Datenstrukturen, Algorithmen und Programmierung 2

Amin Coja-Oghlan

June 20, 2023

Lehrstuhl Informatik 2
Fakultät für Informatik

Minimal spannende Bäume

Worum geht es?

- jeder zusammenhängende Graph besitzt einen spannenden Baum
- in einem gewichteten Graphen sind wir interessiert an einem minimalen spannenden Baum
- dazu lernen wir einen *Greedy-Algorithmus* kennen

Minimal spannende Bäume

Definition

- ein gewichteter Graph ist ein Graph $G = (V, E)$ zusammen mit einer Funktion $c : E \longrightarrow \mathbb{Q}_{\geq 0}$.
- das Gewicht einer Kantenmenge $F \subseteq E$ ist

$$\sum_{e \in F} c(e)$$

- ein minimal spannender Baum von G ist ein spannender Baum $T = (V, E_T)$ von G mit

$$c(E_T) = \min \left\{ c(E_B) : B = (V, E_B) \text{ ist spannender Baum von } G \right\}$$

Minimal spannende Bäume

Bemerkung

- für algorithmische Zwecke nehmen wir stets an, daß die Gewichtsfunktion c rationale Werte annimmt
- diese rationalen Werte werden dann exakt eingegeben, d.h. als Paare ganzer Zahlen

$$(\text{Zähler},\text{Nenner})$$

- äquivalent können wir annehmen, daß alle Gewichte ganzzahlig sind

Minimal spannende Bäume

Algorithmus Kruskal

1. prüfe, ob G zusammenhängend ist; wenn nicht, gib eine Fehlermeldung aus
2. sortiere die Kanten von G aufsteigend nach Gewichten:

$$c(e_1) \leq c(e_2) \leq \cdots \leq c(e_m)$$

3. initialisiere $T = (V, \emptyset)$
4. für $i = 1, \ldots, m$
5.   wenn e_i verschiedenene Komponenten von T verbindet
6.   füge e_i zu T hinzu
7. gib T aus

Minimal spannende Bäume

Bemerkungen

- Schritt 1 kann mit Tiefensuche in Zeit $O(|E|+|V|)$ durchgeführt werden
- für Schritt 2 verwenden wir `Heapsort`; Laufzeit $O(|E|\log|E|)$
- für Schritte 5 und 6 führen wir Buch über die Komponenten von T
- dabei führen wir die Komponentengrößen mit
- jedesmal, wenn eine Kante eingefügt wird, wird die *kleinere* Komponente zur größeren hinzugefügt

Minimal spannende Bäume

Bemerkungen

- der Kruskal-Algorithmus ist ein *Greedy-Algorithmus*
- wir probieren immer die günstigste Kante
- wir werden sehen, daß diese Strategie im MST-Problem zum Erfolg führt
- im allgemeinen sind Greedy-Algorihmen aber *nicht* optimal!

Minimal spannende Bäume

Satz

Sei G, c ein zusammenhängender gewichteter Graph (mit rationalen Gewichten).

- Kruskal hat Laufzeit $O(|E| \log |E|)$
- der Algorithmus gibt einen minimal spannenden Baum aus

Minimal spannende Bäume

Beweis: Laufzeit

- zur Laufzeit ist nur zu klären, wie Schritte 5 und 6 implementiert werden
- da jeweils die kleinere Komponente der größeren hinzugefügt wird, verdoppelt sich die Komponentengröße aus Sicht der kleineren Komponente mindestens
- jeder Knoten wird also höchstens $O(\log|V|)$ mal einer anderen Komponente zugeordnet
- die Laufzeit ist also $O(|V|\log|V|) = O(|E|\log|E|)$

Minimal spannende Bäume

Beweis: Korrektheit

- weil Kruskal nur dann eine Kante hinzufügt, wenn sie verschiedenene Komponenten von T verbindet, ist die Ausgabe kreisfrei
- ferner ist der Ausgabegraph zusammenhängend
- denn angenommen nicht
- dann gäbe es zwei Komponenten K_1, K_2
- weil G zusammenhängend ist, gibt es eine Kante in G, die K_1, K_2 verbindet
- Kruskal hätte also die erste solche Kante einfügen müssen, Widerspruch

Minimal spannende Bäume

Beweis: Korrektheit (fortgesetzt)

- zu zeigen bleibt, daß die Ausgabe minimales Gewicht hat
- angenommen nicht: bezeichne die in T enthaltenen Kanten mit

$$\begin{aligned} & e_{i_1}, \ldots, e_{i_{n-1}} \qquad (n = |V|) \\ & i_1 \leq \cdots \leq i_{n-1} \end{aligned}$$

- sei $0 \leq k < n - 1$ der maximale Index, so daß ein minimal spannender Baum

$$e_{i_1}, \ldots, e_{i_k}$$

enthält; sei B so ein minimal spannender Baum

Minimal spannende Bäume

Beweis: Korrektheit (fortgesetzt)

- dann enthält B die Kante $f = e_{i_{k+1}}$ nicht
- folglich enthält $(V, E(B) \cup \{f\})$ einen Kreis C
- dieser enthält eine Kante $f' \in E(B) \setminus E(T)$
- $B' = (V, (E(B) \cup \{f\}) \setminus \{f'\})$ ist ein spannender Baum
- nach Wahl von k gilt $c(B') > c(B)$
- daraus folgt, daß $c(f') < c(f)$
- also hätte Kruskal f' anstelle von f zu T hinzufügen müssen
- Widerspruch

Minimal spannende Bäume

Der Prim-Algorithmus

- ein alternativer greedy-Algorithmus zur Berechnung minimaler Spannbäume
- ausgehend von einem Startknoten s “züchtet” der Algorithmus einen spannenden Baum
- in jedem Schritt wird der Knoten hinzugefügt, der dem bereits konstruierten Baum mit kleinstem Gewicht hinzugefügt werden kann
- effizient implementierbar mit Hilfe von priority queues

Minimal spannende Bäume

Algorithmus Prim

1. Für alle $u \in V(G)$ setze $c(u) = \infty$, $p(u) = \emptyset$.
2. Setze $c(s) = 0$ und $Q = V(G)$.
3. Solange $Q \neq \emptyset$
4.   finde $v \in Q$ mit minimalem $c(v)$ und entferne v aus Q
5.   für alle $u \in Q \cap \partial_G v$
6.   falls $c(\{u, v\}) < c(u)$, setze $c(u) = c(\{u, v\})$ und $p(u) = v$
7. Gib die Kantenmenge $\{\{v, p(v)\} : p(v) \neq \emptyset\}$ aus.

Minimal spannende Bäume

Satz

Angenommen G, c ist ein zusammenhängender gewichteter Graph. Dann gibt $\texttt{Prim}(G, c)$ die Kantenmenge eines minimalen Spannbaums von G aus.

Minimal spannende Bäume

Beweis

- in jeder Iteration wird ein Knoten aus Q entfernt
- daher erfolgen höchstens $|V(G)|$ Iterationen
- Schritt (6) stellt sicher, daß es keine zwei Knoten $x, y \in Q$ mit $p(x) = y$ gibt
- daher wächst

$$\mathcal{E} = \{\{v, p(v)\} : p(v) \neq \emptyset\}$$

in jeder Iteration um eine Kante, mit Ausnahme des ersten Durchlaufs

Minimal spannende Bäume

Beweis

- Behauptung: stets existiert ein minimaler Spannbaum, der $\mathcal{E}$ enthält
- *erste Iteration*: $Q = V(G) \setminus \{s\}$ und $\mathcal{E} = \emptyset$
- *Induktionsschritt*: sei T' ein MSB, der $\mathcal{E}'$ aus der vorherigen Iteration enthält
- enthalte $\mathcal{V}'$ die Knoten, die mit $\mathcal{E}'$ inzident sind
- enthalte $\mathcal{V} = \mathcal{V}' \cup \{v\}$ die Knoten, die mit $\mathcal{E}$ inzident sind
- sei $e = \{v, p(v)\}$ die zuletzt zu $\mathcal{E}$ hinzugefügte Kante
- angenommen es gäbe *keinen* MSB, der $\mathcal{E}$ enthält

Minimal spannende Bäume

Beweis

- weil $e \notin E(T')$, enthält $T' + \{e\}$ einen Kreis C
- auf C gibt es eine Kante $f \neq e$ mit

$$f \cap \mathcal{V}' \neq \emptyset \qquad \text{und} \qquad f \setminus \mathcal{V}' \neq \emptyset$$

- also gilt $c(f) \geq c(e)$ und $f \notin \mathcal{E}$
- folglich ist $T = T' - \{f\} + \{e\}$ ein MSB von G
- Widerspruch

Minimal spannende Bäume

Laufzeit

- die Menge Q kann mit Hilfe einer min priority queue implementiert werden.
- Schritt (2) fügt die Knoten in die min priority queue ein; alle bis auf den Startknoten haben dasselbe Gewicht.
- Schritt (4) entfernt jeweils das Minimum aus der queue
- Schritt (6) reduziert das Gewicht einiger Nachbarn von v
- Laufzeit $O(|E(G)|\log|V(G)|)$

Minimal spannende Bäume

Anwendung: metrisches TSP

- wir befassen uns mit einem Spezialfall des TSP-Problems
- sei G, w ein vollständiger gewichteter Graph
- wir nehmen an, daß die Gewichte die *Dreiecksungleichung* erfüllen, d.h.

$$w(\{x,z\}) \leq w(\{x,y\}) + w(\{y,z\}) \qquad \text{für alle } x,y,z \in V(G).$$

- man spricht vom *metrischen TSP*

Minimal spannende Bäume

Proposition

Die Länge einer optimalen metrischen TSP-Tour ist

- mindestens so hoch wie das Gewicht eines optimalen Spannbaums von G, w und
- nicht mehr als doppelt so hoch wie das Gewicht eines optimalen Spannbaums von G, w.

Minimal spannende Bäume

Zusammenfassung

- minimal spannende Bäume in gewichteten Graphen
- Kruskal-Algorithmus
- Prim-Algorithmus
- Greedy-Strategie