Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Institut für Theoretische Informatik
Peter Widmayer
Thomas Tschager
Antonis Thomas

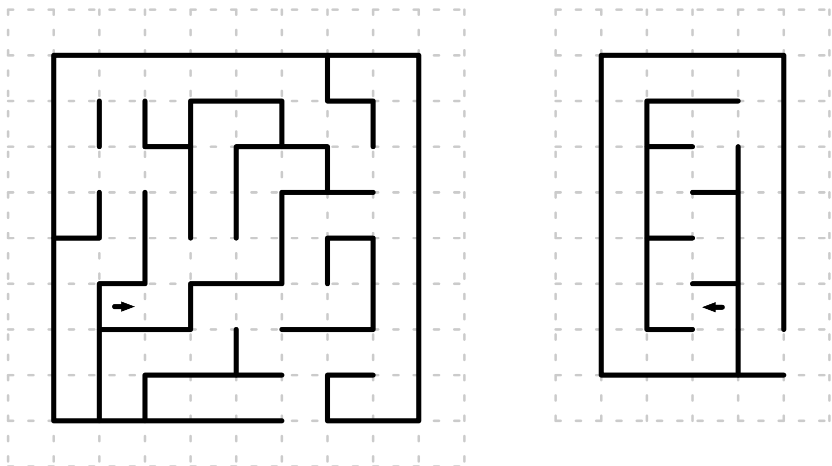
4. Mai 2016

Datenstrukturen & Algorithmen — Blatt 10 — FS 16

Aufgabe 10.1 *Pfadplanung in Labyrinthen.*

Gegeben ist ein Labyrinth als Zeichnung auf kariertem Papier wie in den Beispielen unten. An der Stelle, die mit einem Pfeil markiert ist, befindet sich ein Roboter, der in Pfeilrichtung blickt. Die Frage ist nun, wie schnell der Roboter dem Labyrinth entkommen kann. Der Roboter kann innerhalb von 3 Sekunden jeweils ein ganzes Feld in Blickrichtung "vorwärts" fahren. Es kostet ihn 2 Sekunden, nach einer Vorwärtsbewegung stehen zu bleiben. Nur im Stand kann der Roboter sich um 90 Grad drehen, was ihn 2 Sekunden kostet. Zwischen zwei aufeinander folgenden Vorwärtsbewegungen muss der Roboter nicht erst stehen bleiben (er könnte es zwar, es würde ihn aber mehr Zeit kosten).

In den beiden folgenden Beispielen benötigt der Roboter 113s und 79s, um zu entkommen.



a) Modellieren Sie das obige Problem als Kürzeste-Wege-Problem. Beschreiben Sie dazu, wie man das Labyrinth als Graphen darstellen kann, sodass die Länge des kürzesten Pfades im Graphen gleich der Zeit ist, die der Roboter braucht, um zu entkommen.

b) Nennen Sie einen möglichst effizienten Algorithmus zur Lösung des Problems.

c) Welche Laufzeit in Abhängigkeit von der Anzahl der Felder hat der genannte Algorithmus auf dem in a) konstruierten Graphen?

Aufgabe 10.2 *Varianten zu Kürzeste-Wege-Problemen.*

Seien $G = (V, E, w)$ ein gerichteter, gewichteter Graph mit positiven Kantengewichten und seien $s, t \in V$ zwei Knoten. Entwerfen Sie für die folgenden Varianten des Kürzeste-Wege-Problems einen möglichst effizienten Algorithmus. Geben Sie auch die Laufzeit an.

a) Sei $k \in \mathbb{N}$ beliebig. Unter allen möglichen Wegen von s nach t mit höchstens k Zwischenknoten (d.h., mit maximal $k+1$ Kanten) wird derjenige gesucht, der die kürzeste Länge besitzt.

b) Neben dem kürzesten Weg von s nach t wird zusätzlich der zweitkürzeste Weg gesucht.

c) Gesucht wird die Anzahl aller verschiedenen kürzesten Wege von s nach t.

Aufgabe 10.3 *Closeness centrality (**Programmieraufgabe**).*

In dieser Aufgabe berechnen wir die *closeness centrality* in ungerichteten Graphen $G = (V, E)$. Die closeness centrality misst, wie weit ein Knoten von allen anderen Knoten im Graphen entfernt ist. Für diese Aufgabe definieren wir die closeness centrality für einen Knoten v wie folgt:

$$C(v) = \sum_{u \in V} d(v, u),$$

wobei V die Menge der Knoten des Graphen ist und $d(v, u)$ die Länge eines kürzesten Pfades zwischen v und u bezeichnet. Existiert kein Pfad zwischen zwei Knoten u und v, so definieren wir $d(u, v) = 0$. Sie können annehmen, dass der Graph bis auf einige wenige isolierte Knoten zusammenhängend ist. Ein isolierter Knoten v hat also closeness centrality $C(v) = 0$. Je kleiner $C(v)$ ist (mit Ausnahme von $C(v) = 0$), desto "zentraler" ist ein Knoten im Graph.

(Hinweis: Üblicherweise wird die closeness centrality eines Knoten v als der Kehrwert von $C(v)$ definiert.)

Eingabe In dieser Aufgabe verwenden wir einen Graphen, der die wissenschaftlichen Kollaborationen des berühmten Mathematikers Paul Erdős beschreibt. Die Eingabe enthält die Beschreibung eines Graphen mit 511 Knoten. Jeder Knoten repräsentiert einen Koauthor von Erdős. Zwei Knoten sind durch eine Kante verbunden, wenn beide Autoren zusammen einen wissenschaftlichen Artikel verfasst haben. Mehr Informationen zu diesem Graph finden Sie unter `http://wwwp.oakland.edu/enp/thedata/`. Unsere Eingabe unterscheidet sich geringfügig von folgender Datei: `http://files.oakland.edu/users/grossman/enp/erdos1graph.html`.

Ausgabe Für jeden Knoten (in alphabetischer Reihenfolge, wie in der Eingabedatei) soll die closeness centrality ausgegeben werden.

Hinweise Wir stellen eine Code-Vorlage zur Verfügung, in der das Einlesen der Eingabe bereits implementiert wurde. Implementieren Sie den Algorithmus von Floyd und Warshall und berechnen Sie dann die closeness centrality. Für diese Aufgabe gibt es lediglich ein Testset, das auch vom Judge verwendet wird.

Beispiel Im folgenden werden die Namen der fünf Wissenschaftler mit der kleinsten closeness centrality sortiert aufgelistet. Die Namen entsprechen den Namen in der Eingabedatei.

Top 5:

```
GRAHAM, RONALD LEWIS
ALON, NOGA M.
FUREDI, ZOLTAN
SOS, VERA TURAN
BOLLOBAS, BELA
```

Abgabe: Am Mittwoch, den 11. Mai 2016 in Ihrer Übungsgruppe.