9.6 Spannbäume: der Algorithmus von Kruskal

LK

Eine andere Frage, dieselben Daten: Ein Anbieter will alle sechs Orte an ein Glasfasernetz anschließen und darf das Kabel nur entlang der vorhandenen Straßen verlegen. Wie viel Kabel braucht er mindestens?

Gesucht ist eine Auswahl von Kanten, die *alle* Knoten verbindet und möglichst *wenig* kostet. Kreise wären Verschwendung — jede Kante in einem Kreis ließe sich weglassen, ohne dass etwas zerfällt. Was übrig bleibt, ist nach Abschnitt 9.2.3 ein Baum.

Definition: Minimaler Spannbaum

Ein **Spannbaum** eines zusammenhängenden Graphen $G = (V, E)$ ist ein Teilgraph, der alle Knoten aus V enthält, zusammenhängend ist und keinen Kreis besitzt. Er hat damit stets genau $|V| - 1$ Kanten.

Ein **minimaler Spannbaum** ist ein Spannbaum, bei dem die Summe der Kantengewichte so klein wie möglich ist.

Der Algorithmus von **Joseph Kruskal** (1956) ist erfreulich einfach — und er ist *gierig*: Er trifft in jedem Schritt die lokal beste Entscheidung und nimmt sie nie zurück.

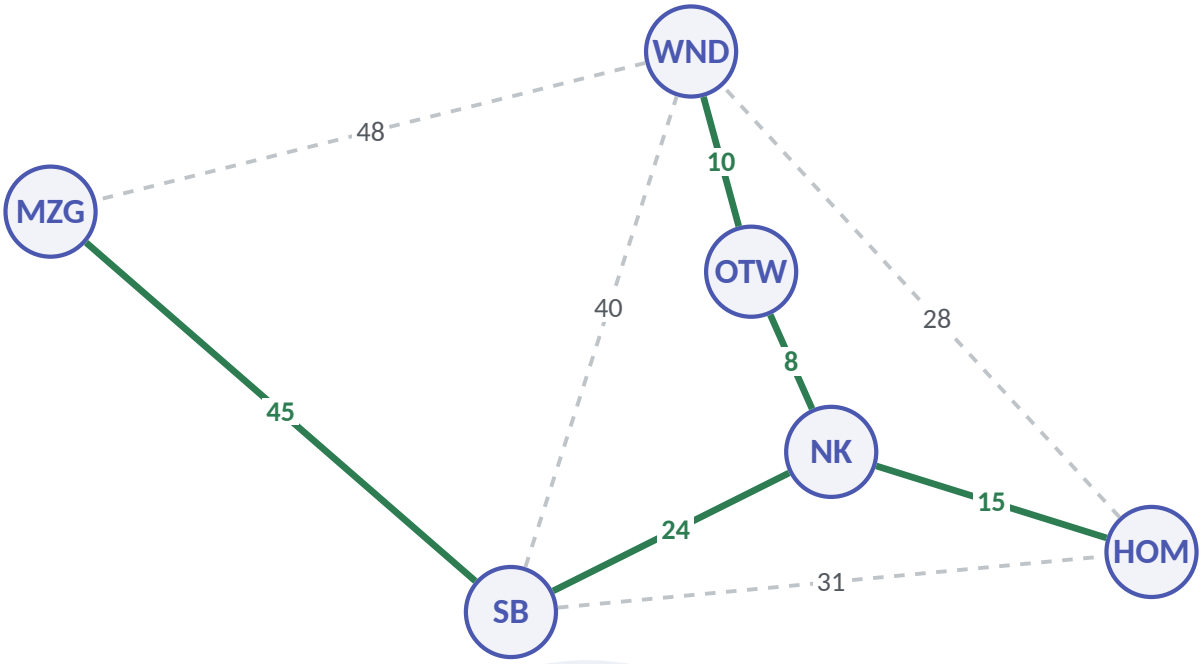
Definition: Algorithmus von Kruskal

1. Sortiere alle Kanten aufsteigend nach ihrem Gewicht.
2. Gehe die Liste von vorn durch. Nimm eine Kante in den Baum auf, **wenn sie keinen Kreis schließt** — wenn ihre beiden Endknoten also noch nicht im selben Teilstück hängen.
3. Fertig, sobald $|V| - 1$ Kanten gewählt sind.

Beispiel: Das Glasfaserkabel im Saarland

Kante	km	Entscheidung
NK – OTW	8	genommen
OTW – WND	10	genommen
NK – HOM	15	genommen
SB – NK	24	genommen
HOM – WND	28	**verworfen** — HOM und WND hängen schon über NK und OTW zusammen
SB – HOM	31	**verworfen** — SB und HOM hängen schon über NK zusammen
SB – WND	40	**verworfen**
SB – MZG	45	genommen — jetzt sind alle sechs Orte verbunden
WND – MZG	48	nicht mehr nötig

Fünf Kanten, zusammen $8 + 10 + 15 + 24 + 45 = 102$ km. Weniger geht nicht.



$8 + 10 + 15 + 24 + 45 \quad = \quad 102\,\text{km}$

Der minimale Spannbaum, durchgezogen. Die vier gestrichelten Kanten hätten jeweils einen Kreis geschlossen.

Die ganze Kunst steckt in der Frage „schließt diese Kante einen Kreis?“. Einfach, aber langsam ist es, jedem Knoten eine Gruppennummer zu geben und beim Verschmelzen zweier Gruppen alle Nummern umzuschreiben:

```
def kruskal(graph):
    kanten = sorted((laenge, a, b)
                    for a in graph for b, laenge in graph[a].items() if a < b)
    gruppe = {knoten: knoten for knoten in graph}
    baum = []
    for laenge, a, b in kanten:
        if gruppe[a] != gruppe[b]:              # verschiedene Gruppen: kein Kreis
            alt, neu = gruppe[b], gruppe[a]
            for knoten in gruppe:               # Gruppen verschmelzen
                if gruppe[knoten] == alt:
                    gruppe[knoten] = neu
            baum.append((a, b, laenge))
    return baum
```

```
>>> baum = kruskal(GRAPH)
>>> sum(laenge for _, _, laenge in baum)
102
```

Stolperstein

„Gierig" funktioniert selten. Dass es hier funktioniert, ist eine Besonderheit des Spannbaumproblems und muss bewiesen werden — bei den Sortierverfahren aus Kapitel 4 hätte eine solche Strategie kläglich versagt, und beim Rundreiseproblem im nächsten Abschnitt tut sie es auch. Verwechseln Sie „einfach" nicht mit „offensichtlich richtig".

Arbeitsauftrag 6

31. Führen Sie Kruskal auf dem Saarland-Graphen durch, ohne in das Beispiel zu schauen, und vergleichen Sie danach.
32. Begründen Sie, warum ein Spannbaum bei sechs Knoten immer genau fünf Kanten hat.
33. Verdoppeln Sie die Kante SB – MZG auf 90 km und bestimmen Sie den neuen minimalen Spannbaum. Welche Kante rückt nach, und um wie viel wird das Netz teurer?
34. Ein minimaler Spannbaum ist nicht immer eindeutig. Geben Sie einen kleinen Graphen an, der zwei verschiedene minimale Spannbäume besitzt.
35. Ist der kürzeste Weg von Merzig nach Ottweiler im minimalen Spannbaum enthalten? Prüfen Sie nach und erklären Sie das Ergebnis.
36. Das Verschmelzen der Gruppen im Programm läuft über alle Knoten. Schätzen Sie den Gesamtaufwand bei n Knoten und m Kanten ab.

Gymnasium Ottweiler