

A Polynomial Combinatorial Algorithm for Generalized Minimum Cost Flow

Naum Kocherovskiy

14. December 2009

Inhaltsverzeichnis

1	Einleitung	1
2	Grundlagen	1
2.1	Voraussetzungen	1
2.2	Residualnetzwerke	2
2.3	Optimalitätsbedingungen	2
3	Minimum-Ratio Circuit Canceling Algorithmus	3
4	Algorithmus mit verbesserter Laufzeit	5
5	Runden auf eine Ecke	6
6	Minimum Ratio Circuit Subroutine	6
6.1	Entdeckung von Circuit	6
6.2	Entdeckung von Circuits mit negativen Kosten	6
6.3	Entdeckung von Circuit mit minimalem Quotient	7
7	Offenes Problem	7

1 Einleitung

Ein grundlegendes kombinatorisches Optimierungsproblem ist das Minimum-Cost-Flow-Problem, dessen Ziel darin liegt, eine Flußmenge durch ein Flußnetzwerk zu möglichst minimalen Kosten zu senden. Generalized Minimum-Cost-Flow-Problem ist eine Verallgemeinerung davon, wobei jede Kante e einen positiven Multiplikator $\gamma(e)$, den sogenannten "gain factor" besitzt. Wenn dieser Faktor > 1 ist, handelt es sich um einen Gewinnfaktor und beim Faktor < 1 geht es um einen Verlustfaktor. Das Problem tritt in zahlreichen Anwendungen auf. Faktoren können dabei verschiedene Transformationen, wie z.B. den Währungskurs darstellen.

Da dieses Problem ein Spezialfall der linearen Programmierung ist, kann es in polynomieller Zeit z.B. mit Hilfe von Ellipsoidmethode oder Innere-Punkte-Verfahren gelöst werden. K. Wayne entwickelt einen schnellen kombinatorischen Algorithmus. Die in seiner Studie vorgestellten Algorithmen lösen eigentlich ein äquivalentes Generalized Minimum-Cost-Circulation (GMCC) Problem, wobei alle Überschüsse und Defizite gleich 0 sind.

Obwohl die besten Innere-Punkte-Verfahren optimale Lösungen schneller als die kombinatorischen Algorithmen finden, sind die letzten beim Berechnen von approximativ optimalen Lösungen bei relativ großen Eingabezahlen besser.

2 Grundlagen

2.1 Voraussetzungen

Sei $G = \{V, E, u, c, \gamma\}$ ein generalisiertes Netzwerk, wobei (V, E) ein gerichteter Graph mit Knoten V und Kanten E ist.

$u : E \rightarrow \mathfrak{R}_{\geq 0}$ bezeichne die Kapazitätsfunktion.

$c : E \rightarrow \mathfrak{R}$ bezeichne die Kostenfunktion.

$\gamma : E \rightarrow \mathfrak{R}_{> 0}$ bezeichne die Gewinnfunktion.

Seien $n = |V|$ und $m = |E|$.

Eine generalisierte Zirkulation $g : E \rightarrow \mathfrak{R}_{\geq 0}$ ist eine nicht negative Funktion, wenn gilt:

$$\forall v \in V : \sum_{w \in V : (v,w) \in E} g(v,w) = \sum_{w \in V : (w,v) \in E} \gamma(w,v)g(w,v).$$

Die Funktion heißt *zulässig* (feasible), falls zusätzlich gilt:

$$\forall (v,w) \in E : g(v,w) \leq u(v,w).$$

$c(g) = \sum_{(v,w) \in E} c(v,w)g(v,w)$ bezeichne die Kostenfunktion einer Zirkulation.

Das Generalized Minimum-Cost-Circulation Problem besteht darin, eine zulässige generalisierte Zirkulation mit minimalen Kosten zu finden. Außerdem wird nach guten Zirkulationen gesucht, die nicht unbedingt optimal sind. Eine ε -optimale generalisierte Zirkulation ist eine Zirkulation, deren Wert sich um $(1 + \varepsilon)$ vom optimalen Wert unterscheidet.

2.2 Residualnetzwerke

Sei \bar{e} die Rückwärtskante für alle $e \in E$. $\bar{E} = \{\bar{e} : e \in E\}$ bezeichne die Menge der Rückwärtskanten. Die Rückwärtskanten existieren im originalen Netzwerk nicht. Das könnte man sich so vorstellen, als ob es virtuelle Kanten wären. Sie entstehen erst im Residualnetzwerk. Der Faktor von \bar{e} ist $1/\gamma(e)$ und die Kosten sind $-c(e)/\gamma(e)$. Beim Berechnen von Kosten/Faktor gehen wir davon aus, dass sich nichts ändern darf, wenn wir den Fluß auf der Kante und dann zurück auf der Rückwärtskante senden.

Sei g eine generalisierte Zirkulation im Netzwerk G . Die Restkapazitätsfunktionen von e bzw. \bar{e} sind durch $u_g(e) = u(e) - g(e)$ bzw. $u_g(\bar{e}) = \gamma(e)g(e)$ definiert. Sei $E_g \subseteq E \cup \bar{E}$ die Menge der Kanten mit positiver Restkapazität. $G_g = \{V, E_g, u_g, c, \gamma\}$ ist ein Residualnetzwerk. Lösen des Problems im Residualnetzwerk ist zum Lösen des Problems im originalen Netzwerk äquivalent. Die Residualkreise und -zirkulationen sind Kreise bzw. Zirkulationen im Residualnetzwerk.

2.3 Optimalitätsbedingungen

Gewinnfaktor vom Kreis ist ein Produkt der Gewinnfaktoren von Kanten, die zu diesem Kreis gehören.

Unit-gain-cycle ist ein Kreis, dessen Gewinn gleich eins ist. *Flussgenerierendes (Flussabsorbierendes) cycle* ist ein Kreis, dessen Gewinn größer (kleiner) eins ist. *Bicycle* ist ein Kreis, der aus einem Flussgenerierenden cycle, einem Flussabsorbierenden cycle und dem Pfad vom ersten zum zweiten besteht (Bsp. Abbildung 1).

Wenn man einen Unit-gain-cycle oder Bicycle im Residualnetzwerk hat, kann man den Fluß auf seinen Kanten unter der Berücksichtigung der Kapazitätsbedingungen erhöhen. Falls mindestens eine der Kanten gesättigt wird (d.h. Fluß wird gleich der Kapazität), können wir diesen Fluß zum gegebenen g "addieren" und ein neues Residualnetzwerk berechnen. In diesem neuen Residualnetzwerk wird der aus einem Schritt davor existierende Unit-gain-cycle or Bicycle nicht mehr existieren. Diese Operation wird *Canceling* genannt.

Ein *Circuit* ist eine Zirkulation, die einen positiven Fluß durch die Kanten eines einzelnen Unit-gain-cycle oder Bicycle im Residualnetzwerk sendet.

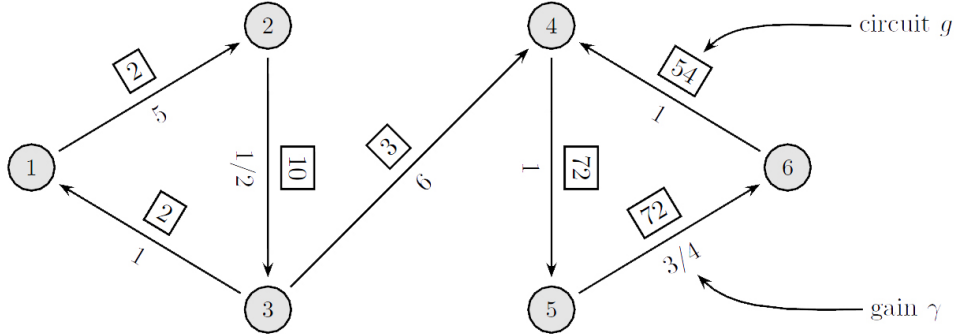


Abbildung 1: Bicycle und der entsprechende Circuit.

Lemma 1. *Eine generalisierte Zirkulation g kann in Komponenten g_1, \dots, g_k mit $k \leq m$ zerlegt werden, so dass $g = \sum_i g_i$ und jede Komponente g_i ein Circuit ist, der nur auf den Kanten e mit $g(e) > 0$ positiv ist.*

Theorem 2. *Eine zulässige generalisierte Zirkulation g ist genau dann optimal, wenn das Residualnetzwerk G_g keinen Circuit mit negativen Kosten enthält.*

3 Minimum-Ratio Circuit Canceling Algorithmus

Der Minimum-Ratio Circuit Canceling Algorithmus ist ein einfacher, aber nicht der effizienteste Algorithmus. Die Idee besteht darin, dass man wiederholend die Canceling-Operation auf den Circuit mit minimalem Quotient durchführt.

Als erstes sollte man sich auf eine Regel einigen um die Kosten von Kreisen zu bestimmen. Die bekannteste Formel dafür wurde von Goldberg und Tarjan entwickelt: Die durchschnittlichen Kosten vom Kreis Γ ist das Verhältnis von Kosten zu der Anzahl von Kanten: $\sum_{e \in \Gamma} c(e) / |\Gamma|$.

Wallacher schlägt eine andere Strategie vor, die offensichtlich besser für das gegebene Problem geeignet ist. Sei $t : E \rightarrow \mathbb{R}_{\geq 0}$ eine nicht negative Zeitfunktion, mit $t(e) = 1/u_g(e)$. Der Quotient von Γ ist dann folgendermaßen definiert: $\sum_{e \in \Gamma} c(e) / \sum_{e \in \Gamma} t(e)$.

Aufgrund der Existenz von Gewinn-Faktoren muss die Formel so angepasst werden, dass der Fluß auf den Kanten berücksichtigt wird. Der Algorithmus führt iterativ die "Canceling"-Operation auf die Kreise mit minimalem Quotient(ratio): $\mu = \sum_{e \in \Gamma} c(e)x(e) / \sum_{e \in \Gamma} t(e)x(e)$. Der Algorithmus terminiert, wenn die Lücke zwischen der aktuellen und der optimalen Zirkulation klein genug wird. Nach einer polynomieller Anzahl von Iterationen,

wird diese Lücke klein genug um zu garantieren, dass die aktuelle Zirkulation ε -optimal ist.

Algorithmus 1 Minimum Ratio Algorithm

Input: generalized network G and precision parameter ε

Output: ε -optimal circulation g

Initialize $g \leftarrow 0$

for $i := 1$ to $\lceil m2\ln(1/\varepsilon) \rceil$ **do**

$\forall e \subseteq E_g : t(e) = 1/u_g(e)$

Cancel a min ratio circuit in G_g

Update g

end for

Die nächsten zwei Lemmas besagen, dass die Optimalitätslücke mit jeder Iteration geometrisch schrumpft.

Lemma 3. *Sei g eine zulässige Zirkulation. Sei X ein Circuit aus G_g mit negativen Kosten, und sei μ sein Quotient. Nach “Canceling”-Operation wird die Zielfunktion um mindestens μ verbessert.*

Beweis. Da X negative Kosten hat, ist μ auch negativ. Nach der “Canceling”-Operation wird mindestens eine Kante gesättigt und folglich $t(x) = \sum_{e \in E_g} x(e)/u_g(e) \geq 1$. Der Quotient μ bleibt unverändert. Somit folgt: $c(x) = \mu t(x) \leq \mu \leq 0$.

Lemma 4. *Sei g eine zulässige Zirkulation, die nicht optimal ist. Sei μ^* der minimale Quotient des Circuit aus G_g . $OPT(G_g)$ sind die Kosten der optimalen Zirkulation in G_g . Dann gilt: $OPT(G_g) \leq \mu^* \leq OPT(G_g)/m < 0$.*

Beweis. Da g nicht optimal ist, muss G_g mindestens einen Circuit mit negativen Kosten enthalten (Theorem 2). Daraus folgt dass ein oder mehrere Circuits mit negativem Quotient existieren. Also $Opt(G_g) < 0, \mu^* < 0$. Letzte Ungleichung ist somit erfüllt.

Jeder beliebige Circuit x mit Quotient μ^* hat negative Kosten $c(x)$ und man kann Lemma 3 anwenden. Das heißt die Zielfunktion kann man mindestens um μ^* verbessern. Andererseits ist $Opt(G_g) = Opt(G) - c(g)$ die bestmögliche Verbesserung. Daraus folgt: $Opt(G_g) \leq \mu^*$.

Sei x^{OPT} eine optimale Zirkulation in G_g . Nach der Definition von μ^* haben wir $\mu^* \leq c(x^{OPT})/t(x^{OPT})$. Wir wissen: $t(x^{OPT}) = \sum_e x^{OPT}(e)/u_g(e)$. Da x^{OPT} zulässig ist, gilt: $t(x^{OPT}) \leq m$. Nach dem Kombinieren der beiden Bedingungen folgt:

$$\mu^* \leq c(x^{OPT})/t(x^{OPT}) \leq c(x^{OPT})/m = OPT(G_g)/m.$$

Theorem 5. *Der Minimum ratio circuit-canceling Algorithmus führt $\mathcal{O}(m \log(1/\varepsilon))$ “Canceling”-Operationen durch, um eine ε -optimale generalisierte Zirkulation mit minimalen Kosten zu finden. Um eine optimale*

Lösung zu finden sind es $\mathcal{O}(m^2 \log B)$ "Canceling"-Operationen nötig. Jede "Canceling"-Operation benötigt $\tilde{\mathcal{O}}(mn^3)$.

Beweis. Nach dem Kombinieren von Lemma 3 und 4 erhält man, dass jede Iteration die Optimalitätslücke um mindestens einen Faktor $1 - 1/m$ verringert. Denn $Opt(G_g) = Opt(G) - c(g) \geq Opt(G_g) - \mu^* \geq Opt(G_g) - \frac{1}{m}Opt(G_g) = (1 - \frac{1}{m})Opt(G_g)$.

Wir suchen eine ε -optimale Lösung.

$$\begin{aligned} (1 + \varepsilon)c(g) &\leq Opt(G) \\ \Rightarrow Opt(G) - Opt(G_g) &\leq \frac{1}{1+\varepsilon}Opt(G) \\ \Rightarrow Opt(G_g) &\geq \frac{\varepsilon}{1+\varepsilon}Opt(G) \\ \Rightarrow Opt(G_g) &\geq \varepsilon^2 Opt(G) \end{aligned}$$

Andererseits wissen wir: $Opt(G_g) \geq (1 - \frac{1}{m})^k Opt(G_{g_0}) = (1 - \frac{1}{m})^k Opt(G)$

Es genügt also z.z. $(1 - \frac{1}{m})^k \leq \varepsilon^2$

Da $(1 - 1/m)^m \leq 1/e$ für $m \geq 2$ gilt (Eigenschaft der Eulerschen Zahl¹), gibt es folglich $\mathcal{O}(m \log(1/\varepsilon))$ Iterationen. Erheben wir die beiden Seiten der Ungleichung in eine Potenz von $\ln(1/\varepsilon^2)$:

$$(1 - 1/m)^{m2\ln(1/\varepsilon)} = ((1 - 1/m)^m)^{\ln(1/\varepsilon^2)} \leq (1/e)^{\ln(1/\varepsilon^2)} = (e^{-1})^{-2\ln\varepsilon} = \varepsilon^2.$$

Also man braucht $m2\ln(1/\varepsilon)$ Iterationen, was auch in der for-Schleife im Algorithmus steht.

Die Laufzeitanalyse für die optimale Lösung bzw. Subroutine vom Algorithmus macht der Autor in §5-§6 des Papers.

4 Algorithmus mit verbesserter Laufzeit

Der modifizierte Algorithmus basiert auf der Idee, dass es eigentlich nicht notwendig ist, den Circuit mit genau minimalem Quotient zu "canceln". Die "Canceling"-Operation eines Circuit mit minimalem Quotient verbessert die Zielfunktion um mindestens μ^* , wo μ^* den minimalen Quotient darstellt. Dies benötigt $\tilde{\mathcal{O}}(mn^3)$ Zeit.

Die verbesserte Version des Algorithmus "cancelt" Circuits in Phasen. Jede Phase ist durch den Parameter $\mu < 0$ charakterisiert, der zu Beginn der Phase mindestens so klein ist wie $\mu^*/2$. Man "cancelt" nun Circuits, deren Quotient mindestens kleiner als μ ist ($c_\mu < 0$). Dafür wird eine neue Funktion $c_\mu(e) := c(e) - \mu t(e)$ eingeführt und man "cancelt" Circuits, deren $c_\mu < 0$ ist.

Die Zeit, die man benötigt, um Circuits mit negativen Kosten zu identifizieren ist $\tilde{\mathcal{O}}(mn^2)$. Somit verbessert man den gesamten Algorithmus um den Faktor n . Wenn alle solche Circuits entdeckt sind, wird μ halbiert. Nachdem μ ziemlich nah an 0 kommt, wird die Optimalitätslücke klein genug, um eine approximative Lösung zu garantieren.

¹<http://de.wikipedia.org/wiki/Eulerschezahl>

5 Runden auf eine Ecke

Die in diesem Abschnitt vorgestellte Methode wird dazu benötigt, um eine approximative Zirkulation in eine optimale Zirkulation zu "runden".

Man konvertiert eine beliebige zulässige Zirkulation in eine Zirkulation, die einem Eckknoten (Vertex) des Polytops entspricht und deren Kosten nicht schlechter als der aktuellen Zirkulation sind: $c(g') \leq c(g)$. Der Algorithmus braucht dafür $\mathcal{O}(m^2n)$. Falls g ε -optimal war, ist g' auch ε -optimal.

Basierend auf den Grundsätzen der Linearen Programmierung, Kramerschen Regel und Manipulationen mit Formeln gilt für zwei Vertex-Zirkulationen g_1, g_2 mit $c(g_1) \neq c(g_2)$: $|c(g_1) - c(g_2)| \geq B^{-4m}$. Der Autor zeigt, dass für g_1 optimal und g_2 B^{-7m} -optimal, gilt: $|c(g_1) - c(g_2)| < B^{-4m}$, d.h. g_2 ist auch optimal. Daraus folgt: Falls Vertex-Zirkulation B^{-7m} -optimal ist, dann ist sie optimal. Insgesamt folgt: Wenn g B^{-7m} -optimal war, ist g' optimal.

6 Minimum Ratio Circuit Subroutine

6.1 Entdeckung von Circuit

Bei der Entdeckung von Circuits wird die Bellman-Ford Methode benutzt:

$$\begin{aligned}\pi(v) &\leftarrow \min\{\pi(v), \pi(w)\gamma(v, w)\}. \\ \pi(w) &\leftarrow \max\{\pi(w), \pi(v)\gamma(v, w)\}.\end{aligned}$$

wobei π Knoten-Potenzial ist.

Im ersten Schritt sucht man nach Bicycles. Der Algorithmus sucht nach der Punktmenge N , deren Punkte entweder auf dem Flow-Absorbing-cycle liegen oder es gibt einen Pfad, der zu einem Flow-Absorbing-cycle führt. Danach suchen wir aus der Menge N den Flow-Generating-cycle. Im zweiten Schritt sucht man nach Unit-Gain-cycle in $V \setminus N$ und in N .

6.2 Entdeckung von Circuits mit negativen Kosten

Die Aufgabe besteht darin, die Circuits mit negativen Kosten zu finden, die im Algorithmus im §4 benötigt werden.

$$c(x) < 0, x \text{ circulation} \tag{1}$$

Zu dem Problem einen Circuit mit negativen Kosten zu finden (1) ist folgendes Problem dual:

$$\forall (v, w) \in E : c^\pi(v, w) := c(v, w) + \pi(v) - \gamma(v, w)\pi(w) \geq 0 \tag{2}$$

(1) und (2) können nicht gleichzeitig erfüllt werden. Bei der Gleichung (2) handelt es sich um ein 2VPI(2 variables per inequality) System, d.h es gibt

zwei Variablen pro Ungleichung und das ist eine spezielle LP-Formulierung für das Problem. Um dieses Problem zu lösen, existieren viele Algorithmen (LP-Solver), die entweder eine gültige Lösung oder die Information darüber liefern, dass es keine Lösung gibt, den sogenannten “certificate of infeasibility”. Dieses Zertifikat entspricht dem Circuit mit negativen Kosten.

6.3 Entdeckung von Circuit mit minimalem Quotient

Für ein gegebenes μ können wir entweder einen Circuit finden, dessen Quotient kleiner als μ ist oder bestimmen, dass es so einen Circuit nicht existiert ($ratio < \mu \Leftrightarrow c_\mu < 0$). Nach μ^* sucht man durch Schätzung. Man startet mit einem schätzungsweise gewählten μ . Falls man einen Circuit mit Quotient $< \mu$ findet, kann man μ entsprechend dem binären Suchprinzip verringern, ansonsten vergrößern. Basierend auf dem “straightforward binary search framework” kann man μ^* in $\tilde{O}(m^2 n^2 \log B)$ finden. Mit “Megiddo’s parametric search framework” bekommt man einen $\tilde{O}(mn^3)$ Algorithmus.

7 Offenes Problem

Bis heute wurden nur schwach-polynomielle Algorithmen für das Generalized Minimum Cost Flow Problem gefunden. Ein stark-polynomieller Algorithmus stellt immer noch eine Herausforderung dar. 2VPI-Optimierung ist ebenso ein offenes Problem. Andererseits sind stark-polynomielle Algorithmen für das traditionelle Minimum Cost Flow Problem bekannt.