Der linearzeit MST-Algorithmus

Maximilian Springenberg

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 $m = |E|$. 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 $w : E \to \mathbb{R}$.

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 Warum ein nichtdeterministischer Anzatz von Vorteilen sein kann

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 aproximiert 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 β 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

3 Reduzieren des Graphen

3.1 Borůvka-Algorithmus

Wir werden uns zum Reduzieren der Knoten einen Teil des Algorithmus von Borůvka, der Borůvka-Phase, zur Hilfe nehmen. In der finalen Laufzeitanalyse werden wir zeigen, dass Borůvkas Phasen kombiniert mit randomisierten Stichproben der Kanten zu einem Algorithmus mit erwarteter Linearzeit führt.

3.1.1 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.
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 insebesondere dann interessant, wenn wir rekursiv einen Wald aufabauen möchten.

Darauf aufbauent 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.

3.2 Randomisierte Stichproben

4 der MST-Algorithmus