
. Joseph Bernhard Kruskal

Joseph Bernard Kruskal (* 1929 in New York City) veröffentlichte 1956 in einer Zeitschrift einen Algorithmus zum Finden der kürzesten Verbindung von einem Knoten zu allen Knoten eines Graphen (Single-source shortest paths problem). Er stützte sich dabei auf Forschungen von Vojtech Jarnik,
der bereits 1930 die Grundlagen für die Algorithmen von Prim und Kruskal legte. Dieser liefert eiinen minimal aufspannenden Baum. Das Prinzip, welches dem Vorgehen zu Grunde liegt, bezeichnet man als sogenanntes Greedy-Prinzip: Nach dem Motto *Nimm-was-Du-kriegen-kann st* suchen
Greedy-Algorithmen stehts nach der lokal optimalen Variante und leiten daraus das Ziel ab, auch global ein optimales Ergebnis zu erhalten.

**Vorgehen - Algorithmus von Kruskal:**

1. alle Kanten des Graphen werden nach ihrem Gewicht in einer Kantenliste sortiert
2. jeder Knoten wird als Baum aufgefasst, dabei ist jeder Knoten Sohn und Vater zugleich
3. die Kante mit dem geringsten Gewicht wird genommen (Greedy-Prinzip) und überprüft, ob die Knoten am Kantenende in unterschiedlichen Bäumen sind (kein Zyklus), ist dies der Fall, werden die Kanten vereinigt und die Kante aus der Kantenliste gelöscht, im anderen Fall wird die Kante nicht aufgenommen

Der Algorithmus von Prim hat als Ergebnis, vergleichbar mit dem Kruskal-Algorithmus, einen minimalen aufspannenden Baum eines bewerteten ungerichteten Graphen. Das Verfahren ist gegen über Kruskal effektiver und schneller.

**Vorgehen - Algorithmus von Prim:**

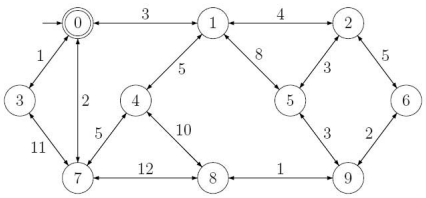
1. Ausgangspunkt für das Verfahren ist ein beliebiger Startknoten
2. alle Kanten zu Nachbarknoten werden in eine *Nachbarliste* eingefügt, man wählt eine Kante minimaler Länge aus der
*Nachbarliste*
und fügt diese Kante dem bereits initialisierten Spannbaum zu

3. von dort wird wieder der minimale Weg, basierend auf der ausgewählten Kante, zum nächsten Knoten gewählt, ist dieser Knoten bereits besucht worden, wird er nicht berücksichtigt

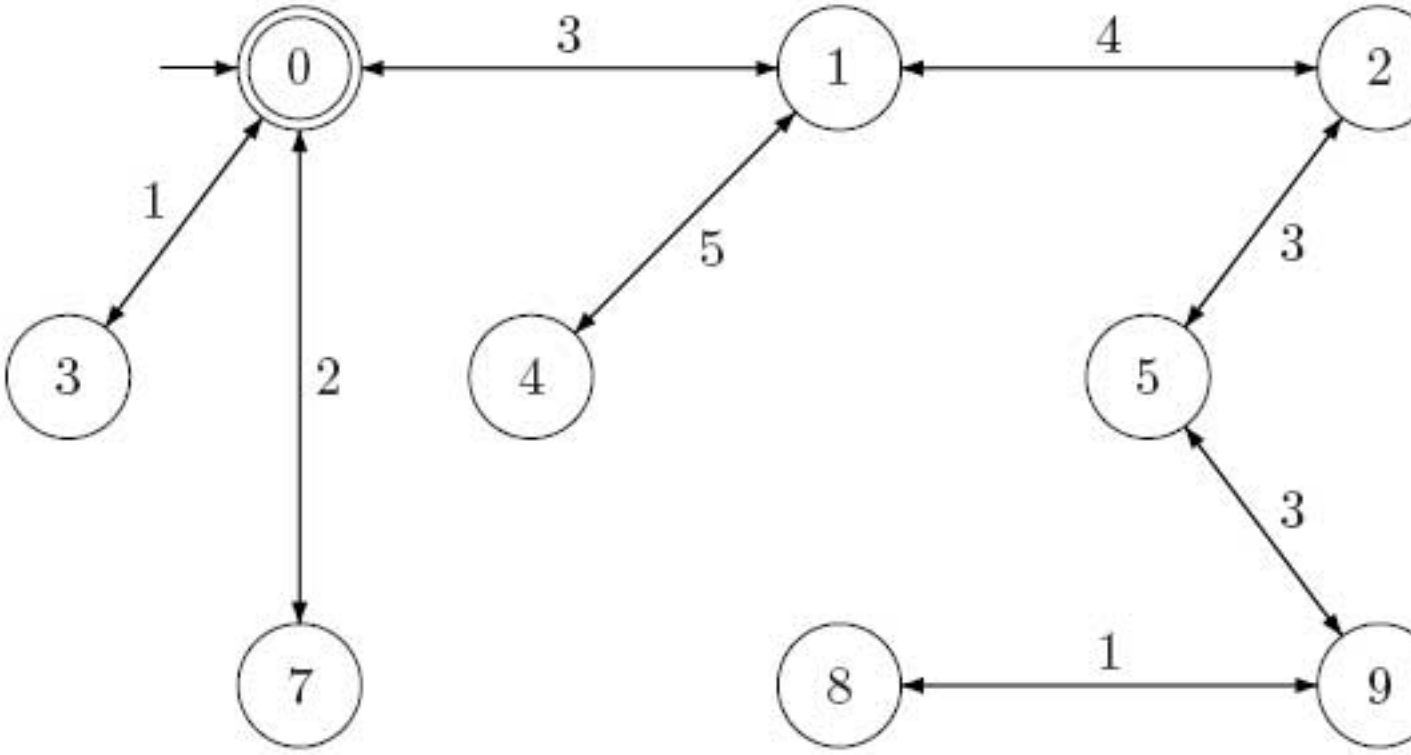
4. dieses Verfahren führt man durch, bis alle Knoten besucht wurden

Die Effizienz des Verfahrens basiert auf der Organisation der Nachbarliste. Um möglichst schnell eine minimale Kante in der Nachbarliste zu finden, verwendet man in der Implementierung eine *priority queue*. Im Regelfall wird diese über einen Heap organisiert, im nachfolgenden Beispiel
wird eine dagegen eine Queue verwendet, die sich nur bei der Methode *entfernen* von einer herkömmlichen Queue unterscheidet.

Beispielgraph




minimaler Spannbaum




Script zur Graphentheorie [unvollständig]