

Aufgabenblatt 4

Planung eines optimalen Telefonnetzes mit minimal aufspannenden Bäumen

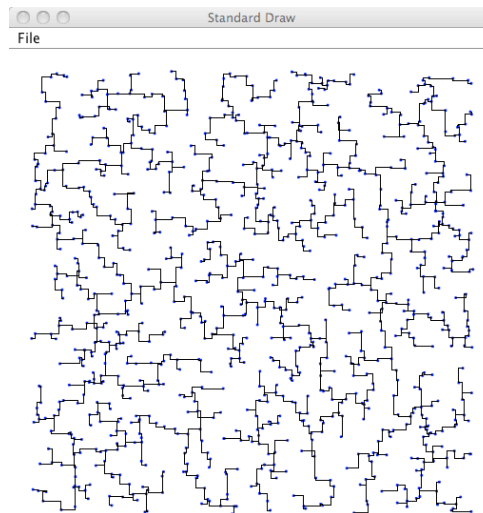


Abb. 1: Telefonnetz mit 1000 Knoten

In einer gitterförmig aufgebauten Stadt ist eine Menge von Telefonknoten mit ganzzahligen x-y-Koordinaten gegeben. Die Kosten für die Verbindung zweier Telefonknoten (x_1, y_1) und (x_2, y_2) wird mit Hilfe der sogenannten Manhattan-Distanz berechnet:

$$\text{dist}((x_1, y_1), (x_2, y_2)) = |x_1 - x_2| + |y_1 - y_2|$$

Die Abbildung 2 zeigt eine Stadt mit zwei blau gefärbten Telefonknoten. Die Knoten mit den Koordinaten (1,1) und (3,5) haben eine Manhattan-Distanz von $\text{dist} = |1-3| + |1-5| = 6$. Die Manhattan-Distanz drückt aus, dass Telefonleitungen nur längs von Straßen (horizontale und vertikale Linien) gelegt werden dürfen. Beispielsweise wäre die rote Linie der Länge 6 eine mögliche Verbindung zwischen den beiden Knoten.

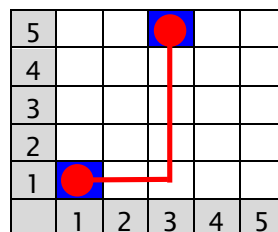


Abb. 2: Telefonnetz mit 2 Knoten

Zwei Knoten gelten als nicht direkt verbindbar, wenn ihre Manhattan-Distanz über einen Leitungsbegrenzungswert lbg liegt. Damit sind die Kosten cost für die Verbindung zweier Telefonknoten (x_1, y_1) und (x_2, y_2) wie folgt definiert:

$$\text{cost}(((x_1, y_1), (x_2, y_2))) = \begin{cases} \text{dist}((x_1, y_1), (x_2, y_2)), & \text{falls } \text{dist}((x_1, y_1), (x_2, y_2)) \leq lbg, \\ \infty, & \text{sonst} \end{cases}$$

Mit dem **Algorithmus von Kruskal** soll für eine Stadt mit einer gegebenen Menge von Telefonknoten ein optimales Telefonnetz, d.h. ein minimal aufspannender Baum, berechnet werden.

Beispielsweise ergibt sich für die Stadt mit 7 Knoten und $lbg = 7$ in Abbildung 3 einen minimal aufspannenden Baum mit den Gesamtkosten von $3+3+5+3+2+2=18$. (Die Lösung muss nicht eindeutig sein!)

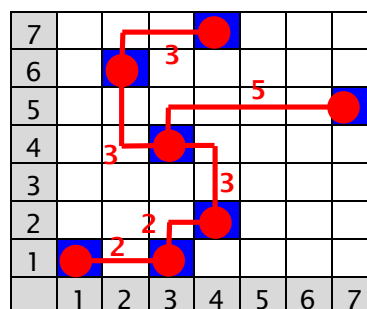


Abb. 3: Optimales Telefonnetz mit 7 Knoten und Gesamtkosten 18.

Lösen Sie folgende Teilaufgaben:

1. Implementieren Sie eine Klasse **UnionFind**. Die Beschreibung der Klasse finden Sie auf der Web-Seite in Javadoc. Testen Sie Ihre Klasse ausgiebig.
2. Realisieren Sie eine Klasse **TelNet** zur Verwaltung der Telefonknoten und Berechnung eines minimal aufspannenden Baums mit dem Algorithmus von Kruskal. Die Beschreibung der Klasse finden Sie auf der Web-Seite in Javadoc.

Um die UnionFind-Klasse verwenden zu können, sehen Sie für jeden Telefonknoten eine einfache Nummerierung vor: 0, 1, 2, ..., n-1. Die Zuordnung von Telefonknoten zu interner Nummerierung wird in einer Map abgespeichert.

Alle möglichen Telefonverbindungen speichern Sie in einer `java.util.PriorityQueue` ab.

Der Algorithmus benötigt sonst keine Informationen über den Graph. Eine Graphenklasse aus der Aufgabe 2 ist daher nicht notwendig.

3. Testen Sie Ihre Klasse mit den Beispieldaten aus Abbildung 3.
4. Generieren Sie $n = 1000$ zufällige Knoten in einem $xMax \times yMax$ großen Gitter mit $xMax = yMax = 1000$. Setzen Sie dabei $lbg = 100$. Berechnen Sie ein optimales Telefonnetz und animieren Sie das Netz, wie in Abb. 1 gezeigt. Sie können zum Zeichnen die aus Programmiertechnik 2 bekannte Klasse `StdDraw` verwenden.