Dr. S. Lewandowski Universität Stuttgart, Fak. 5, FMI-FK, SS 2008

Wegeprobleme in Graphen **Übungsblatt 03**

Ausgabe: 05.06. Besprechung am 13.06

Ablauf der Übungen: Ausgabe der Übungen spätestens 8 Tage vor dem nächsten Übungstermin in der Vorlesung (pdf auf der Vorlesungsseite). Abgabe spätestens am Tag vor der Übung in der Vorlesung oder per E-Mail an `Lewandowski@fmi.uni-stuttgart.de`.

---

1. (mittel) **Fibonacci-Heaps:** Die Worst-Case-Laufzeiten der Decrease_Key-Funktion beträgt $O(n)$. Finden Sie Beispiele für Operationsreihenfolgen, bei denen tatsächlich für ein Decrease_Key-Aufruf dieser Worst-Case erreicht wird.

   Untersuchen Sie, ob solch eine Reihenfolge auch tatsächlich im Rahmen der Kürzeste-Wege-Berechnung mit Dijkstras Algorithmus auftreten kann.

   Überlegen Sie sich, wie die Datenstruktur für Fibonacci-Heaps (z.B. in Ada) aussehen kann. Wie groß muss das Array für die Liste der Bäume sein? Beachten Sie dabei, dass ein Baum mit Rang k zwar nur F_{k+2} Knoten haben kann, dass diese Knotenanzahl jedoch nicht ausreicht, um einen solchen zu erstellen. Z.B. existiert ein Baum mit Rang 2 mit 3 Knoten, zur Erstellung werden aber 4 Knoten gebraucht (von dem später durch ein Decrease_Key einer wieder abgetrennt wird).

2. (leicht–mittel) **Stimmt's?** Welche der folgenden Aussagen sind korrekt? Begründen Sie Ihre Antworten mittels Beweis oder Gegenbeispiel!

   - Haben die Kanten alle verschiedene Gewichte, so ist der Kürzeste-Wege-Baum eindeutig.
   - Erhöht (und verringert) man alle Kantengewichte um eine Konstante k, so erhöhen (bzw. verringern) sich die kürzesten Entfernungen um ein Vielfaches von k.
   - Sind die kürzesten Wege nicht eindeutig, so findet Dijkstras Algorithmus denjenigen kürzesten Weg mit den wenigsten Kanten. Ist die Aussage für den Bellman-Ford-Algorithmus korrekt?

3. (mittel) **Dijkstra vs. Bellman-Ford:** Zeigen Sie an einem Beispiel, dass beim Bellman-Ford-Algorithmus bei Auswahl eines Knotens der Wert $\mathrm{D}(v)$ aus R nicht unbedingt mit der Weglänge des Weges übereinstimmt, der über die Vorgängerknoten $\mathrm{pred}(\cdot)$ definiert ist.

   Zeigen Sie, dass beim Dijkstra-Algorithmus bei Auswahl eines Knotens der Wert $\mathrm{D}(v)$ aus R stets mit der Weglänge des Weges übereinstimmt, der über die Vorgängerknoten $\mathrm{pred}(\cdot)$ definiert ist. Betrachten Sie dabei auch den für die Bearbeitung von Graphen mit negativen Kantengewichten modifizierten Dijkstra-Algorithmus.

4. (schwer) **Dijkstra $\cup$ Bellman-Ford:** Dijkstra funktioniert gut auf Graphen mit positiven Kantengewichten, Bellman-Ford hat einen deutlich besseren Worst-Case bei Graphen mit negativen Kantengewichten.

   Überlegen Sie, wie man diese beiden Algorithmen so kombinieren kann, dass soweit möglich die Knotenauswahl nach Dijkstra durchgeführt wird und nur bei Korrekturen (d.h., ein Knoten wird nach Auswahl aus R zu einem späteren Zeitpunkt erneut in R eingefügt) auf Bellman-Ford umgeschwenkt wird. Wann kann man dann erneut die Auswahl nach Dijkstras Kriterium ausführen? Welchen Worst-Case hat Ihr Algorithmus?

5. (leicht) **A^*-Algorithmus:** Sind den Knoten Koordinaten (in der Ebene) zugeordnet, so kann der euklidische Abstand als Schätzfunktion verwendet werden. Zeigen Sie: Die so gewählte Schätzfunktion ist sowohl zulässig als auch konsistent.

6. (mittel) **A^*-Algorithmus:** Das Schiebepuzzle besteht aus einem 4×4 Felder großen Quadrat. Die Felder sind mit 1 bis 15 durchnummeriert (ein Feld bleibt frei). Zu Beginn sind die Zahlen zufällig auf die 16 möglichen Plätze verteilt. Auf das freie Feld kann jeweils eine der direkt benachbarten Zahlen verschoben werden und so kann man von einem Zustand des Spiels in den nächsten übergehen. Ziel ist es, den Zustand zu erreichen, bei dem die Zahlen 1 bis 15 von links oben nach rechts unten (zeilenweise) sortiert sind und das Feld rechts unten frei ist.

   Aufgabe: Finden Sie eine zulässige und konsistente Schätzfunktion, um im Zustandsgraphen dieses Spiels mit Hilfe des A^*-Algorithmus eine Folge von möglichst wenigen Zügen zum Zielzustand zu ermitteln. (Jeder Zustand des Spielfelds ist ein Knoten, eine Kante wird gezogen, wenn der Übergang zwischen den Zuständen möglich ist.)

7. (leicht–schwer) **Zwei Linearzeit-Algorithmen:** In ungewichteten Graphen (bzw. Graphen, in denen alle Kantengewichte 1 sind) lassen sich durch einfache Breitensuche in $O(n+m)$ die kürzesten Wege vom Startknoten zu allen anderen Knoten berechnen.

   (leicht–mittel) Entwerfen Sie einen $O(n+m)$-Algorithmus, wenn für alle $e \in E$ die Kantengewichte ganzzahlig sind und $1 \leq \gamma(e) \leq 3$ gilt.

   (schwer) Entwerfen Sie einen $O(n+m)$-Algorithmus, wenn für alle $e \in E$ für die Kantengewichte $1 \leq \gamma(e) \leq 3$ gilt (nicht notwendigerweise ganzzahlig).
