

# Der Linearzeit MST-Algorithmus

## Der schnellste Algorithmus für das MST/ MSF Problem

Maximilian Springenberg

Proseminar Randomisierte Algorithmen

### 0 Konventionen

---

In dieser Ausarbeitung wird sich an die üblichen Konventionen zur Notation von Variablen in Graphen orientiert. So definieren wir einen Graphen als  $G = (V, E)$ , mit der Anzahl von Knoten  $n = |V|$  und Kanten  $m = |E|$ . Zu  $G$  zugehörige Kanten und Knotenmengen können auch durch  $V_G, E_G$  als solche gekennzeichnet werden. Für jeden Knoten  $v$  nehmen wir an, dass eine Adjazenzliste  $adj(v)$  von adjazenten Knoten für ihn vorhanden ist. Ferner betrachten wir ungerichtete gewichtete Graphen und bezeichnen die Gewichtungsfunktion als  $w : E \rightarrow \mathbb{R}$ .

Wir werden auch Wege, bzw. Pfade betrachten. Dazu sei  $P(\{v, u\})$ , eine Funktion die in einem Baum den Pfad von  $v$  nach  $u$  als Knotenfolge angibt. Da es für uns handlicher sein wird gleich die Kanten angegeben zu haben definieren wir  $P_e$  als die Folge von Kanten auf dem Pfad.

### 1 Motivation

---

#### 1.1 MST und MSF

Der minimale Spannbaum, oder auch MST, stellt einen azyklischen zusammenhängenden Teilgraph aus  $G$ , der alle Knoten verbindet und dessen Summe von Kantengewichte  $\sum_{e \in E_{MST}} w(e)$  minimal ist dar. Ist  $G$  selbst nicht zusammenhängend, so werden wir einen minimalen Spannwald als nächst beste Lösung betrachten. Ein minimaler Spannwald besteht aus einer Sammlung von minimalen Spannbäumen für die verbundenen Komponenten in  $G$ .

#### 1.2 Vorteile eines nichtdeterministischen Ansatz

Wir werden in dieser Ausarbeitung zum Kapitel 10.3 aus dem Buch „Randomized Algorithms“ von Motwani, R., Raghavan, P. einen randomisierten Ansatz für einen Algorithmus, der einen minimalen Spannbaum in erwarteter Linearzeit approximiert betrachten.

Das MST Problem ist in  $P$ , und es sind bereits deterministische Algorithmen wie die von Prim, Kruskal, Borůvka bekannt, die mit einer Worst-Case Laufzeitschranke von  $O(m * \log(n))$  das Problem lösen. Zudem existiert der Algorithmus von Bernard Chazelle, für den eine Worst-Case Laufzeitschranke von  $O(m * \log \beta(m, n))$  bekannt ist, wobei mit  $\beta(m, n) = \{i | \log^{(i)} n \leq m/n\}$  die inverse Ackermann Funktion

verwendet wird.  $\log^{(i)} n$  ist hierbei die  $i$ -te Anwendung von  $\log$  auf  $n$ . Dementsprechend steigt die Funktion  $\beta$  so schwach, dass die aus ihr resultierenden Faktoren für die Worst-Case Laufzeit hinsichtlich der Größe von Graphen in der Praxis als nahezu konstant angesehen werden.

Wozu dient also ein nichtdeterministischer Algorithmus, der im Erwartungswert linear verläuft? - Die Antwort auf diese Frage kann sowohl durch die komplexe Implementierung des Algorithmus von Chazelle, als auch der Stabilität, bzw. Güte, des vorgestellten Algorithmus hinsichtlich seiner Laufzeit und nicht zuletzt durch das Betrachten des Algorithmus als akademisches Beispiel für kreative Laufzeitverbesserungen begründet werden.

## 2 $F$ -schwere/-leichte Kanten

Fehlt:  $F$ -leichte/-schwere Kanten als Gütemaß

$F$ -schwere und -leichte Kanten sind ein wesentlicher Bestandteil des MST-Algorithmus. Mittels der Identifizierung von Kanten in einem Graphen  $G$  als  $F$ -schwer hinsichtlich eines Waldes  $F$  in  $G$  kann bereits entschieden werden, dass diese Kante nicht im MST von  $G$  enthalten ist. Der **umkehrschluss** für  $F$ -leichte Kanten gilt jedoch nicht, wie wir sehen werden.

Beweis fehlt

### 2.1 Definition

Wir betrachten neben der Gewichtungsfunktion  $w$  nun die Funktion  $w_F$ , mit

$$w_F(\{u, v\}) = \begin{cases} \infty, & P(\{u, v\}) = \emptyset \\ \max\{w(P_e(\{u, v\}))\}, & \text{sonst} \end{cases}$$

, wobei  $w(P_e(\{u, v\}))$  bedeutet, dass  $w$  auf alle Kanten des Pfades angewandt wurde.

$w_F$ , gibt also das Kantengewicht der Kante mit maximalen Gewicht auf dem Pfad von  $u$  nach  $v$  in  $F$  aus. Sollte dieser Pfad nicht existieren, so nehmen wir an, dass diese Kante unendlich schwer ist.

Ist das Gewicht einer Kante  $w(e)$  echt größer als das maximale Gewicht auf dem Pfad  $P_e(e)$  in  $F$ , bzw.  $w(e) > w_F(e)$ , so bezeichnen wir sie als  $F$ -schwer. Sonst ist sie  $F$ -leicht.

2.1.1 Beweis e  $F$ -schwer  $\rightarrow$  e nicht im MST/MSF

2.1.2 e  $F$ -leicht  $\nrightarrow$  e in MST/MSF

### 2.2 Verifikation durch $F$ -schwere/-leichte Kanten

Es liegt nahe, dass  $F$  kein MSF in  $G$  ist, wenn eine Kante  $\{u, v\}$  existiert dessen gewicht echt kleiner als das des maximalen Gewichts auf dem Pfad  $P_e(\{u, v\})$  in  $F$  ist. Würde man annehmen, dass  $F$  ein MSF wäre, so könnte man  $E_F$  um die Kante  $\{u, v\}$  erweitern und den dadurch entstandenen Zyklus mittels entfernen der Kante maximalen Gewichts auf dem Pfad  $P_e(\{u, v\})$  lösen. Dadurch hätte man die Zusammenhängende Komponente als solche gewahrt und eine schwerere durch eine leichtere Kante substituiert. Folglich hätte man einen leichteren Wald und  $F$  wäre damit kein MSF.

Diese Erkenntnis reicht bereits aus um einen Verifikationsalgorithmus zum MST und MSF Problem zu konstruieren, der lediglich über alle Kanten iteriert und schaut ob die einzigen  $F$ -leichten Kanten die aus  $F$  selbst sind. zeige  $O(m)$

### 2.1.1 2.3 Informationsgewinn durch $F$ -schwere Kanten

Wir haben bereits erwähnt, dass  $F$ -schwere Kanten nicht in einem MSF, bzw. MST enthalten sein können. Dies lässt sich durch folgenden Beweis verdeutlichen.

Wir können also sämtliche  $F$ -schwere Kanten für das Erfassen des MST ignorieren, bzw. sogar aus  $G$  eliminieren.

Diese Erkenntnis nimmt sich auch der MST-Algorithmus zum Nutzen. In gewisser Hinsicht werden wir den umgekehrten Ansatz von Kruskal, welcher Kanten nach gewichten aufsteigend sortiert und immer nur  $F$ -leichte Kanten hinzunimmt, durch das Erfassen von  $F$ -schweren Kanten über Stichproben von Wäldern in  $G$  verfolgen.

### 3 Reduzieren des Graphen

siehe abstract

Wir werden einen rekursiven Algorithmus konstruieren, der genau dann aufhört, wenn nur noch ein Knoten in Graphen übrig bleibt. Folglich müssen wir bei jedem rekursiven Aufruf unseren Graphen reduzieren. Wir werden in diesem Teil betrachten, wie wir durch Verwendung von Borůvka-Phasen die Knoten des Graphen reduzieren können und anschließend durch randomisierte Stichproben zuzüglich eine Methode kennen lernen um die Anzahl von Kanten zu reduzieren. Letzteres ermöglicht uns erst eine erwartete lineare Laufzeit, da die Rekursionstiefe durch das alleinige Reduzieren von Knoten für unseren Anspruch zu groß ist.

#### 3.1 Borůvka-Phasen

Borůvka Phasen Beruhen auf der Erkenntnis, dass in einem beliebigen ungerichteten Graphen  $G$  für jeden Knoten  $v \in V$  die inzidente Kante mit minimaler Gewichtung  $e_{v,min} := \{v, u\}, u \in adj(v) : \nexists e' = \{v, u'\}, u' \in adj(v) : w(e') < w(e)$  im MST von  $G$  enthalten ist. (Beweis) Ferner bedeutet das, dass wir durch die für die Kontraktion markierten Kanten  $E_{min}$  nach einer Borůvka-Phase einen Wald  $F$  in  $G$ , mit Kanten aus einem MST von  $G$ . Dies wird für uns insbesondere dann interessant, wenn wir rekursiv einen Wald aufbauen möchten.

Darauf aufbauend betrachten wir nun den Ablauf einer Borůvka-Phasen:

1. Markiere inzidente Kanten  $E_{min}$  mit minimaler Gewichtung
2. Bestimme die verbundenen Komponenten in  $G' = (V, E_{min})$
3. Ersetze jede verbundene Komponente durch einen sie repräsentierenden Knoten in  $G'$  und erhalte den Graphen  $G''$
4. Entferne alle Selbstschleifen in  $G''$

Wir stellen fest, dass eine Borůvka-Phase die Menge an Knoten in  $G$  auf maximal die Hälfte reduziert, nämlich genau dann, wenn die markierten Kanten ein perfektes Matching beim Bestimmen der verbundenen Komponenten induzieren.

#### 3.2 Randomisierte Stichproben

Wir erfassen randomisierte Stichproben, indem wir für einen Graphen  $G$ , den Graphen  $G(p)$  konstruieren, in dem jede Kante je mit Wahrscheinlichkeit  $p \in \mathbb{R}$  enthalten ist. Der Vorgang der Entscheidung, ob eine Kante mit Wahrscheinlichkeit  $p$  in  $G(p)$  aufgenommen wird, bezeichnen wir im Folgenden als Zufallsexperiment. Ferner gilt  $V_{G(p)} = V_G$ .

Uns sollte bewusst sein, dass im Erwartungswert  $|E_{G(p)}| = m/p$  Kanten in  $G(p)$  enthalten sein sollten. Dies ist für uns im Kontext des Reduzierens von Kanten zielführend. Ferner ist die Konstruktion von  $G(p)$  nicht komplex.

### 3.2.1 Güte einer randomisierten Stichprobe

Wir haben festgestellt, dass randomisierte Stichproben uns eine Möglichkeit bieten Kanten elegant zu reduzieren. Jedoch haben wir im Teil zu  $F$ -leichten/-schweren Kanten gelernt, dass manche Kanten nicht im MST/MSF von  $G$  vorkommen und es wünschenswert ist möglichst wenige  $F$ -leichte Kanten in  $G$  hinsichtlich einer Approximation  $F$  des MST/MSF zu haben.

Im Folgenden werden wir zeigen, dass die Anzahl von  $F_{MSF}$ -leichten Kanten in  $G$  bezüglich des MSF  $F_{MSF}$  von unserer Stichprobe  $G(p)$  im Erwartungswert nach oben durch  $n/p$  beschränkt ist. Diese Annahme folgt aus dem Lemma 10.19 aus dem Buch. zitieren

Um dies zu beweisen werden wir zeigen, dass die Annahme vom Lemma 10.19 des Buches, nämlich dass die Zahl  $F_{MSF}$ -leichter Kanten in  $G$  durch eine Zufallsvariable mit negativer Binomialverteilung und den Parametern  $n, p$  stochastisch dominiert wird.

Zunächst möchten wir festlegen, dass im Folgenden alle Kanten aus  $G$  nach ihrer Gewichtung aufsteigend sortiert betrachtet werden. Dies wird für uns insbesondere dann handlich sein, wenn wir über die zu betrachtende Kante wissen möchten, ob sie  $F$ -leicht ist.

Nun da wir uns auf eine Iteration der Kanten festgelegt haben möchten wir  $G(p = 0, 5)$  wie üblich durch das Hinzunehmen der betrachteten Kante mit der Wahrscheinlichkeit  $p$  konstruieren. Während wir  $G(p)$  konstruieren können wir auch simultan, bzw. „online“ den MSF  $F_{MSF}$  von  $G(p)$  konstruieren. Dies ist beispielsweise durch einen Ansatz wie den von Kruskal möglich, bei dem wir genau dann eine Kante  $\{u, v\}$  in  $F_{MSF}$  aufnehmen, wenn  $\{u, v\}$  in verschiedenen verbundenen Komponenten in  $F_{MSF}$  liegen, wobei wir  $F_{MSF}$  mit allen Knoten aus  $G$  und einer leeren Kantenmenge initialisieren. Für uns ist dabei interessant, dass dies gleichbedeutend damit ist, dass die Kante  $\{u, v\}$  aus  $G$  genau dann  $F_{MSF}$ -leicht ist, wenn  $u, v$  in verschiedenen verbundenen Komponenten in  $F_{MSF}$  liegen. Die Korrektheit von  $F_{MSF}$  folgt aus der Sortierung der Kanten. als MSF von  $G(p)$

Wir können bereits folgende Eigenschaften zur Konsistenz unseres Verfahrens beobachten: (i) Ob die zu betrachtende Kante  $F_{MSF}$ -leicht oder -schwer ist, ist alleinig von den vorhergehenden Zufallsexperimenten abhängig, (ii) es werden keine Kanten aus  $F_{MSF}$  entfernt, (iii) eine Kante ist genau dann nach dem  $i$ -ten Zufallsexperiment  $F_{MSF}$ -leicht, wenn sie auch vor dem  $i$ -ten Zufallsexperiment  $F_{MSF}$ -leicht war.

Definieren wir nun den Begriff von Phasen in unserem Zufallsexperiment um auf eine Zufallsvariable zu schließen. Die  $k$ -te Phase unseres Verfahrens beginne, sobald  $|E_{F_{MSF}}| = k - 1$  Kanten vorhanden sind und ende bei  $|E_{F_{MSF}}| = k$ . Wir befassen uns in einer Phase also unter anderem mit der Anzahl von Zufallsexperimenten bis eine Kante hinzugenommen wird. Insbesondere ist die hinzugenommene Kante per Definition unseres Verfahrens  $F_{MSF}$ -leicht. Erinnern wir uns nun an unsere Annahme, so erfassen wir, dass  $F_{MSF}$ -schwere Kanten für unsere Zufallsvariable irrelevant sein sollten. Aus diesem Grund ignorieren wir alle  $F_{MST}$ -schweren Kanten und befassen uns in einer Phase nun nur noch mit  $F_{MST}$ -leichten Kanten und der Anzahl von Zufallsexperimenten, bis eine  $F_{MST}$ -leichte Kante zum ersten mal hinzugenommen wurde.

Nehmen wir nun an, dass nach Ablauf unseres Verfahrens wir  $F_{MST}$  mit  $|E_{F_{MST}}| = s$  erhalten. Es sollte ersichtlich sein, dass bis zum Ende von der  $s$ -ten Phase im Erwartungswert pro Phase  $1/p$   $F_{MSF}$ -leichte Kanten betrachtet wurden. Dies entspricht der geometrischen Verteilung mit Parameter  $p$ . Da wir auch davon ausgehen könnten, dass jede  $F_{MST}$ -leichte Kante sofort genommen wurde müssen wir noch solange weitere Zufallsexperimente für Pseudokanten durchführen, bis zusätzliche  $c = n - s$  Kanten hinzugenommen wären. Die Zufallsvariable, die die Anzahl von Zufallsexperimenten beschreibt sei  $X_{zexp}$ . Da  $s$

Aus (i), (ii), (iii) folgt insbesondere um eine Zufallsvariable zu erhalten die unser Verfahren stochastisch dominiert

zu Beginn erwähnen

maximal gleich  $n - 1$  sein kann, wird die Anzahl von  $F_{MST}$ -leichten Kanten durch  $X_{zexp}$  stochastisch dominiert. Die maximale erwartete Anzahl von  $F_{MST}$ -leichten Kanten in  $G$  ist folglich durch den Erwartungswert von  $X_{zexp}$  nach oben beschränkt.

$X_{zexp}$  entspricht der negativen Binomialverteilung, da wir Eine Anzahl von  $n$  Erfolgen je mit Wahrscheinlichkeit  $p$  erreichen möchten. Folglich ist die erwartete Anzahl von  $F_{MST}$ -leichten Kanten in  $G$  durch  $Pr[X_{zexp}] = n/p$  nach oben beschränkt.

## 4 Der MST-Algorithmus

---

Nun da wir Verfahren zum reduzieren von Knoten und Kanten des Graphen  $G$  kennen gelernt haben können wir anfangen einen Algorithmus zu konstruieren. Die Ausgabe unseres Algorithmus sei ein Wald  $F$  in  $G$ .

Die erste Entscheidung die wir treffen müssen, <sup>ist</sup> ob wir zuerst Knoten oder Kanten reduzieren. Da Borůvka-Pahsen nur Kanten markieren, die im MST/MSF enthalten sind fangen wir mit Borůvka-Pahsen an und erhalten den Graphen  $G_1$ . Dadurch verringern wir zunächst noch deterministisch die Anzahl  $F$ -leichter Kanten in  $G$ . Anschließend konstruieren wir  $G_2 = G_1(p)$ , wir wissen also, dass  $G$   $\frac{n/2}{p}$   $F$ -leichte Kanten nach diesen Verfahren enthält.

**zweite Rekursion: F-schwere kanten minimiert, da nicht verwendet**

**Data :** Graph  $G$

**Result :** Approximation eines MST/ MSF in  $G$

3 Borůvka-Phasen werden auf  $G$  angewandt.

- 1:  $G_1, C \leftarrow$  Dabei wird der resultierende Graph und die zur Kokatenierung markierten Kanten zurück gegeben.
- 2:  $G_2 \leftarrow G_1(p = 0, 5)$
- 3:  $F_2 \leftarrow MST(G_2)$
- 4:  $G_3 \leftarrow (V_{G_1}, E_{G_1} - E_{F_2-heavy})$
- 5:  $F_3 \leftarrow MST(G_3)$
- 6: **return**  $C \cup F_3$

### 4.1 Laufzeit

$T(n, m)$  sei die erwartete Laufzeit des MST-Algorithmus für einen Graphen  $G$ . Zeile 1 benutzt 3 Borůvka-Phasen und läuft damit in  $O(n+m)$ .  $G_2 = G_1(p)$  aus Zeile 2 kann ebenfalls in  $O(m+n)$  berechnet werden.  $G_1$  hat nur noch  $n/2^3 = n/8$  Knoten und für  $G_2$  hat damit  $|V_{G_2}| = n/8$  Knoten und im Erwartungswert  $|E_{G_2}| = m/2$  Kanten. Folglich benötigt die Berechnung von  $F_3$  die erwartete Laufzeit  $T(n/8, m/2)$ . Die

**Berechnung**  $F_2$ -schwerer Kanten und die Konstruktion von  $G_3$  mittels derer benötigt ebenfalls  $O(m+n)$ . Nach dem **Lemma 10.19** <sup>zitieren</sup> ist die Anzahl von  $F$ -leichten Kanten in  $G$  durch  $n/p = 2n$ , mit  $p = 0, 5$  im Erwartungswert nach oben beschränkt. Da  $G_1$  aber nur  $n/8$  Knoten hat, können wir die Anzahl an Kanten in  $G_3$  durch  $2 * n/8 = n/4$  im Erwartungswert abschätzen. Damit beläuft sich die erwartete Laufzeit der Berechnung von  $F_3$  auf  $T(n/2, n/4)$ . Die Vereinigung von  $C$  und  $F$  in Zeile 6 benötigt  $O(n)$ .

Aus den Laufzeiten der Teilschritte folgt  $T(n, m) \leq T(n/8, m/2) + T(n/8, n/4) + c(n + m)$ . Nach der Literatur kann  $T(n, m)$  durch  $2c(n + m)$  abgeschätzt werden. Ein Beweis, dass  $T(n, m)$  tatsächlich in  $O(n + m)$  enthalten ist befindet sich im Anhang.

$$\begin{aligned}
T(n, m) &\leq T(n/8, m/2) + T(n/8, n/4) + c(n + m) \\
&\leq (T(n/8^2, m/2^2) + T(n/8^2, \frac{m/2}{4}) + c(n/8 + m/2)) \\
&\quad + (T(n/8^2, \frac{n/4}{2}) + T(n/8^2, n/4^2) + c(n/8 + n/4)) \\
&\quad + c(n + m) \\
&= (T(n/8^2, m/2^2) + T(n/8^2, m/2^3) + c(n/8 + m/2)) \\
&\quad + (T(n/8^2, n/2^3) + T(n/8^2, n/2^4) + c(n/8 + n/2^2)) \\
&\quad + c(n + m) \\
&\leq (T(n/8^3, m/2^3) + T(n/8^3, \frac{m/2^2}{4}) + c(n/8^2 + m/2^2)) \\
&\quad + (T(n/8^3, \frac{m/2}{2}) + T(n/8^3, \frac{m/2}{4^2}) + c(n/8^2 + \frac{m/2}{4})) \\
&\quad + (T(n/8^3, \frac{n/4}{2^2}) + T(n/8^3, \frac{n/4}{4}) + c(n/8 + \frac{n/4}{2})) \\
&\quad + (T(n/8^3, \frac{n/4^2}{2}) + T(n/8^3, n/4^3) + c(n/8 + n/4^2)) \\
&\quad + c(n + m) \\
&= (T(n/8^3, m/2^3) + T(n/8^3, m/2^4) + c(n/8^2 + m/2^2)) \\
&\quad + (T(n/8^3, m/2^4) + T(n/8^3, m/2^5) + c(n/8^2 + m/2^3)) \\
&\quad + (T(n/8^3, n/2^4) + T(n/8^3, n/2^5) + c(n/8^2 + n/2^3)) \\
&\quad + (T(n/8^3, n/2^4) + T(n/8^3, n/2^6) + c(n/8^2 + n/4^2)) \\
&\quad + c(n + m) \\
&\leq \dots
\end{aligned}$$

$$\begin{aligned}
T(n, m) &\leq c \left( \sum_{i=1}^{k_1} 2 * (n/8^i) + m/2^i + m/2^{i+1} \right. \\
&\quad \left. + \sum_{i=1}^{k_2} 2 * (n/8^i) + n/2^{2+i} + n/2^{2+i+1} \right. \\
&\quad \left. + m + n \right) \\
&= c \left( 2n \sum_{i=1}^{k_1} 1/8^i + m \left( \sum_{i=1}^{k_1} 1/2^i + \sum_{i=1}^{k_1} 1/2^{i+1} \right) \right. \\
&\quad \left. 2n \sum_{i=1}^{k_2} 1/8^i + n \left( \sum_{i=1}^{k_2} 1/2^{2+i} + \sum_{i=1}^{k_2} 1/2^{2+i+1} \right) \right. \\
&\quad \left. + m + n \right) \\
&\leq c \left( 4n \sum_{i=0}^{\infty} 1/8^i \right. \\
&\quad \left. + m \left( \sum_{i=0}^{\infty} 1/2^i + \sum_{i=0}^{\infty} 1/2^{i+1} \right) \right. \\
&\quad \left. + n/2 \left( \sum_{i=0}^{\infty} 1/2^i + \sum_{i=1}^{\infty} 1/2^i \right) \right. \\
&\quad \left. + m + n \right) \\
&= c \left( 4n(8/7) \right. \\
&\quad \left. + m(2 + 2) \right. \\
&\quad \left. + n/2(2 + 2) \right. \\
&\quad \left. + m + n \right) \\
&= c(n(3 + 32/7) + 5m) \in O(m + n)
\end{aligned}$$