, dass wir α_j als die Kosten interpretieren können, die von Stadt j bezahlt werden.
Hierbei bezahlt jede Stadt mindestens c_{ij}, wenn sie mit der Facility i verbunden ist, und beteiligt sich mit β_{ij} an den Eröffnungskosten der Facility i.

4. Approximationsalgorithmus von Jain und Vazirani

Der Approximationsalgorithmus für das metrische unkapazitierte Facility Location Problem von Jain und Vazirani (siehe [3]) besteht aus zwei Phasen und baut auf den Relaxationen aus Kapitel 3 auf. Da es sich um ein *metrisches* unkapazitiertes Facility Location Problem handelt, gilt jetzt noch zusätzlich für alle $(i, j) \in E$:

$$c_{ij} \leq c_{i'j} + c_{i'j'} + c_{ij'},\ \forall j' \in C, \forall i' \in F$$



Bemerkung 4.1. *Anschaulich interpretiert bedeutet diese Ungleichung, dass die Verbindungskosten einer Facility i mit einer Stadt j nicht höher sein dürfen als die Kosten, die entstehen würden, wenn man die Stadt j mit der Facility i über eine Facility i′ und eine Stadt j′ verbinden würde. Dies macht auch Sinn, da man als Unternehmen zur Bestimmung der Verbindungskosten einer Facility mit einer Stadt immer die Kosten heranziehen würde, die auf dem kostengünstigsten Weg entstehen.*

Die Lösung des Algorithmus, der auf dem Primal-Dual-Verfahren aufbaut, genügt fast allen Optimalitätsbedingungen des linearen Problems. Lediglich Bedingung (3.7) wird nicht von allen $j \in C$ erfüllt, sondern nur von allen $j \in D \subset C$. Hierbei bezeichnet D die Menge aller Städte $j \in C$, die in Phase 2 als *direkt verbunden* deklariert wurden.
Für alle $j \in D^c$ wird nur gewährleistet, dass gilt:

$$x_{ij} > 0 \Rightarrow \frac{c_{ij}}{3} \leq \alpha_j - \beta_{ij} \leq 3\ c_{ij} \quad (4.1)$$

Ist $j \in D^c$, so wurde j in Phase 2 als *indirekt verbunden* bezeichnet.

4.1. Phase 1

Am Anfang der Phase 1 des Algorithmus werden alle Städte $j \in C$ als *unverbunden* markiert und es gilt:

$$(x, y) = (0, 0)$$
$$(\alpha, \beta) = (0, 0)$$

d.h. der Algorithmus beginnt mit einer primal unzulässigen (erste Nebenbedingung von (3.1) ist verletzt) und einer dual zulässigen Lösung.

Nun wird, solange nicht alle Städte *verbunden* sind, die Dualvariable α_j jeder unverbundenen Stadt j erhöht. Die Erhöhung der α_j findet hierbei gleichzeitig statt und es werden alle α_j mit derselben Rate erhöht.

Sobald für ein $j' \in C$ gilt:

$$\alpha_{j'} = c_{ij'}$$

wird $\beta_{ij'}$ ab sofort mit den α_j zusammen und um denselben Wert erhöht. Zusätzlich wird die Kante (i, j') als *bezahlt* markiert.

Gilt: $\beta_{ij} > 0$, so wird die Kante (i, j) zusätzlich als *speziell* markiert.

Erläuterung 4.2. *Wurde die Kante (i, j) als speziell markiert, so muss auf Grund der Bedingung (3.10) darauf geachtet werden, dass gilt: $x_{ij} = y_i$.*
Würde $\beta_{ij'}$ nicht mit $\alpha_{j'}$ zusammen und mit derselben Rate erhöht werden, sobald die Kante (i, j') als bezahlt markiert wurde, so wäre (α, β) keine dual zulässige Lösung mehr. Die erste Nebenbedingung von (3.4) wäre nämlich nicht mehr erfüllt. Jede bezahlte Kante (i, j') genügt Bedingung (3.7) bzw. (4.1) für jeden Wert von $x_{ij'}$, da gilt: $\beta_{ij'} = max(\alpha_{j'} - c_{ij'}, 0) = \alpha_{j'} - c_{ij'}$. Dies gilt, da $\beta_{ij'}$ erst mit $\alpha_{j'}$ zusammen erhöht wird, wenn die Kante (i, j') als bezahlt markiert wurde.

Gilt nun für ein $i' \in F$:

$$\sum_{j \in C} \beta_{i'j} = f_{i'},$$

so wird i' als *temporär eröffnet* deklariert. Anschließend wird für alle unverbundenen Städte überprüft, ob sie eine bezahlte Kante zu i' besitzen. Ist dies bei Stadt j der Fall, so wird j als *verbunden* markiert. Des Weiteren wird dann der *Verbindungszeuge* von j, welcher angibt mit welcher Facility die Stadt j verbunden wurde, auf i' gesetzt. Sobald eine Stadt j als verbunden markiert wurde, werden α_j und alle β_{ij} mit $i \in F$ ab sofort nicht mehr erhöht.

Wird eine Kante (i, j) als bezahlt markiert und wurde i bereits temporär eröffnet, so wird j als verbunden markiert, der zugehörige Verbindungszeuge wird auf i gesetzt und es werden α_j und alle $\beta_{i''j}$ mit $i'' \in F$ ab sofort nicht mehr erhöht.

Erläuterung 4.3. *Sobald die Facility $i \in F$ temporär eröffnet wurde, darf kein β_{ij} mit $j \in C$ mehr erhöht werden, da ansonsten gelten würde:*

$$\sum_{j \in C} \beta_{ij} > f_i.$$

Es käme somit zu einer Verletzung der zweiten Nebenbedingung von (3.4). Handelt es sich bei i um eine temporär geöffnete Facility, so wäre die Bedingung (3.8) auch erfüllt, wenn gelten

würde: $y_i > 0$.
Sobald eine Stadt j *mit der temporär eröffneten Facility* i *verbunden wird, darf* α_j *nicht mehr erhöht werden, da* β_{ij} *nicht mehr erhöht werden darf. Würde* α_j *weiter erhöht werden, so würde gelten:* $\alpha_j - \beta_{ij} > c_{ij}$. *Es werden auch alle* $\beta_{i'j}$ *mit* $i' \in F$ *nicht mehr erhöht, sobald* j *mit* i *verbunden wird, denn die Erhöhung von* $\beta_{i'j}$ *resultiert immer aus einer Erhöhung von* α_j . *Prinzipiell könnte* $\beta_{i'j}$ *weiterhin erhöht werden, ohne dass die duale Zulässigkeit verletzt wird. Jedoch könnte es dann im späteren Verlauf zu einer Verletzung der Bedingung (3.7) bzw. (4.1) kommen.*
Da β_{ij} *mit* $i \in F$ *nicht mehr erhöht wird, sobald* $j \in C$ *verbunden ist, gilt weiterhin:* $\beta_{ij} = max(\alpha_j - c_{ij}, 0)$.

Die Phase 1 terminiert, sobald alle Städte als verbunden markiert wurden. Finden mehrere Ereignisse gleichzeitig statt, so können sie in beliebiger Reihenfolge abgearbeitet werden.

Bemerkung 4.4.
Phase 1 liefert uns eine dual zulässige Lösung, da stets gilt (siehe Erläuterung 4.2 und 4.3):

$$\begin{aligned} &\alpha_j - \beta_{ij} \leq c_{ij}, && \forall\ i \in F,\ j \in C \\ &\sum_{j \in C} \beta_{ij} \leq f_i, && \forall i \in F. \end{aligned}$$

Beobachtung
Wenn man nun nach Phase 1 x_{ij} und y_i folgendermaßen definieren würde

$$x_{ij} = \begin{cases} 1, & \text{falls} \quad i \text{ ist Verbindungszeuge von } j \\ 0, & \text{sonst} \end{cases}$$

$$y_i = \begin{cases} 1, & \text{falls} \quad i \text{ ist } \textit{temporär eröffnet} \\ 0, & \text{sonst} \end{cases}$$

so hätte man zwar eine primal und eine dual zulässige Lösung, jedoch würde diese noch nicht der Bedingung (3.10) genügen, denn es kann gelten:

$$\beta_{ij} > 0 \text{ und } x_{ij} = 0 \neq 1 = y_i.$$

Es könnten nämlich eine Facility $i' \in F$ und eine Facility $i \in F$ existieren, mit denen die Stadt $j \in C$ jeweils eine spezielle Kante besitzt. Wurden nun beide Facilities temporär eröffnet, so wäre Bedingung (3.10) nicht mehr erfüllt, da die Stadt j nur einen Verbindungszeugen besitzt.

Bemerkung 4.5. *Interpretiert man die Erhöhung aller zu erhöhender α_j um eine Einheit als einen Fortschritt der „Zeit“ um eine Einheit, so hat Phase 1 eine Art zeitlichen Verlauf. Zu jedem Zeitpunkt entspricht dann in Phase 1 der größte Wert aller α_j der aktuellen „Zeit“, sofern diese zu Beginn auf 0 gesetzt wurde. Jedem Ereignis kann somit eine „Zeit“ zugeordnet werden, die angibt, wann das Ereignis in Phase 1 eintrat.*

4.2. Phase 2

Im Folgenden gelte: $F_t := \{i \in F \mid i \text{ ist temporär eröffnet}\}$.

Nun bezeichne T den Teilgraphen von G, der aus allen Knoten von G und aus allen Kanten besteht, die in Phase 1 als speziell markiert wurden.
Dann wird mit Hilfe von T ein Graph T^2 konstruiert, in dem eine Kante (i, j) genau dann existiert, wenn entweder (i, j) eine Kante in T ist oder wenn ein Knoten $k \in T$ existiert, so dass (i, k) und (k, j) Kanten in T sind. Sei nun H der von F_t induzierte Teilgraph von T^2.
Des Weiteren bezeichne I eine beliebige maximale unabhängige Menge aus H. Mit einer unabhängige Menge aus H ist hierbei eine Menge von Knoten gemeint, für die gilt, dass kein Knoten aus der Menge mit einem anderen Knoten der Menge eine gemeinsame Kante in H besitzt. Ist diese Menge zusätzlich maximal, so bedeutet dies, dass bei Hinzufügen eines weiteren Knotens aus H die Menge nicht mehr unabhängig wäre.
Es werden nun alle $i \in I$ als *geöffnet* markiert.

Erläuterung 4.6. *Alle $i \in I$ erfüllen auf jeden Fall Bedingung (3.8), denn auf Grund der Konstruktion von I gilt:*

$$i \in I \;\Rightarrow\; i \text{ ist Knoten in } H \;\Rightarrow\; i \in F_t \;\Rightarrow i \text{ wurde in Phase 1 temporär eröffnet.}$$

Mit Hilfe von Erläuterung 4.3, folgt damit die Gültigkeit der Bedingung (3.8) für alle $i \in I$.

Für jede Stadt $j \in C$ gelte nun: $G_j := \{i \in F_t \mid (i, j) \text{ ist } \textit{speziell}\}$.

Dann tritt bei jeder Stadt j genau einer der folgenden Fälle ein:

Fall 1: Es existiert eine geöffnete Facility $i \in G_j$.
Dann setze $\varphi(j) = i$ und deklariere j als *direkt verbunden.*

Fall 2: Es existiert keine geöffnete Facility $i \in G_j$.
Dann betrachte die bezahlte Kante (i', j), bei der i' der Verbindungszeuge von j ist. Dabei kann nun genau einer der folgenden Fälle auftreten:

Fall 2a: $i' \in I$.
Setze $\varphi(j) = i'$ und deklariere j als *direkt verbunden.*

Fall 2b: $i' \notin I$.
Sei nun $i'' \in I$ ein Nachbar von i' in H. Dann setze $\varphi(j) = i''$ und deklariere j als *indirekt verbunden.*

Erläuterung 4.7. *Es gilt:*

$$|G_j \cap I| \leq 1 \text{ für alle } j \in C.$$

Wäre dem nicht so, so würden zwei Facilities $i \in I$ und $i' \in I$ mit $i \neq i'$ existieren, die beide eine spezielle Kante zur Stadt j hätten. Folglich würde T die Kanten (i, j) und (i', j) enthalten. Damit wäre aber (i, i') eine Kante in T^2 und somit auch eine Kante in H. Dies steht im Widerspruch dazu, dass gilt: $i \in I$ und $i' \in I$.

Gilt:

$$y_i > 0 \text{ und } i \in G_j,$$

so muss j mit i verbunden werden, damit Bedingung (3.10) erfüllt ist. (Würde j mit i nicht verbunden werden, so würde gelten: $\beta_{ij} > 0$ und $y_i \neq x_{ij}$.)

Wurde nun jede Stadt $j \in C$ direkt oder indirekt verbunden, so wird

$$x_{ij} = \begin{cases} 1, & \text{falls} \quad \varphi(j) = i \\ 0, & \text{falls} \quad \varphi(j) \neq i \end{cases}$$

und

$$y_i = \begin{cases} 1, & \text{falls} \quad i \in I \\ 0, & \text{falls} \quad i \notin I \end{cases}$$

gesetzt.Anschließend terminiert der Algorithmus.

Bemerkung 4.8.

a.) Tritt in Phase 2 des Algorithmus der Fall 2a für die Stadt $j \in C$ ein, so gilt: $\beta_{ij} = 0$.

b.) Ist $j \in C$ indirekt mit $i \in I$ verbunden, so gilt: $\beta_{ij} = 0$.

c.) Für jede Facility $i \in F$ mit $f_i = 0$ gilt: $i \in I$.

Begründung

für a.) Da Fall 2a eintrat, gilt: i war bereits nach Phase 1 *Verbindungszeuge* von j und $i \in I$. Würde nun gelten: $\beta_{ij} > 0$, so wäre (i, j) eine *spezielle* Kante und somit $i \in G_j$. Folglich wäre Fall 2a nicht eingetreten.

für b.) Da es sich bei (i, j) um eine indirekte Verbindung handelt, gilt $i \in I$ (siehe Fall 2b). Würde nun gelten: $\beta_{ij} > 0$, so wäre (i, j) eine *spezielle* Kante und somit würde gelten: $i \in G_j$ und $i \in I$. Dies steht aber im Widerspruch dazu, dass j *indirekt verbunden* ist.

für c.) Gilt für eine Facility $i \in F$: $f_i = 0$, so wurde die Facility i in Phase 1 des Algorithmus temporär eröffnet. Folglich gilt: $i \in F_t$. Da T alle Knoten von G enthält, ist i ebenfalls ein Knoten in T^2 und somit auch ein Knoten in H. Wäre nun i kein Element der Menge I, so müsste i ein Nachbarn in H besitzen. Dies kann jedoch nicht sein, da i keine spezielle Kanten besitzt.

5. Analyse des Algorithmus

In diesem Kapitel wird der Algorithmus von Jain und Vazirani analysiert. Dabei wird die Approximationsgüte und die Laufzeit in Anlehnung an Jain und Vazirani (siehe [3]) bestimmt. Darüber hinaus wird eine Sensitivitätsanalyse vorgenommen.

5.1. Bestimmung der Approximationsgüte

Für die Bestimmung der Approximationsgüte wird im Folgenden Theorem 2.5. angewendet. Dazu muss vorab überprüft werden, ob es sich bei (x, y), bzw. (α, β) um zulässige Lösungen für (3.1), bzw. (3.4) handelt und ob diese Lösungen den relaxierten primalen und dualen Schlupfbedingungen mit bestimmten Relaxierungsfaktoren genügen.
Hierbei werden die Bedingungen (3.7), (3.8), (3.9), (3.10) und (4.1) jeweils einzeln betrachtet.

Satz 5.1. *Der Algorithmus von Jain und Vazirani liefert eine zulässige Lösung für (3.1).*

Beweis: Klarerweise gilt:

$$\begin{aligned} x_{ij} &\in \{0,1\} && ,\forall i \in F,\ j \in C \\ y_i &\in \{0,1\} && ,\forall i \in F. \end{aligned}$$

Folglich bleibt noch zu zeigen, dass für alle $i \in F$ und für alle $j \in C$ gilt: $y_i - x_{ij} \geq 0$ und $\sum_{i \in I} x_{ij} \geq 1$.
Da der Algorithmus erst terminiert, wenn jede Stadt $j \in C$ verbunden worden ist, und da jede Stadt nur genau einmal verbunden wird, gilt sogar für alle Städte $j \in C$: $\sum_{i \in I} x_{ij} = 1$.

Die Ungleichung $y_i - x_{ij} \geq 0$ wird von allen $j \in C$ und von allen $i \in F$ erfüllt, da für den Verbindungszeugen i' von j am Ende von Phase 2 gelten muss: $i' \in I$ bzw. $y_{i'} > 0$.
Es gilt somit: $x_{ij} > 0 \ \Rightarrow \ y_i > 0$. □

Satz 5.2. *Der Algorithmus von Jain und Vazirani liefert uns eine für (3.4) zulässige Lösung.*

Beweis: Da in Phase 2 die Werte von α und β nicht verändert werden, folgt die Behauptung unmittelbar aus Bemerkung 4.4. □

Damit ist die Zulässigkeit der Lösungen gezeigt worden. Im Folgenden wird die Gültigkeit der relaxierten Schlupfbedingungen überprüft.

Lemma 5.3. *Sei $(i, j) \in I \times C$ eine direkte Verbindung. Dann gilt:*

$$\alpha_j - \beta_{ij} = c_{ij.}$$

Beweis: Da es sich bei (i, j) um eine direkte Verbindung handelt, muss in Phase 2 der Fall 1 oder der Fall 2a eingetreten sein. In beiden Fällen war (i, j) mindestens eine bezahlte Kante, weswegen gilt: $\alpha_j \geq c_{ij}$.
Nach Erläuterung 4.3. galt in Phase 1 stets:

$$\beta_{ij} = max(\alpha_j - c_{i'j}, 0),\ \forall i' \in F,\ j \in C.$$

Dies gilt auch noch weiterhin, da es zu keiner Veränderung der Dualvariablen in Phase 2 kam.

Damit gilt also für jede direkte Verbindung $(i, j) \in I \times C$:

$$\alpha_j - \beta_{ij} = \alpha_j - \alpha_j + c_{ij} = c_{ij}.$$

□

Lemma 5.4. *Sei $(i, j) \in I \times C$ eine indirekte Verbindung. Dann gilt:*

$$\frac{c_{ij}}{3} \leq \alpha_j - \beta_{ij} \leq c_{ij}.$$

Beweis: Nach Bemerkung 4.8.b.) gilt: $\beta_{ij} = 0$. Da nach Satz 5.2 (α, β) eine zulässige Lösung für (3.4) ist, folgt damit:

$$\alpha_j \leq c_{ij}.$$

Also bleibt nur noch zu zeigen:

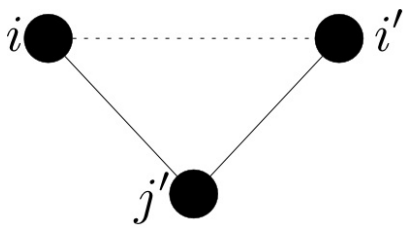
$$\frac{c_{ij}}{3} \leq \alpha_j.$$

Da j indirekt verbunden ist, muss in Phase 2 der Fall 2b eingetreten sein, d.h.:
Es muss eine Facility $i' \notin I$ geben, die ein Nachbar von i in H ist und die am Ende von Phase 1 der Verbindungszeuge von j war.

Also ist $(i, i') \in H$, weshalb nach Konstruktion von H eine Stadt $j' \in C$ existieren muss, für die gilt:

$$(i, j') \text{ und } (i', j') \text{ sind spezielle Kanten.}$$

Somit muss gelten: $\alpha_{j'} \geq c_{ij'}$ und $\alpha_{j'} \geq c_{i'j'}$. $(*)$



Im Folgenden bezeichnen wir nun mit t_i und $t_{i'}$ die Zeiten, zu denen i und i' temporär eröffnet wurden. Mit der Zeit t_i ist hierbei der, bei der temporären Eröffnung von i, größte aufgetretene Wert aller α_j gemeint.
In Phase 1 trat eines der beiden folgenden Ereignisse ein und führte dazu, dass $\alpha_{j'}$ im weiteren Verlauf nicht mehr erhöht wurde. Entweder wurde eine Facility temporär eröffnet, mit der j'

eine bezahlte Kante hat, oder eine Kante von j' zu einer bereits temporär eröffneten Facility wurde als bezahlt markiert.
In jedem Fall gilt, da (i, j') und (i', j') spezielle Kanten sind:

$$\alpha_{j'} \leq \ min(t_i, \ t_{i'}).$$

Weil i' der Verbindungszeuge von j nach Phase 1 war, muss zudem gelten: $\alpha_j \geq t_{i'}$.

Würde diese Ungleichung nicht gelten, so müsste eine Stadt j'' existieren, für die gilt:
$t_{i'} = \alpha_{j''} > \alpha_j$.
Da aber alle α_j immer mit derselben Rate erhöht werden, könnte diese Ungleichung nur gelten, wenn j vorher schon mit einer Stadt verbunden worden wäre. Dann wäre aber i' nicht der Verbindungszeuge von j nach Phase 1 gewesen.

Folglich gilt: $\alpha_j \geq t_{i'} \geq \ min(t_i, \ t_{i'}) \geq \alpha_{j'}$. $(**)$

Da es sich bei dem Problem um ein metrisches Faciliy Location Problem handelt, folgt:

$$c_{ij} \overset{metrisch}{\leq} c_{i'j} + c_{i'j'} + c_{ij'} \overset{(*)}{\leq} c_{i'j} \ + \ 2\alpha_{j'} \overset{(**)}{\leq} \alpha_j \ + \ 2\alpha_j = 3\alpha_j.$$

Also gilt:

$$\frac{c_{ij}}{3} \leq \alpha_j \leq c_{ij}$$

bzw.

$$\frac{c_{ij}}{3} \leq \alpha_j - \beta_{ij} \leq c_{ij}.$$

□

Korollar 5.5. *Für alle $(i, j) \in E$ gilt:*

$$x_{ij} > 0 \Rightarrow \frac{c_{ij}}{3} \leq \alpha_j - \beta_{ij} \leq c_{ij}$$

Beweis: Die Behauptung folgt unmittelbar aus Lemma 5.3. und Lemma 5.4. □

Lemma 5.6. *Sei $i \in I$. Dann gilt:*

$$\sum_{j \in C} \beta_{ij} = f_i.$$

Beweis: Nach Konstruktion von I bzw. Erläuterung 4.6. gilt:

$$i \in I \Rightarrow i \text{ wurde in Phase 1 temporär eröffnet.}$$

Eine Facility $i \in F$ wurde in Phase 1 temporär eröffnet, als galt:

$$\sum_{j \in C} \beta_{i'j} = f_{i'}.$$

Da nach der temporären Eröffnung der Facility i die Werte aller β_{ij} mit $j \in C$ nicht mehr verändert wurden, folgt daraus nun die Behauptung. □

Satz 5.7. *Die Lösung des Algorithmus von Jain und Vazirani genügt den primalen Schlupfbedingungen mit Relaxierungsfaktor 3.*

Beweis: Klarerweise gilt auf Grund von Lemma 5.6:

$$\frac{f_i}{3} \leq \sum_{j:\varphi(j)=i} \beta_{ij} \leq f_i \quad \forall i \in I.$$

Die Behauptung folgt somit aus dieser Ungleichung und Korollar 5.5. □

Lemma 5.8. *Für die Lösung des Algorithmus von Jain und Vazirani gilt für alle $j \in C$:*

$$\alpha_j > 0 \;\Rightarrow\; \sum_{i \in F} x_{ij} = 1. \quad (3.9)$$

Beweis: Diese Behauptung wurde bereits im Beweis von Satz 5.1 gezeigt. □

Lemma 5.9. *Für die Lösung des Algorithmus von Jain und Vazirani gilt:*

$$\beta_{ij} > 0 \;\Rightarrow\; y_i - x_{ij} = 0 \quad (3.10)$$

Beweis. Ist $\beta_{ij} >0$, so handelt es sich bei (i, j) um eine spezielle Kante. Es können dabei die folgenden Fälle auftreten:

a.) y_i=1 bzw. $i \in I$.
Da (i, j) eine spezielle Kante ist, gilt: $i \in G_j$. Folglich muss in Phase 2 des Algorithmus Fall 1 eingetreten sein, was x_{ij}=1 impliziert. Daraus folgt: $y_i - x_{ij} = 0$.

b.) y_i=0.
Klarerweise gilt dann hier: x_{ij}=0.

□

Satz 5.10. *Die Lösung des Algorithmus von Jain und Vazirani genügt den dualen Schlupfbedingungen, d.h. sie genügt den relaxierten dualen Schlupfbedingungen mit dem Relaxierungsfaktor 1.*

Beweis: Folgt unmittelbar aus Lemma 5.8 und Lemma 5.9. □

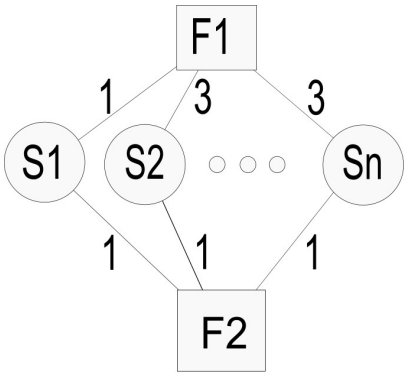
Theorem 5.11. *Bei dem Algorithmus von Jain und Vazirani handelt es sich, sofern er polyonimell ist, um einen 3-Approximations-Algorithmus für das metrische unkapazitierte Facility Location Problem.*

Beweis: Bei dem metrischen unkapazitierten Facility Location Problem bzw. bei Modell (3.1) handelt es um ein Problem der Form (2.5P).
Wählen wir nun als Relaxierungsfaktor für die primale Schlupfbedingungen $\alpha = 3$ und als Relaxierungsfaktor für die dualen Schlupfbedingungen $\beta = 1$, so folgt die Behauptung mit Hilfe der Sätze 5.1, 5.2, 5.7. und 5.10. aus Theorem 2.5. □

Die nachfolgende Familie von Beispielen (übernommen aus [3]) zeigt, dass die Approximationsgüte 3 scharf ist.

Beispiel 5.12. *In diesem Beispiel müssen n Städte versorgt werden. Hierfür stehen zwei Facilities F1 und F2 zur Verfügung stehen. Die Eröffnungkosten für Facility F1 bzw. F2 betragen ε bzw. $(n+1)\varepsilon$, wobei ε eine kleine Zahl ist. Die Verbindungskosten sind:*

$$\begin{aligned} c_{11} &= 1 \\ c_{2j} &= 1 && , \ \forall j \in C \\ c_{1j} &= 3 && , \ \forall j \in C \backslash \{1\} \end{aligned}$$



In Phase 1 des Algorithmus von Jain und Vazirani werden beide Facilities temporär eröffnet. In Phase 2 kommt es jedoch nur zu einer Eröffnung der Facility F1, da sowohl die Facility F1 als auch die Facility F2 eine spezielle Kante mit der Stadt 1 besitzen und sie somit eine gemeinsame Kante in H haben. Die Lösung des Algorithmus von Jain und Vazirani für dieses Beispiel besagt demnach, dass alle Städte mit der Facility F1 verbunden werden soll. Die Gesamtkosten der Lösung lauten damit: $1 + 3(n-1) + \varepsilon$.
Die Optimallösung für dieses Problem wäre aber alle Städte mit der Facility F2 zu verbinden, was zu Kosten von $\varepsilon(n+1) + n$ *führen würde.*

5.2. Laufzeitanalyse

Im Folgenden wird gezeigt, dass der Algorithmus polynomielle Laufzeit hat und damit tatsächlich ein 3-Approximations-Algorithmus ist.

Dazu gelte: $n_c = |C|$, $n_f = |F|$ und $m = n_c n_f$.
m ist somit die Anzahl der Kanten.

Zu Beginn des Algorithmus müssen die Kanten aufsteigend nach ihren Verbindungskosten sortiert werden, um zu ersehen, wann jede einzelne Kante in Phase als bezahlt markiert wird bzw. um die Erhöhungsrate der α_j bestimmen zu können. Die Sortierung der m Werte benötigt bekanntlich eine Laufzeit von $O(m\ log\ m)$ (siehe [2]). Des Weiteren müssen die Öffnungszeiten der Facilities gespeichert werden, da diese die Erhöhungsrate ebenfalls mitbestimmen. Unter der Öffnungszeit einer Facility verstehen wir hierbei den Wert, den die zu erhöhenden α_j annehmen müssen, damit diese Facility temporär eröffnet wird. Die Öffnungszeiten der Facilities können in einem binären Heap gespeichert werden, bei dem jeder einzelne Wert in $O(log\ n_f)$ aktualisiert und in der selben Zeit jeweils auch das Minimum bestimmt werden kann (siehe [2]). Am Anfang setzen wir für jede Facility die Eröffnungszeit als *unendlich*, da sich noch keine Stadt an den Eröffnungskosten beteiligt.

Während der Phase 1 treten die folgenden verschiedenen Ereignisse ein:

- Die Kante (i, j) wird als bezahlt markiert

  Fall 1: Facility i ist noch nicht temporär eröffnet worden

  In diesem Fall muss die Eröffnungszeit der Facility i neu berechnet werden, da sich die Stadt j ab sofort ebenfalls an den Eröffnungskosten beteiligt. Die Berechnung der neuen Eröffnungszeit findet in konstanter Zeit statt. Folglich löst dieses Ereignis auf Grund der Aktualisierung eines Eintrags im Heap eine Arbeit von $O(log\ n_f)$ aus.

  Fall 2: Facility i wurde bereits temporär eröffnet

  In diesem Fall müssen die Eröffnungskosten aller Facilities, zu denen j eine bezahlte Kante hat, neu berechnet und aktualisiert werden, da der Beitrag von j nicht mehr mitwächst. Folglich müsste pro Facility, an der j beteiligt ist, eine Arbeit von $O(log\ n_f)$ getätigt werden. Insgesamt löst dieses Ereignis also eine Arbeit von $O(n_f\ log(n_f))$ aus.

- Die Facility i wird temporär eröffnet

  In diesem Fall wird jede Stadt j, die eine bezahlte Kante zu i hat, als verbunden markiert. Die dazugehörigen α_j und $\beta_{i'j}$ mit $i' \in F$ werden nicht mehr miterhöht. Folglich müssen die Eröffnungszeiten jeder Facility, an der die neu-verbundene Stadt beteiligt ist, neuberechnet und aktualisiert werden. Dies führt dazu, dass pro neu-verbundener Stadt eine Arbeit von $O(n_f\ log(n_f)\ )$ getätigt werden muss.

Bemerkung 5.13. *Jede Kante (i, j) löst höchstens zwei Ereignisse aus, da jede Kante (i, j) nur einmal als bezahlt markiert wird und jede Stadt nur einmal verbunden wird.*

Theorem 5.14. *Der Approximationsalgorithmus von Jain und Vazirnai hat eine Laufzeit von $O(m \log m)$ und ist somit ein 3-Approximations-Algorithmus für das metrische unkapazitierte Facility Location Problem.*

Beweis: Insgesamt wird die Laufzeit des Algorithmus von Phase 1 dominiert. Deswegen beschränken wir uns im Folgenden auf die Laufzeitbestimmung von Phase 1.

Für jede Stadt j kann es insgesamt n_f-mal dazu kommen, dass eine von j ausgehende Kante als bezahlt markiert wird. Dies führt jedes Mal zu einer Arbeit von $O(log\ n_f)$. Werden alle Städte als bezahlt markiert, ergibt sich somit eine Arbeit von $O(n_c\ n_f\ log(n_f))$.
Zusätzlich erhält jede Stadt in Phase 1 genau einen Verbindungszeugen. Folglich tritt pro Stadt entweder genau einmal Fall 2 ein oder es wird genau einmal eine Facility temporär eröffnet, zu der die Stadt eine bezahlte Kante besitzt. Beide Ereignisse können nicht zusammen bei ein und derselben Stadt auftreten. Da jedes der Ereignisse zu einer Arbeit von $O(n_f\ log(n_f))$ führt, kommt es bei der Verbindung aller Städte in Phase 1 zu einer Arbeit von $O(n_c\ n_f\ log(n_f)\ )= O(m\ log(n_f)\ )$.

Daraus folgt, dass ohne die Sortierung der Verbindungskosten eine Arbeit von $O(m\ log(n_f)\ )$ in Phase 1 getätigt werden muss. Damit hat also der Algorithmus eine Laufzeit von $O(m\ log\ m)$ und ist somit polynomiell.

Mit Hilfe von Theorem 5.11. ergibt sich somit, dass es sich bei dem Approximationsalgorithmus von Jain und Vazirani um eine 3-Approximations-Algorithmus handelt. □

5.3. Sensititvitätsanalyse

Bisher wurde davon ausgegangen, dass die Daten fest vorgegeben sind. Dies ist jedoch nicht unbedingt immer der Fall. Daten können sich ändern, bzw. können falsch angegeben worden sein. Deswegen wird nun untersucht, wie sich die Änderungen auf die Lösung auswirken. Hierbei wird jeweils überprüft, ob die Lösung weiterhin primal und dual zulässig ist und ob sie weiterhin den Bedingungen (3.7) bzw. (4.1), (3.8), (3.9) und (3.10) genügt.

Veränderung der möglichen Standorte

Hier unterscheiden wir zwischen dem Fall, dass an einem zunächst in Erwägung gezogenen Standort doch keine Facility gebaut werden kann, und dem Fall, dass an einem weiteren Standort eine Facility gebaut werden kann.

Fall 1: Facility i kann doch nicht eröffnet werden

In diesem Fall haben wir weiterhin eine dual zulässige Lösung, da nur einige Nebenbedingungen beim dualen Problem wegfallen. Außerdem werden klarerweise die Bedingungen (3.7) bzw. (4.1), (3.8) und (3.10) weiterhin von der bisherigen Lösung des Algorithmus erfüllt.
Folglich ist nur noch zu überprüfen, ob die Lösung weiterhin primal zulässig ist und ob Bedingung (3.9) weiterhin erfüllt ist.

- Wurde die Facility i nicht eröffnet, so galt für alle $j \in C$: $x_{ij} = 0$.
  Folglich gilt:
  $$\sum_{i' \in F \setminus \{i\}} x_{i'j} = 1, \ \forall j \in C.$$
  Somit sind Bedingung (3.9) und die erste Nebenbedingung des primalen Problems ebenfalls weiterhin erfüllt. Da die zweite Nebenbedingung des primalen Problems weiterhin für alle $i' \in F \setminus \{i\}$ und für alle $j \in C$ erfüllt ist, folgt daraus, dass die bisherige Lösung weiterhin maximal 3 mal schlechter als die Optimallösung des neuen Problems ist.

- Wurde die Facility i jedoch eröffnet, so ist unsere bisherige Lösung nicht mehr primal zulässig und genügt auch nicht mehr Bedingung 3.9. In diesem Fall gab es nämlich mindestens eine Stadt $j \in C$, die mit der Facility i verbunden war und für die somit galt: $x_{ij} = 1$. Dementsprechend gilt nun:
  $$\sum_{i \in F \setminus \{i\}} x_{ij} = 0.$$
  Des Wegen müsste in diesem Fall der Algorithmus von neuem gestartet werden, um weiterhin garantieren zu können, dass die Lösung zulässig und maximal 3-mal schlechter als die Optimallösung ist. Hierbei können die Berechnung aus Phase 1 bis zu dem Zeitpunkt, wo Facility i temporär eröffnet wurde, übernommen werden.

Fall 2: Es kann an einem weiteren Standort eine Facility gebaut werden, d. h. es existiert nun eine zusätzlich Facility $i' \in F$.

Gelte nun für alle $j \in C$ und für die zusätzliche Facility i':

$$\begin{aligned} \beta_{i'j} &:= max(\alpha_j - c_{i'j}, 0) \\ x_{i'j} &:= 0 \\ y_{i'} &:= 0, \end{aligned}$$

so ist unsere bisherige Lösung weiterhin primal zulässig und genügt den relaxierten Schlupfbedingungen.

- Gilt zusätzlich:
$$\sum_{j \in C} \beta_{i'j} \leq f_{i'},$$
so ist die bisherige Lösung weiterhin auch dual zulässig. Diese Ungleichung impliziert nämlich, dass die zweite Nebenbedingung des dualen Problem erfüllt ist. Die erste Nebenbedingung wird auf Grund der Definition von $\beta_{i'j}$ klarerweise erfüllt. Folglich muss keine neue Lösung berechnet werden.

- Existiert nun aber eine Menge $C' \subset C$, für die gilt:
$$\sum_{j \in C'} \beta_{i'j} > f_{i'},$$
so ist die vorher berechnete Lösung nicht mehr dual zulässig, da die zweite Nebenbedingung des dualen Problems verletzt wird. Deshalb muss der Algorithmus in diesem Fall von neuem gestartet werden.

Veränderung der Menge der zu versorgenden Städte

Fall 1: Es muss eine Stadt $j \in C$ weniger bedient werden.

In diesem Fall gilt weiterhin:

$$\begin{aligned}
&\sum_{i \in F} x_{ij'} = 1 && ,\forall j' \in C \setminus \{j\} \\
&y_i - x_{ij'} \geq 0 && ,\forall i \in F,\ j' \in C \setminus \{j\} \\
&\alpha_{j'} - \beta_{ij'} \leq c_{ij'} && ,\forall i \in F,\ j' \in C \setminus \{j\} \\
&\sum_{j' \in C \setminus \{j\}} \beta_{ij'} \leq f_i && ,\forall i \in F.
\end{aligned}$$

Folglich ist unsere bisherige Lösung weiterhin primal und dual zulässig. Außerdem sind weiterhin die Bedingungen (3.9), (3.10) und (3.7) bzw. (4.1) erfüllt.

- Gilt: $\beta_{ij} = 0, \forall i \in I$,
d. h. trat in Phase 2 des Algorithmus bei der Stadt $j \in C$ der Fall 2 ein, so gilt weiterhin:
$$y_i > 0 \Rightarrow \sum_{j \in C} \beta_{ij} = f_i. \quad (3.8)$$
Daraus folgt, dass die bisherige Lösung weiterhin maximal 3-mal schlechter als die Optimallösung ist.

- Trat in Phase 2 des Algorithmus jedoch Fall 1 für die Stadt j ein, d.h. existiert ein $i \in I$, für das gilt:

  $$\beta_{ij} > 0,$$

  so ist Bedingung (3.8) nicht mehr erfüllt. Es gilt dann nämlich für ein $i \in I$:

  $$\sum_{j \in C \setminus \{j\}} \beta_{ij} < f_i.$$

  Folglich muss in diesem Fall der Algorithmus von neuem gestartet werden, wenn die Lösung alle genannten Bedingungen erfüllen sollen.
  Möchte man nur garantieren, dass die Lösung maximal 3-mal schlechter als die Optimallösung des veränderten Problems ist, so muss in diesem Fall der Algorithmus nicht von neuem gestartet werden, wenn gilt:

  $$\frac{f_i}{3} \leq \sum_{j \in C \setminus \{j\}} \beta_{ij} < f_i$$

  Gilt diese Ungleichung nämlich, so genügt die Lösung weiterhin den relaxierten primalen Schlupfbedingungen mit Relaxierungsfaktor 3.

Fall 2: Es muss eine weitere Stadt $j' \in C$ bedient werden.

Sei nun $A_j := \{i \in F : c_{ij} \leq c_{i'j}$, für alle $i \in F\}$. Dann wähle ein Element r aus A_i.

- Gilt nun $r \in I$ und $c_{rj'} = t_r$, dann setzen wir $\alpha_{j'} = c_{rj'}$ und deklarieren j' als mit r *direkt verbunden.* Die auf diesem Weg gewonnene Lösung ist sowohl primal als auch dual zulässig und genügt den relaxierten Schlupfbedingungen.
  Da jede Stadt $j \in C$ genau mit einer geöffneten Facility verbunden wurde, ist die so gewonnene Lösung auch primal zulässig. Auf Grund der Wahl von r und der Gleichung $c_{rj'} = t_r = \alpha_{j'}$ gilt:

  $$\begin{aligned}
  &\beta_{ij'} = 0 && ,\forall i \in F \\
  &\alpha_{j'} - \beta_{ij'} \leq c_{ij'} && ,\forall i \in F \\
  &\alpha_{j'} - \beta_{rj'} = c_{rj'} && \\
  &\sum_{j \in C \cup \{j'\}} \beta_{ij} \leq f_i && ,\forall i \in F \\
  &\sum_{j \in C \cup \{j'\}} \beta_{ij} = f_r &&
  \end{aligned}$$

  Deshalb ist die Lösung auch dual zulässig und genügt den relaxierten Schlupfbedingungen.

- Gilt $r \notin I$, aber r wurde in Phase 1 temporär eröffnet und $t_r \leq c_{rj'}$, so können wir die Ergebnisse aus Phase 1 übernehmen. Im Falle, dass wir noch zusätzlich den Verbindungszeugen von j' auf r und $\alpha_{j'} = c_{rj'}$ setzen, müssen wir nur Phase 2 neu starten.

- In allen anderen Fällen müssen wir jedoch den Algorithmus komplett von neuem starten, wobei die Berechnungen bis zum Zeitpunkt $c_{rj'}$ übernommen werden können.

Veränderung der Verbindungskosten

Fall 1: $c_{ij_{neu}} > c_{ij}$

Klarerweise ist in diesem Fall die bisherige Lösung weiterhin primal und dual zulässig und genügt auch den Bedingungen (3.8), (3.9) und (3.10). Bedingung (3.7) bzw. (4.1) werden jedoch nicht unbedingt erfüllt.

- Wurde die Stadt j nicht mit Facility i verbunden, so ist die Bedingung (3.7) bzw. (4.1) weiterhin erfüllt, da dann gilt: $x_{ij} = 0$.

- Wurde die Stadt j aber mit Facility i verbunden, so ist Bedingung (3.7), bzw. (4.1) nicht erfüllt, es sei denn, j ist indirekt verbunden und es gilt:
  $\frac{c_{ij_{neu}}}{3} \leq \alpha_j - \beta_{ij} \leq c_{ij_{neu}}$.
  Möchte man nur, dass die Lösung zulässig und maximal 3-mal schlechter als die Optimallösung ist, so genügt, es wenn i direkt verbunden war und $\frac{c_{ij_{neu}}}{3} \leq \alpha_j - \beta_{ij} \leq c_{ij_{neu}}$ gilt, i als indirekt verbunden zu markieren. In diesem Fall genügt nämlich die Lösung immer noch den relaxierten primalen Schlupfbedingungen mit Relaxierungsfaktor 3.
- In allen anderen Fällen muss der Algorithmus von neuem gestartet werden.

Fall 2: $c_{ij_{neu}} < c_{ij}$

In diesem Fall ist die bisherige Lösung nicht mehr unbedingt dual zulässig, da gelten könnte: $c_{ij_{neu}} < \alpha_j - \beta_{ij}$.
Sie ist jedoch weiterhin primal zulässig und genügt den Bedingungen (3.8), (3,9) und (3.10).
Im Folgenden nehmen wir an, dass unsere bisherige Lösung nicht mehr dual zulässig ist:

- Ist j mit i direkt verbunden und gilt $\alpha_j > c_{ij_{neu}} - c_{ij}$, so können wir $\alpha_j = c_{ij_{neu}} + \beta_{ij}$ setzen. Setzen wir α_j so, dann ist die neue Lösung primal zulässig und genügt den Bedingungen (3.8), (3.9) und (3.10). Außerdem ist sie dual zulässig und genügt Bedingung (3.7) bzw. (4.1), da gilt: $\alpha_j - \beta_{ij} = c_{ij_{neu}}$.

- Ist j mit i indirekt verbunden, können wir nicht unbedingt $\alpha_j = c_{ij_{neu}} + \beta_{ij}$ setzen, da dann unter Umständen Bedingung (4.1) nicht mehr erfüllt ist. Es könnte nämlich gelten: $\frac{c_{ij_{neu}}}{3} > \alpha_j - \beta_{ij}$.

- Ist $i \notin I$ und gilt: $\sum_{j' \in C \setminus \{j\}} \beta_{ij'} + \alpha_j - c_{ij_{neu}} \leq f_i$, so genügt es $\beta_{ij_{neu}} = \alpha_j - c_{ij_{neu}}$ zu setzen, da dann beide Nebenbedingungen des dualen Problems wieder erfüllt werden und die veränderte Lösung den relaxierten Schlupfbedingungen genügt. Die relaxierten Schlupfbedingungen werden erfüllt, da gilt in diesem Fall gilt: $y_i = 0$ und $x_{ij} = 0$.

- Tritt keiner der genannten Fälle ein, so müssen wir den Algorithmus neu ausführen.

Veränderung der Eröffnungskosten

Fall 1: $f_{i_{neu}} > \tilde{f}_i$.

In diesem Fall gilt: $\sum_{j \in C} \beta_{ij} \leq \tilde{f}_i \leq f_{i_{neu}}$.
Folglich haben wir weiterhin eine primal und dual zulässige Lösung, welche allen relaxierten Schlupfbedingungen bis auf Bedingung (3.8) genügt.

- Wurde die Facility i nicht eröffnet, dann gilt: $y_i = 0$.
  In diesem Fall erfüllt die bisherige Lösung somit immer noch Bedingung (3.8); deshalb muss keine neue Lösung berechnet werden.

- Wurde die Facility i hingegen eröffnet, so ist Bedingung (3.8) nicht mehr erfüllt. Der Algorithmus muss neu gestartet werden. Hierbei können Berechnungen bis zu dem Zeitpunkt, zu dem i temporär eröffnet wurde, wieder übernehmen werden.

Fall 2: $f_{i_{neu}} < \tilde{f}_i$

In diesem Fall ist die bisherige Lösung eventuell nicht mehr dual zulässig und genügt eventuell auch nicht mehr Bedingung (3.8).

- Wurde die Facility i nicht eröffnet, so erfüllt die Lösung auch weiterhin Bedingung (3.8), da $y_i = 0$ gilt. Gilt zusätzlich:

  $$\sum_{j \in C} \beta_{ij} \leq f_{i_{neu}},$$

  so ist die Lösung auch weiterhin dual zulässig. Es ist keine Neuberechnung notwendig.
  Ist diese Ungleichung jedoch nicht erfüllt, muss der Algorithmus von neuem gestartet werden.

- Wurde die Facility i hingegen eröffnet, so ist die bisherige Lösung nicht mehr dual zulässig und genügt nicht mehr Bedingung (3.8), da nun gilt:

$$\sum_{j \in C} \beta_{ij} > f_{i_{neu}}.$$

Jedoch können wir mit folgender Hilfsprozedur die bisherige Lösung so verändern, dass sie wieder primal und dual zulässig wird und dass sie wieder den relaxierten Schlupfbedingungen genügt.

Am Anfang einer jeden Iteration sucht die Hilfsprozedur die Stadt $u \in C$, für die gilt:

- u hat mit i eine spezielle Kante.
- $\beta_{iu} \leq \beta_{ij}$, für alle j, die mit i eine spezielle Kante besitzen.

Anschließend wird $d_{iu} = min(\beta_{iu}, f_i - f_{i_{neu}})$ gesetzt und α_u und β_{iu} werden jeweils um d_{iu} reduziert.
Gilt $d_{iu} = \beta_{iu}$, so wird bei Kante (i, j) die Markierung als *spezielle* Kante gelöscht. Sobald nach einer Iteration gilt:

$$\sum_{j \in C} \beta_{ij} = f_{i_{neu}},$$

terminiert die Hilfsprozedur.

Da innerhalb der Hilfsprozedur nur die Dualvariablen verändert werden, ist die neue Lösung weiterhin primal zulässig und genügt den Bedingungen (3.9) und (3.10). Des Weiteren erfüllt die neue Lösung klarerweise auch die zweite Nebenbedingung des dualen Problems und die Bedingung (3.8). Auf Grund der Tatsache, dass innerhalb der Hilfsprozedur α_u und β_{iu} jeweils um denselben Wert gesenkt werden und u durch unsere Wahl immer eine mit i direkt verbundene Stadt ist, erfüllt die neue Lösung folgende Ungleichungen:

$$\begin{aligned}
&\alpha_{ij} - \beta_{ij} \leq c_{ij} && ,,\forall j \in C \; i \in F \\
&\alpha_{\varphi(j)j} - \beta_{\varphi(j)j} \leq c_{\varphi(j)j} && ,\; \forall j \in D \\
&\frac{c_{\varphi(j)j}}{3} \leq \alpha_{\varphi(j)j} - \beta_{\varphi(j)j} \leq c_{\varphi(j)j} && ,\; \forall j \in D^c
\end{aligned}$$

Hierbei bezeichnet D wieder die Menge der *direkt verbundenen* Städte und D^c die Menge der *indirekt verbundenen* Städte.
Aus den Ungleichungen folgt nun, dass die neue Lösung ebenfalls dual zulässig ist und die Bedingung (3.7) bzw. (4.1) erfüllt. Der Algorithmus von Jain und Vazirani muss somit nicht von neuem gestartet werden.

6. Varianten des Facility Location Problems

6.1. Quadratische Verbindungskosten

In diesem Abschnitt werden Varianten des metrischen unkapazitierten Facility Location Problems vorgestellt, zu deren Lösung sich ebenfalls der Approximationsalgorithmus von Jain und Vazirani eignet.
Bei dieser Variante sind die Verbindungskosten der Stadt $j \in C$ mit der Facility $i \in F$ gegeben durch c_{ij}^2. Hierbei genügen die c_{ij} weiter der Dreiecksungleichung, für die c_{ij}^2s muss dies jedoch nicht gelten.
Mit der Dreiecksungleichung ist in diesem Zusammenhang gemeint, dass für alle $(i,j) \in E$ gilt:

$$c_{ij} \leq c_{i'j} + c_{i'j'} + c_{ij'},\ \forall j' \in C, \forall i' \in F.$$

Satz 6.1. *Der Approximationsalgorithmus von Jain und Vazirani ist ein 9-Approximations-Algorithmus für diese Varainte.*

Beweis: Der Beweis läuft fast analog zur Bestimmung der Approximationsgüte des Algorithmus (siehe Abschnitt 5.1). Lediglich der Beweis von Lemma 5.4. und daraus resultierend das Korollar 5.5, der Satz 5.7 und das Theorem 5.11 müssen angepasst werden. Das Lemma 5.4. muss insofern abgewandelt werden, dass für jede indirekte Verbindung (i,j) nur noch gilt:

$$\frac{c_{ij}^2}{9} \leq \alpha_j^2 \leq c_{ij}^2.$$

Der Beweis dieser Ungleichung läuft fast analog zum Beweis von Lemma 5.4. Nur der letzte Teil des Beweises, in dem ausgenutzt wurde, dass es sich um ein metrisches Problem handelt, muss verändert werden. Dort müsste nun stehen:

$$(c_{ij})^2 \overset{c_{ij}\ metrisch}{\leq} (c_{i'j} + c_{i'j'} + c_{ij'})^2 \overset{(*)}{\leq} (c_{i'j} + 2\alpha_{j'})^2 \overset{(**)}{\leq} (\alpha_j + 2\alpha_j)^2 = (3\alpha_j)^2 = 9\alpha_j^2.$$

Also gilt:

$$\frac{c_{ij}^2}{9} \leq \alpha_j^2 \leq c_{ij}^2.$$

Die Veränderung von Lemma 5.4 bewirkt, dass die Lösung des Approximationsalgorithmus von Jain und Vazirani für dieses Problem nicht mehr garantiert den relaxierten primalen Schlupfbedingung mit Relaxierungsfaktor 3 genügt, sondern nur noch den relaxierten primalen Schlupfbedingungen mit Relaxierungsfaktor 9.
Folglich hat der Approximationsalgorithmus von Jain und Vazirani nach Theorem 2.5. für dieses Problem die Approximationsgüte 9.

□

6.2. Arbitrary-demands-Variante

Bei dieser Variante des unkapazitierten metrischen Facility Location Problems ist für jede Stadt $j \in C$ eine nichtnegative Nachfrage d_j gegeben, wobei jede Facility $i \in F$ diese Nachfrage befriedigen kann. Die Verbindungskosten von Stadt j mit Facility i betragen nun $c_{ij}d_j$ anstelle von c_{ij}. Es gilt aber weiterhin für alle $(i,j) \in E$:

$$c_{ij} \leq c_{i'j} + c_{i'j'} + c_{ij'},\ \forall j' \in C, \forall i' \in F.$$

Als IP geschrieben lautet das Problem somit:

$$
\begin{aligned}
&min && \sum_{i\in F, j\in C} c_{ij}d_j x_{ij} + \sum_{i\in F} f_i y_i && && \\
&s.t. && \sum_{i\in F} x_{ij} \geq 1 && , \ j \in C && (6.1)\\
& && y_i - x_{ij} \geq 0 && ,\forall i \in F, \ j \in C && \\
& && x_{ij} \in \{0,1\} && ,\forall i \in F, \ j \in C && \\
& && y_i \ \in \{0,1\} && ,\forall i \in F. &&
\end{aligned}
$$

Als LP-Relaxation ergibt sich damit analog zu Kapitel 3:

$$
\begin{aligned}
&min && \sum_{i\in F, j\in C} c_{ij}d_j x_{ij} + \sum_{i\in F} f_i y_i && && \\
&s.t. && \sum_{i\in F} x_{ij} \geq 1 && , \ j \in C && (6.2)\\
& && y_i - x_{ij} \geq 0 && ,\forall i \in F, \ j \in C && \\
& && x_{ij} \geq 0 && ,\forall i \in F, \ j \in C && \\
& && y_i \ \geq 0 && ,\forall i \in F. &&
\end{aligned}
$$

Das duale Problem zu (6.2) ist:

$$
\begin{aligned}
&max && \sum_{j\in C} \alpha_j && && \\
&s.t. && \alpha_j - \beta_{ij} \leq c_{ij}d_j && ,\forall i \in F, \ j \in C && (6.3)\\
& && \sum_{j\in C} \beta_{ij} \leq f_i && ,\forall i \in F && \\
& && \alpha_j \geq 0 && ,\forall j \in C && \\
& && \beta_{ij} \geq 0 && ,\forall i \in F, j \in C. &&
\end{aligned}
$$

Zur Lösung des Problems (6.1) modifizieren wir den Algorithmus von Jain und Vazirani (siehe [3]) wie folgt:
In Phase 1 werden nicht mehr alle zu erhöhenden α_j und β_{ij} um den selben Wert gleichzeitig erhöht, sondern sie werden jeweils um d_j-Einheiten der Erhöhungsrate e' gleichzeitig erhöht. Die Erhöhungsrate e' wird hierbei wie in Anhang B bestimmt. Eine Kante (i, j) wird in der modifizierten Phase 1 als bezahlt markiert, wenn $\alpha_j = c_{ij}d_j$ gilt.
Ansonsten kommt es zu keiner Änderung von Phase 1. Die Phase 2 des Algorithmus wird überhaupt nicht verändert.

Satz 6.2. *Durch die Modifikation des Approximationslgorithmus von Jain und Vazirani ergibt sich ein 3-Approximations-Algorithmus für die Arbitrary-demands-Variante des metrischen unkapazitierten Facility Location Problems, sofern für die Lösung gilt:*

$$x_{ij} > 0 \Rightarrow \frac{c_{ij}d_j}{3} \leq \alpha_j - \beta_{ij} \leq d_j c_{ij}.$$

Beweis: Die modifizierte Phase 1 liefert uns auf Grund der geschickten Wahl von e' (siehe Anhang B) zu jedem Zeitpunkt eine zulässige Lösung für (6.3). Durch die Wahl von e' wird nämlich sichergestellt, dass weder das Ereignis der Markierung einer Kante (i, j) als bezahlt noch das Ereignis der temporären Eröffnung einer Facility i übergangen werden kann. Werden diese Ereignisse nicht übergangen, so lösen sie Handlungen aus, die sicherstellen, dass (α, β) weiterhin zulässig bleiben (siehe Erläuterungen in Abschnitt 4.1).
Die Phase 2, angewendet auf das Ergebnis der modifizierten Phase 1, liefert nun zusätzlich nach Satz 5.1. eine zulässige Lösung für (6.1). In Phase 2 wird die duale Lösung nicht verändert, weshalb sie weiterhin zulässig ist.
Außerdem genügen die primale und die duale Lösung des modifizierten Algorithmus den dualen Schlupfbedingungen, da die Beweise von Lemma 5.8. und Lemma 5.9. weiterhin gelten. Würden die primale und die duale Lösung nun ebenfalls den relaxierten primalen Schlupfbedingungen mit Relaxierungsfaktor 3 genügen, so würde die Behauptung aus Theorem 2.5. folgen, da der modifizierte Algorithmus immer noch polynomiell ist.

Auf Grund dessen, dass der Beweis von Lemma 5.6. weiterhin gültig ist, gilt:

$$y_i > 0 \Rightarrow \sum_{j \in C} \beta_{ij}.$$

Mit Hilfe der Voraussetzung folgt nun, dass die Lösung des modifizierten Algorithmus den relaxierten primalen Schlupfbedingungen mit Relaxierungsfaktor 3 genügt, weshalb aus Theorem 2.5. die Behauptung folgt. □

Lemma 6.3. *Bei der Lösung des modifizierten Algorithmus gilt für alle $(i, j) \in E$:*

$$x_{ij} > 0 \Rightarrow \frac{c_{ij}d_j}{3} \leq \alpha_j - \beta_{ij} \leq d_j c_{ij}.$$

Beweis: Handelt es sich bei der Kante $(i'', j'') \in E$ um eine direkte Verbindung, so kann analog zum Beweis von Lemma 5.3 gezeigt werden, dass sogar

$$\alpha_{j''} - \beta_{i''j''} = c_{i''j''}d_{j''}$$

gilt. Folglich bleibt nur noch zu zeigen, dass für jede indirekte Verbindung $(i, j) \in E$ gilt:

$$\frac{c_{ij}d_j}{3} \leq \alpha_j - \beta_{ij} \leq d_j c_{ij}.$$

Dass diese Ungleichung für jede indirekte Verbindung $(i, j) \in E$ gilt, kann man nun aber fast analog zu dem Beweis von Lemma 5.4 zeigen. Auf Grund der Tatsache, dass (i, j) eine

indirekte Verbindung ist, muss nämlich wieder eine Facility $i' \in H$ existieren, die nach Phase 1 der Verbindungszeuge von j war und die in H mit i eine gemeinsame Kante besitzt. Daraus folgt mit demselben Argument wie in Lemma 5.4, dass eine Stadt $j' \in C$ existieren muss, für die gilt:

$$\alpha_{j'} > c_{ij'} d_{j'} \text{ und } \alpha_{j'} > c_{i'j'} d_{j'}. \quad (*)$$

Interpretiert man nun eine Erhöhung von $\alpha_{j'}$ um $d_{j'}$ als einen Fortschritt der Zeit um eine Einheit, so gibt $\tilde{\alpha_{j'}} = \frac{\alpha_{j'}}{d_{j'}}$ die Zeit an, zu der die Stadt j' in Phase 1 verbunden wurde.
Ist nun t_i die Zeit, zu der die Facility i temporär eröffnet wurde, so muss gelten:

$$\tilde{\alpha_{j'}} \leq t_i, \text{ für alle } j' \in C, \text{ die mit } i \text{ eine spezielle Kante haben.}$$

Würde diese Ungleichung nicht gelten, so wäre j' weiter erhöht worden, obwohl j' bereits mit einer temporär eröffneten Facility eine spezielle Kante hat. Dies kann auf Grund des Ablaufs der Phase 1 des modifizierten Algorithmus aber nicht sein.
Da (i, j') und (i', j') spezielle Kanten muss demnach gelten:

$$\frac{\alpha_{j'}}{d_j} \leq min(t_i, t_{i'}).$$

Des Weiteren gilt, da i' der Verbindungszeuge von j ist und somit α_j erst aufhörte zu wachsen als j mit i' verbunden wurde::

$$\frac{\alpha_j}{d_j} \geq t_{i'}.$$

Daraus folgt:

$$\frac{\alpha_j}{d_j} \geq t_{i'} \geq min(t_i, t_{i'}) \geq \frac{\alpha_{j'}}{d_j}. \quad (**)$$

Mit Hilfe von (*) und (**) ergibt sich:

$$c_{ij} d_j \overset{c_{ij}s \ metrisch}{\leq} (c_{i'j} + c_{i'j'} + c_{ij'}) d_j \overset{(*)}{\leq} c_{i'j} d_j \ + \ 2\frac{\alpha_{j'}}{d_{j'}} d_j \overset{(**)}{\leq} \frac{\alpha_j}{d_j} \ + \ 2\frac{\alpha_j}{d_j} d_j = 3\alpha_j.$$

Des Weiteren gilt, da (i, j) eine indirekte Verbindung ist:

$$\alpha_j \leq c_{ij} d_j.$$

Wäre diese Ungleichung nicht erfüllt, so müsste $\beta_{ij} > 0$ gelten, da die Lösung des modifizierten Algorithmus dual zulässig ist. Damit wäre aber (i, j) mit $i \in I$ eine spezielle Kante, weshalb j nicht mit i indirekt verbunden sein könnte. In diesem Fall wäre nämlich in Phase 2 bei der Stadt j der Fall 1 eingetreten.

Damit gilt also insgesamt:

$$\frac{c_{ij} d_j}{3} \leq \alpha_j \leq c_{ij} d_j$$

bzw.

$$\frac{c_{ij} d_j}{3} \leq \alpha_j - \beta_{ij} \leq c_{ij} d_j.$$

□

6.3. Prize-Collection-Variante

Bei der Prize-Collection-Variante des Facility Location Problems fallen für jede Stadt j Strafkosten in Höhe von p_j an, falls die Stadt j nicht mit einer geöffneten Facility verbunden wurde.
Die Modellierung der Prize-Collection-Variante und die Modifikation des Algorithmus von Jain und Vazirani erfolgen wie bei Charikar et al. (siehe [1]) beschrieben. Als IP ergibt sich für dieses Problem:

$$
\begin{aligned}
min \quad & \sum_{i\in F, j\in C} c_{ij}x_{ij} + \sum_{i\in F} f_i y_i + \sum_{j\in C} r_j p_j & & \\
s.t. \quad & \sum_{i\in F} x_{ij} + r_j \geq 1 & , \ j \in C & \qquad (6.4)\\
& y_i - x_{ij} \geq 0 & , \forall i \in F, \ j \in C & \\
& x_{ij} \in \{0,1\} & , \forall i \in F, \ j \in C & \\
& y_i \ \in \{0,1\} & , \forall i \in F & \\
& r_j \ \in \{0,1\} & , \forall j \in C. &
\end{aligned}
$$

Als duales Problem zu dem zu (6.4) gehörenden linearen Problem ergibt sich:

$$
\begin{aligned}
max \quad & \sum_{j\in C} \alpha_j & & \\
s.t. \quad & \alpha_j - \beta_{ij} \leq c_{ij} & , \forall i \in F, \ j \in C & \qquad (6.5)\\
& \sum_{j\in C} \beta_{ij} \leq f_i & , \forall i \in F & \\
& \alpha_j \leq p_j & , \forall j \in C & \\
& \alpha_j \geq 0 & , \forall j \in C & \\
& \beta_{ij} \geq 0 & , \forall i \in F, j \in C. &
\end{aligned}
$$

Es stellt sich wiederum die Frage, ob durch eine Modifikation des Algorithmus von Jain und Vazirani dieses Problem approximativ gelöst werden kann.

Dazu modifizieren wir im Folgenden die Phase 1 des Algorithmus von Jain und Vazirani:
Gilt nämlich während Phase 1: $\alpha_j = p_j$, so werden α_j und die dazugehörigen β_{ij} mit $i \in I$ eingefroren, d. h. sie werden im weiteren Verlauf nicht mehr erhöht. Zusätzlich wird die Stadt j als *eingefroren* markiert.

Erläuterung 6.4. *Würden wir α_j weiterhin erhöhen, so hätten wir keine dual zulässige Lösung mehr, den es würde gelten:*

$$\alpha_j > p_j.$$

Wird nun eine Facility $i \in F$ temporär eröffnet, so wird zusätzlich zu den Vorgängen aus der ursprünglichen Phase 1 noch bei jeder eingefrorenen Stadt j, die noch nicht verbunden ist und die eine bezahlte Kante zu i besitzt, der Verbindungszeuge von j auf i gesetzt. Zusätzlich wird in diesem Fall die Stadt j als verbunden markiert.

Die modifizierte Phase 1 terminiert, sobald keine unverbundene und nicht eingefrorene Stadt mehr existiert. Ansonsten stimmt die modifizierte Phase 1 mit der ursprünglichen Phase 1 überein. Am Ende der modifizierten Phase 1 gilt nun nur noch zusätzlich für alle Städte $j \in C$:

$$r_j = \begin{cases} 1, & \text{falls} \quad j \text{ keinen Verbindungszeugen besitzt} \\ 0, & \text{sonst} \end{cases}.$$

Gilt für eine Stadt j $r_j = 1$, so wird $x_{ij} = 0$ für alle $i \in F$ gesetzt.

Bemerkung 6.5. *Bei der oben genannten modifizierten Phase 1 des Algorithmus von Jain und Vazirani kann es dazu kommen, dass eine Stadt $j \in C$ am Ende von Phase 1 keinen Verbindungszeugen besitzt.*

Die Phase 2 des Algorithmus von Jain und Vazirani wird ebenfalls modifiziert. In Phase 2 werden jetzt nicht mehr alle Städte betrachtet, sondern nur noch die, die nach Phase 1 einen Verbindungszeugen besaßen. Eine weitere Änderung in Phase 2 besteht darin, dass r_j=1 und x_{ij}=0 für alle i in F gesetzt wird, wenn eine in Phase 1 eingefrorene Stadt j als indirekt verbunden markiert werden würde. Die Stadt j wird somit nicht verbunden.

Theorem 6.6. *Durch die beschriebene Modifikation des Algorithmus von Jain und Vazirani ergibt sich ein 3-Approximations-Algorithmus für die Prize-Collection-Variante des metrischen unkapazitierten Facility Location Problems.*

Beweis: Auf Grund der Veränderung des Problems kommt es auch zu einer Veränderung der Schlupfbedingungen. Anstelle von Bedingung (3.9) muss nun für die Lösung gelten:

$$\alpha_j > 0 \Rightarrow \sum_{i \in F} x_{ij} + r_j = 1. \quad (6.6)$$

Zusätzlich muss die Lösung auch noch der folgenden Bedingung genügen:

$$r_j > 0 \Rightarrow \alpha_j = p_j. \quad (6.7)$$

Gilt nun für eine Stadt $j \in C$ $r_j = 1$, so besaß j entweder am Ende von Phase 1 keinen Verbindungszeugen oder j war eingefroren und wäre in Phase 2 indirekt verbunden worden. In jedem Fall gilt für alle $j \in C$:

$$r_j > 0 \Leftrightarrow x_{ij} = 0, \ \forall i \in F.$$

Daraus folgt:

$$\sum_{i \in F} x_{ij} + r_j = 1, \ \forall j \in C,$$

weshalb Bedingung (6.6) garantiert erfüllt ist.
Da eine Stadt nur dann am Ende nicht verbunden sein kann, wenn sie zwischenzeitlich eingefroren war, gilt:

$$r_j > 0 \Rightarrow \alpha_j = p_j. \quad (6.7)$$

Des Weiteren gilt, da α_j nicht mehr erhöht wird, sobald j eingefroren wurde:

$$\alpha_j \leq p_j.$$

Folglich liefert der modifizierte Algorithmus eine für (6.4) und (6.5) zulässige Lösung.
Lemma 5.4. gilt weiterhin, da in Phase 2 nur Städte indirekt verbunden werden, die in Phase 1 nicht eingefroren waren. Korollar 5.7 und Lemma 5.10 gelten klarerweise ebenfalls weiterhin. Daraus folgt nun, dass die Lösung des modifizierten Algorithmus den relaxierten primalen bzw. dualen Schlupfbedingungen mit Relaxierungsfaktor 3 bzw. 1 genügt. Somit folgt die Behauptung wieder aus Theorem 2.5, da die modifizierte Version weiterhin eine polynomielle Laufzeit hat. □

Bemerkung 6.7. *Würden auch Städte indirekt verbunden werden, die eingefroren waren, so würde der Beweis von Lemma 5.4. nicht mehr gelten (siehe [1]).*

6.4. Capacitated-Variante

Beim metrischen kapazitierten Facility Location Problem kann im Gegensatz zum metrischen unkapazitierte Problem jede Facility nur einen bestimmten Bedarf befriedigen.

Die nachfolgende Variante wird, wie bei Jain und Vazirani beschrieben (siehe [3]), dargestellt und gelöst. Bei ihr können an jedem Standort unbegrenzt viele Facilities errichtet werden. Werden an dem Standort $i \in F$ y_i Facilities eröffnet, so können von diesem Standort aus $u_i y_i$ Städte versorgt werden.
Weiterhin muss jede Stadt mit einer Facility verbunden werden. Als IP ergibt sich somit:

$$\begin{aligned}
&min && \sum_{i \in F, j \in C} c_{ij} x_{ij} + \sum_{i \in F} f_i y_i && \\
&s.t. && \sum_{i \in F} x_{ij} \geq 1 && , \ j \in C \qquad (6.8)\\
& && y_i - x_{ij} \geq 0 && , \forall i \in F, \ j \in C \\
& && u_i y_i - \sum_{j \in C} x_{ij} \geq 0 && , \ \forall i \in F \\
& && x_{ij} \in \{0, 1\} && , \forall i \in F, \ j \in C \\
& && y_i \geq 0 && , \forall i \in F \\
& && y \ \in \mathbb{Z} && , \forall i \in F.
\end{aligned}$$

Hierbei stellt die dritte Nebenbedingung sicher, dass von dem Standort i aus nicht mehr als $u_i y_i$ Städte versorgt werden.

Bezeichnen wir mit γ_i die zur dritten Nebenbedingung gehörigen Dualvariablen, so ergibt sich als duales Problem der linearen Relaxation:

$$
\begin{aligned}
max \quad & \sum_{j \in C} \alpha_j & & \\
s.t. \quad & \alpha_j - \beta_{ij} - \gamma_i \leq c_{ij} & , \forall i \in F,\ j \in C & \qquad (6.9)\\
& u_i \gamma_i + \sum_{j \in C} \beta_{ij} \leq f_i & , \forall i \in F & \\
& \alpha_j \geq 0 & , \forall j \in C & \\
& \beta_{ij} \geq 0 & , \forall i \in F, j \in C & \\
& \gamma_i \geq 0 & , \forall i \in F. &
\end{aligned}
$$

Da auf dieses Problem der Algorithmus von Jain und Vazirani nicht direkt angewendet werden kann, betrachten wir nun einen Spezialfall des Problems (6.9). Bei diesem Spezialfall ist γ_i keine Variable mehr, sondern es gilt:

$$\gamma_i := \frac{3f_i}{4u_i} \geq 0,\ \forall i \in F.$$

Das spezielle Problem lautet also:

$$
\begin{aligned}
max \quad & \sum_{j \in C} \alpha_j & & \\
s.t. \quad & \alpha_j - \beta_{ij} \leq c_{ij} + \frac{3f_i}{4u_i} & , \forall i \in F,\ j \in C & \qquad (6.10)\\
& \sum_{j \in C} \beta_{ij} \leq \frac{1}{4} f_i & , \forall i \in F & \\
& \alpha_j \geq 0 & , \forall j \in C & \\
& \beta_{ij} \geq 0 & , \forall i \in F, j \in C. &
\end{aligned}
$$

(6.10) ist das duale Problem der linearen Relaxation des folgenden Problems:

$$
\begin{aligned}
min \quad & \sum_{i \in F, j \in C} (c_{ij} + \frac{3f_i}{4u_i}) x_{ij} + \sum_{i \in F} \frac{f_i}{4} \tilde{y}_i & & \\
s.t. \quad & \sum_{i \in F} x_{ij} \geq 1 & , \forall j \in C & \qquad (6.11)\\
& \tilde{y}_i - x_{ij} \geq 0 & , \forall i \in F,\ j \in C & \\
& x_{ij} \geq 0 & , \forall i \in F,\ j \in C & \\
& \tilde{y}_i \ \geq 0 & , \forall i \in F. &
\end{aligned}
$$

Bei Problem (6.11) handelt es sich um ein unkapazitiertes Facility Location Problem, Folglich kann der Algorithmus von Jain und Vazirani auf dieses Problem angewendet werden. Jedoch lässt sich keine Aussage über die Güte der Lösung sagen, wenn das Problem nicht metrisch ist.

Satz 6.8. *Für die vom Algorithmus von Jain und Vazrini berechnete Lösung für (6.11) gilt:*

$$\sum_{i\in F, j\in C}(c_{ij}+\frac{3f_i}{4u_i})x_{ij}+3\sum_{i\in F}\frac{f_i}{4}\tilde{y}_i \leq 3\sum_{j\in C}\alpha_j.$$

Beweis: Ist das Problem (6.11) ein metrisches Problem, so folgt die Ungleichung aus Theorem 5.14. Würde die Ungleichung nämlich nicht gelten, so würde aus der Dualitätstheorie folgen, dass es sich bei dem Algorithmus von Jain und Vazirani nicht um einen Approximationsalgorithmus mit Approximationsgüte 3 für das metrische unkapazitierte Facility Location Problem handelt. Dies stände aber im Widerspruch zu Theorem 5.14.

Das Problem (6.10) ist ein metrisches Problem, denn es gilt:

$$c_{ij}+\frac{3f_i}{4u_i} \overset{c_{ij} metrisch}{\leq} c_{ij'}+c_{i'j'}+c_{i'j}+\frac{3f_i}{4u_i} \leq c_{ij'}+\frac{3f_i}{4u_i}+c_{i'j'}+\frac{3f_{i'}}{4u_{i'}}+c_{i'j}+\frac{3f_{i'}}{4u_{i'}}.$$

□

Sei nun

$$y_i := \left\lceil \frac{\sum_{j\in C} x_{ij}}{u_i} \right\rceil,$$

dann gilt:

$$y_i \leq \tilde{y}_i + \frac{\sum_{jinC} x_{ij}}{u_i}. \ (*)$$

Diese Ungleichung gilt, da $x_{ij} > 0 \Rightarrow \tilde{y}_i = 1$.

Theorem 6.9. *Bestimmen wir mit Hilfe des Algorithmus von Jain und Vazirani eine Lösung $(x, \tilde{y})$ für das Problem (6.11) und wählen anschließend y wie oben angegeben, so erhalten wir einen 4-Approximations-Algorithmus für das beschriebene Problem.*

Beweis: Zunächst wird gezeigt, dass (x, y) eine zulässige Lösung für (6.8) handelt. Hierbei ist nur zu überprüfen, ob

$$u_i y_i - \sum_{j\in C} x_{ij} \geq 0, \ \forall i \in F \text{ gilt.}$$

Dass alle andere Nebenbedingungen gelten, folgt nämlich unmittelbar aus der Definition von y_i und der Tatsache, dass $(x, \tilde{y})$ eine zulässige Lösung für das Problem (6.11) ist.
Es gilt nun:

$$u_i y_i = u_i \left\lceil \frac{\sum_{j\in C} x_{ij}}{u_i} \right\rceil \geq u_i \frac{\sum_{j\in C} x_{ij}}{u_i} = \sum_{j\in C} x_{ij}, \ \forall i \in F.$$

Damit handelt es sich also bei (x, y) um eine zulässige Lösung für das Problem (6.8).

Des Weiteren gilt:

$$3\sum_{j\in C}\alpha_j \overset{Satz\ 6.8}{\geq} \sum_{i\in F, j\in C}(c_{ij}+\frac{3f_i}{4u_i})x_{ij}+3\sum_{i\in F}\frac{f_i}{4}\tilde{y}_i = \sum_{i\in F, j\in C}c_{ij}x_{ij}+\frac{3}{4}\sum_{i\in F}f_i(\tilde{y}_i+\frac{\sum_{j\in C}x_{ij}}{u_i})$$
$$\overset{(*)}{\geq} \sum_{i\in F, j\in C}c_{ij}x_{ij}+\frac{3}{4}\sum_{i\in F}f_iy_i.$$

Also gilt:

$$4\sum_{j\in C}\alpha_j \geq \frac{4}{3}\sum_{i\in F, j\in C}c_{ij}x_{ij}+\sum_{i\in F}f_iy_i \geq \frac{4}{3}\sum_{i\in F, j\in C}c_{ij}x_{ij}+\sum_{i\in F}f_iy_i.$$

Mit Hilfe dieser Ungleichung folgt die Behauptung nun aus Satz 2.4a, da der modifizierte Algorithmus immer noch eine polynomielle Laufzeit besitzt. □

6.5. Abhängigkeit der Eröffnungskosten von der Anzahl der versorgten Städte

In der Realität sind die Eröffnungskosten einer Facility meist nicht unabhängig von der Anzahl der Städte, die diese Facility versorgt. In der Folgenden Variante des metrischen unkapazitierten Facility Location Problems wird dieser Sachverhalt berücksichtigt. Die Eröffnungskosten von Facility i sind nun gegeben durch $s_i + k_it_i$, wobei k_i die Anzahl der Städte angibt, die i versorgt. Des Weiteren gilt: $t_i, s_i \in \mathbb{R}^+$.

Folglich ergibt sich als neues IP:

$$\begin{aligned}
&min && \sum_{i\in F, j\in C}c_{ij}x_{ij}+\sum_{i\in F}(s_i+k_it_i)y_i && && \\
&s.t. && \sum_{i\in F}x_{ij}\geq 1 && ,\ j\in C && (6.12)\\
& && y_i - x_{ij}\geq 0 && ,\forall i\in F,\ j\in C && \\
& && \sum_{j\in C}x_{ij}=k_i && ,\forall i\in F && \\
& && x_{ij}\in\{0,1\} && ,\forall i\in F,\ j\in C && \\
& && y_i\ \in\{0,1\} && ,\forall i\in F. &&
\end{aligned}$$

Setzen wirdie dritte Nebenbedingung in die Zielfunktion von (6.12) ein, so ergibt sich als neue Zielfunktion:

$$min \sum_{i\in F, j\in C}c_{ij}x_{ij}+\sum_{i\in F}s_iy_i+\sum_{i\in F, j\in C}x_{ij}t_i$$

Hierbei wurde ausgenutzt, dass für die Lösung des Problems gilt:

$$y_i > 0 \Leftrightarrow \text{ Es existiert ein j} \in \text{C: } x_{ij} > 0.$$

Theorem 6.10. *Der Algorithmus von Jain und Vazirani ist ein 3-Approximations-Algorithmus für diese Variate.*

Beweis: Damit wir den Algorithmus von Jain und Vazirani anwenden können, müssen wir zeigen, dass es sich bei (6.11) um ein metrisches unkapazitiertes Facility Location Problem handelt.

Sei dazu nun:
$f_i := s_i,\ \forall i \in F.$
$\tilde{c_{ij}} = c_{ij} + t_i,\ \forall i \in F, j \in C.$

Dann lässt sich (6.11) in ein Problem der Form (3.1) und damit in ein unkapazitiertes Facility Location Problem mit den Verbindungskosten $\tilde{c_{ij}}$ und den Eröffnungkosten f_i umschreiben. Des Weiteren gilt:

$$\tilde{c_{ij}} = c_{ij} + t_i \leq c_{ij'} + t_i + c_{i'j} + c_{i'j'} \leq \tilde{c_{ij'}} + \tilde{c_{i'j}} + \tilde{c_{i'j'}}.$$

Folglich ist (6.11) zusätzlich ein metrisches Problem, weshalb die Behauptung nun aus Theorem 5.11 folgt. □

7. Fazit

Wie in der Einleitung erwähnt, wurde bisher noch kein Algorithmus gefunden, der das unkapazitierte Facility Location Problem in polynomieller Zeit löst. In Kapitel 3 konnte gezeigt werden, dass das unkapazitierte Facility Location Problem ein NP-schweres Problem ist und somit nicht in polynomieller Zeit lösbar ist, sofern nicht gilt: $NP \subset P$.
Eine interessante Möglichkeit zur Lösung des metrischen unkapazitierten Facility Location Problems stellt der Approximationsalgorithmus von Jain und Vazirani dar. Er basiert auf dem Primal-Dual-Verfahren und ist auf Grund der Interpretationsmöglichkeit der Dualvariablen als „bezahlte Kosten“ sehr anschaulich. Ein weiterer Vorteil dieses Algorithmus besteht darin, dass er in einer kurzen Laufzeit von $O(m\ log\ m)$ eine Lösung berechnet. Diese kann jedoch bis zu 3-mal schlechter als die Optimallösung sein. Aus der Sensitivitätsanalyse geht hervor, dass häufig trotz Veränderung einzelner Daten keine komplette Neuberechnung erforderlich ist, um weiterhin garantieren zu können, dass die Lösung zulässig und maximal 3-mal schlechter als die Optimallösung ist.
Der Algorithmus von Jain und Vazirani eignet sich außerdem auch zur approximativen Lösung einiger Varianten des metrischen unkapazitierten Facility Location Problems, wie zum Beispiel der Arbitrary-demands-Variante und der Prize-Collection-Variante. Darüber hinaus kann der Algorithmus von Jain und Vazirani sogar zur approximativen Lösung einer Variante des metrischen unkapazitierten Facility Location Problems herangezogen werden. Der Algorithmus von Jain und Vazirani ist somit vielseitig anwendbar.

Literaturverzeichnis

[1] Charikar, Moses ; Khuller, Samir ; Mount, David M. ; Narasimhan, Giri: Algorithms for facility location problems with outliers. In: *SODA '01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA : Society for Industrial and Applied Mathematics, 2001. – ISBN 0–89871–490–7, S. 642–651

[2] Cormen, Thomas H. ; Leiserson, Charles E. ; Rivest, Ronald L. ; Stein, Clifford: *Introduction to Algorithms*. Second. Cambridge, MA : MIT Press, 2001. – ISBN 0–262–03293–7

[3] Jain, Kamal ; Vazirani, Vijay V.: Approximation algorithms for metric facility location and k-Median problems using the primal-dual schema and Lagrangian relaxation. In: *J. ACM* 48 (2001), Nr. 2, S. 274–296. `http://dx.doi.org/http://doi.acm.org/10.1145/375827.375845`. – DOI http://doi.acm.org/10.1145/375827.375845. – ISSN 0004–5411

[4] Karp, Richard M.: Reducibility Among Combinatorial Problems. In: Miller, R. E. (Hrsg.) ; Thatcher, J. W. (Hrsg.): *Complexity of Computer Computations*. Plenum Press, 1972, S. 85–103

[5] Korte, Bernhard ; Vygen, Jens: *Combinatorial optimization. Theory and applications. 3rd ed.* Algorithms and Combinatorics 21. Berlin: Springer, 2006

[6] Kuehn, A. A. ; Hamburger, M. J.: A heuristic program for locating warehouses. In: *Manage. Sci. 9* (1963), S. 643–666

[7] Vazirani, Vijay V.: Primal-dual schema based approximation algorithms. (2002), S. 198–207. ISBN 3–540–43328–7

A. Erhöhungsrate von α_j im Algorithmus von Jain und Vazirani

Würde man den Algorithmus von Jain und Vazirani implementieren, so würde man während des Algorithmus die α_j nicht mit der Zeit zusammen ansteigen lassen, sondern immer direkt um einen größeren Wert erhöhen.
Im Folgenden möchten wir nun herleiten, um wie viel die zu erhöhenden α_j zu jedem Zeitpunkt maximal erhöht werden dürfen, ohne dass eines der in Phase 1 auftretenden Ereignisse übergangen wird. Den Wert, um den die α_j zu dem jeweiligen Zeitpunkt erhöht werden dürfen, bezeichnen wir mit Erhöhungrate e.

Die für die Bestimmung der Erhöhungsrate e relevanten Ereignisse sind die Markierung einer Kante (i, j) als bezahlt und die temporäre Eröffnung einer Facility i'.
Wäre nämlich die Erhöhungsrate e der α_j so hoch, dass nach der Erhöhung der α_j für eine unbezahlte Kante (i, j') gelten würde: $\alpha_{j'} > c_{ij'}$, so hätten wir keine dual zulässige Lösung mehr, da weiterhin $\beta_{ij'} = 0$ gelten würde. $\beta_{ij} = 0$ würde weiterhin gelten, da β_{ij} erst dann zusammen mit α_j erhöht wird, wenn die Kante (i, j) schon bereits als bezahlt markiert wurde.
Des Weiteren hätten wir auch keine dual zulässige Lösung mehr, wenn nach der gleichzeitigen Erhöhung aller α_j um e für eine nicht temporär geöffnete Facility i gelten würde:

$$\sum_{j \in C} \beta_{ij} > f_i.$$

Damit nun zu keinem Zeitpunkt das Ereignis, der Markierung einer Kante (i, j) als bezahlt, übergangen wird, muss e nun immer so gewählt werden, dass für alle unbezahlten Kanten (i, j), bei denen α_j erhöht wird, gilt:

$$e \leq c_{ij} - \alpha_j. \quad (*)$$

Des Weiteren muss e auch zu jedem Zeitpunkt die Ungleichung

$$e \leq \frac{1}{|B_i|}(f_i - \sum_{j \in C} \beta_{ij}). \quad (**)$$

für jede nicht temporär eröffnete Facility i erfüllen. Bei der Ungleichung $(**)$ bezeichnet B_i die Menge aller Städte, die eine bezahlte Kante zu i haben, aber noch mit keiner Facility verbunden sind. B_i ist somit die Menge aller Städte, deren Beitrag an den Eröffnungskosten von i mit der nächsten Erhöhung steigt.

B. Erhöhungsrate e' bei der Arbitrary-demands-Variante

Analog zum Anhang A wird im Folgenden erläutert, wie zu jedem Zeitpunkt in der modifizierten Phase 1 die Erhöhungsrate e' gewählt werden muss, so dass keines der relevanten Ereignisse aus Phase 1 übergangen wird.
Die hierfür relevanten Ereignisse sind die Markierung einer Kante $(i, j) \in E$ als bezahlt und die temporäre Eröffnung einer Facility $i' \in F$.
Damit nun zu keinem Zeitpunkt das Ereignis der Markierung einer Kante (i, j) als bezahlt, übergangen wird, muss für alle unbezahlten Kanten (i, j), bei denen α_j erhöht wird, gelten:

$$e' \leq c_{ij} - \frac{\alpha_j}{d_j}.$$

Würde diese Ungleichung für eine unbezahlte Kante (i', j'), bei der $\alpha_{j'}$ erhöht wird, nicht gelten, so würde nach der Erhöhung aller zu erhöhenden α_j und β_{ij} für eine nicht-bezahlte Kante (i', j') gelten:

$$\alpha_{j'_{nach\ Erhöhung}} = \alpha_{j'_{vor\ Erhöhung}} + e'd_j > \alpha_{j'_{vor\ Erhöhung}} + c_{ij}d_j - \frac{\alpha_{j_{vor\ Erhöhung}}}{d_j}d_j = c_{ij}.$$

Damit zu keinem Zeitpunkt das Ereignis der Markierung einer Facilty $i \in F$ als temporär geöffnet übergangen wird, muss für jede nicht temporär geöffnete Facility i' gelten:

$$e' \leq \frac{1}{\sum_{j \in \tilde{B_{i'}}} d_j}(f_i - \sum_{j \in C} \beta_{ij})$$

Hierbei bezeichnet $\tilde{B_{i'}}$ die Menge aller Städe $j \in C$, die mit i' eine bezahlte Kante haben und die noch nicht verbunden sind.