
Grundlagen: Algorithmen und Datenstrukturen

Abgabetermin: In der jeweiligen Tutorübung

Hausaufgabe 1

Implementieren Sie in der Klasse `SSSP` den Algorithmus von Dijkstra zur Berechnung der kürzesten Wege von einem Knoten s zu allen anderen Knoten in dem ungerichteten Graphen G .

Verwenden Sie für Ihre Implementierung die auf der Übungswebseite bereitgestellten Klassen und verändern Sie für Ihre Implementierung *ausschließlich* die Klasse `SSSP` aber auf keinen Fall deren Interface `ISSSP`.

Achten Sie bei der Abgabe Ihrer Aufgabe darauf, dass Ihre Klassen den Rechnern der Rayhalle (rayhalle.informatik.tu-muenchen.de) mit der bereitgestellten Datei `main_dij` kompiliert werden kann. Anderenfalls kann eine Korrektur nicht garantiert werden. Achten Sie darauf, dass Ihr Quelltext ausreichend kommentiert ist.

Schicken Sie die Lösung per Email mit dem Betreff `[GAD] Gruppe <Gruppennummer>` an Ihren Tutor.

Hinweis: Verwenden Sie grundlegende Datenstrukturen, wie sie Java bereitstellt, und implementieren Sie diese *nicht* neu.

Hausaufgabe 2

Implementieren Sie in der Klasse `MST` den Algorithmus von Jarnik und Prim zur Berechnung eines Minimum Spanning Tree beginnend von einem Knoten s von einem ungerichteten Graphen G .

Der Algorithmus von Jarnik und Prim wählt zu einer Menge V' den Knoten der über eine Kante mit dem geringsten Gewicht erreichbar ist und fügt ihn der Menge V' hinzu. Gestartet wird der Algorithmus mit $V' = s$.

Verwenden Sie für Ihre Implementierung die auf der Übungswebseite bereitgestellten Klassen und verändern Sie für Ihre Implementierung *ausschließlich* die Klasse `MST` aber auf keinen Fall deren Interface `IMST`.

Achten Sie bei der Abgabe Ihrer Aufgabe darauf, dass Ihre Klassen den Rechnern der Rayhalle (rayhalle.informatik.tu-muenchen.de) mit der bereitgestellten Datei `main_mst` kompiliert werden kann. Anderenfalls kann eine Korrektur nicht garantiert werden. Achten Sie darauf, dass Ihr Quelltext ausreichend kommentiert ist.

Schicken Sie die Lösung per Email mit dem Betreff `[GAD] Gruppe <Gruppennummer>` an Ihren Tutor.

Hinweis: Verwenden Sie grundlegende Datenstrukturen, wie sie Java bereitstellt, und implementieren Sie diese *nicht* neu.

Aufgabe 1

Argumentieren Sie, warum es unklug ist, bei der Berechnung eines kürzesten Weges von s nach t in einem Graphen mit echt positiven Gewichten alle möglichen (einfachen) Wege aufzuzählen und deren Distanz zu berechnen.

Benutzen Sie dazu exemplarisch ein Gitter der Größe n . Die n^2 Knoten werden durch die Tupel (x, y) mit $x, y \in [n]$ identifiziert.

Die Kanten $((x, y), (x + 1, y))$ seien gerichtet während die Kanten $((x, y), (x, y + 1))$ bidirektional verlaufen. Berechnen Sie die Anzahl aller Pfade von $s = (1, 1)$ zu $t = (n, n)$.

Aufgabe 2

Geben Sie einen möglichst effizienten Algorithmus an, der die *Anzahl* der kürzesten Wege von einem Knoten s zu allen anderen Knoten in einem ungerichteten Graphen G mit echt positiven Gewichten berechnet.