Hochschule Konstanz
Fakultät Informatik

Algorithmen und Datenstrukturen
Angewandte Informatik
SS 2025
Prof. Dr. Oliver Bittel

Aufgabenblatt 4

Planung eines optimalen Telefonnetzes mit minimal aufspannenden Bäumen

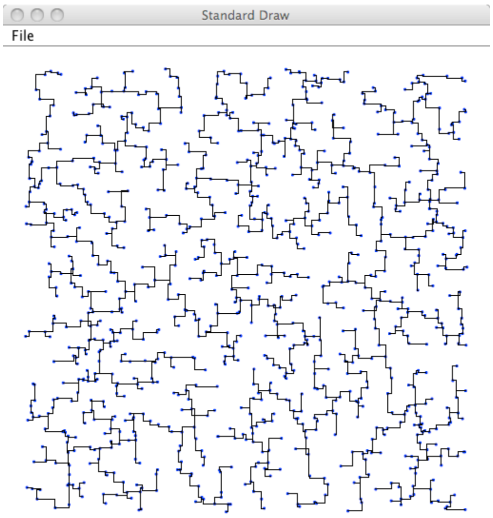

Abb. 1: Telefonnetz mit 1000 Knoten

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:

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

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 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.

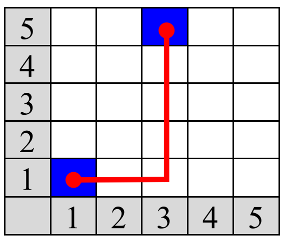

Abb. 2: Telefonnetz mit 2 Knoten

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:

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

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+4+3+3+2+2=17. (Die Lösung muss nicht eindeutig sein!)

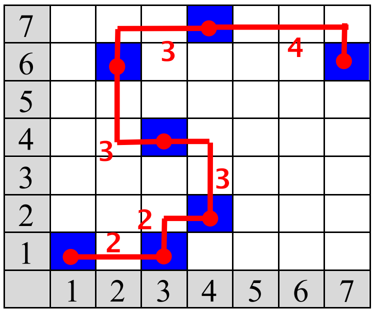

Abb. 3: Optimales Telefonnetz mit 7 Knoten und Gesamtkosten 17.

Lösen Sie folgende Teilaufgaben:

1. Implementieren Sie die generische Klasse **UnionFind**. Die Beschreibung der Klasse finden Sie auf der Web-Seite in Javadoc. Die Union-Find-Struktur ist in Kap. 14 beschrieben. Implementieren Sie find mit Pfadkompression und Union-By-Rank (Seite 14-15 bis 14-17). Beachten Sie außerdem den Hinweis aus Seite 14-19. Die Methode size() liefert aus Effizienzgründen direkt eine Instanzvariable size zurückliefert. 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.

   Alle möglichen Telefonverbindungen speichern Sie in einer java.util.PriorityQueue ab. Übergeben Sie beim Konstruktor einen geeigneten Comparator.

   Der Algorithmus benötigt sonst keine Informationen über den Graph. Die Graphenklasse aus der Aufgabe 2 bzw. 3 wird daher nicht benötigt.
3. Beachten Sie dass die Klassen **TelKnoten** und **TelVerbindung record-Klassen** sind und als Einzeller implementiert werden können.
4. Testen Sie Ihre Klasse mit den Beispieldaten aus Abbildung 3.
5. Generieren Sie n = 1000 zufällige Knoten in einem xMax*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.