Grundlagen der Künstlichen Intelligenz (CS 205)

Prof. Dr. M. Helmert
Dr. M. Wehrle
Frühjahrssemester 2014

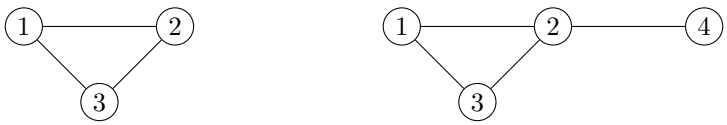
Universität Basel
Fachbereich Informatik

Übungsblatt 9

Abgabe: 16. Mai 2014

Aufgabe 9.1 (1+1+1+2 Punkte)

Betrachten Sie die folgende Variante des *Königsberger Brückenproblems*: Gibt es für eine gegebene Menge von Brücken über Flüsse eine Tour, die jede Brücke genau einmal überquert? Dieses Problem wurde ursprünglich durch 7 Brücken, die sich in der Stadt Königsberg befinden, motiviert. Formal kann das Problem folgendermassen definiert werden: Gibt es für einen gegebenen Graphen $G = (V, E)$ mit Knoten V und Kanten $E \subseteq V \times V$ sowie einem gegebenen Anfangsknoten $v_0 \in V$ eine Sequenz von Knoten aus V, die in v_0 startet, sodass je zwei aufeinanderfolgende Knoten durch eine Kante aus E verbunden sind, und jede Kante in E genau einmal in dieser Sequenz vorkommt. Das heisst, Knoten dürfen mehrfach, aber Kanten müssen genau einmal besucht werden. Beispiel:



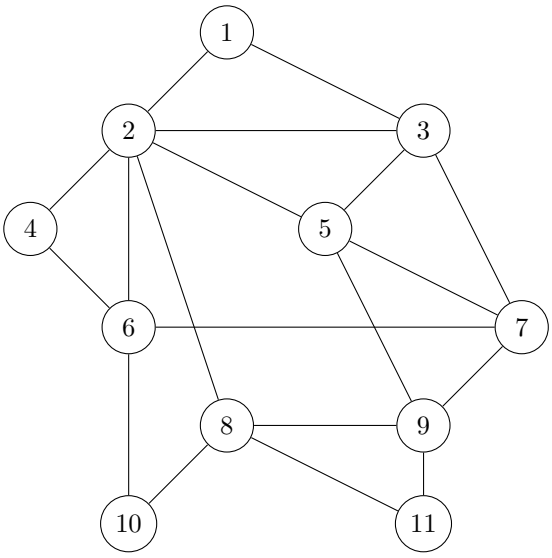
Im linken Graphen gibt es eine Tour, die im Knoten 1 startet und jede Kante genau einmal verwendet. Im rechten Graphen gibt es keine solche Tour, die im Knoten 1 startet (aber eine solche Tour, die im Knoten 2 startet).

(a) Auf der Vorlesungsseite finden Sie eine Beschreibung des Königsberger Brückenproblems (bestehend aus 7 Brücken) in PDDL. Die Beschreibung der Domäne finden Sie in der Datei `bridges.pddl`, die Beschreibung des Problems in `koenigsberg-strips.pddl`. Geben Sie im Stil des obigen Beispiels eine graphische Repräsentation (inklusive Anfangsknoten) des Problems in der Form eines Graphen an.

(b) In dieser Aufgabe soll das Königsberger Brückenproblem von dem domänenunabhängigen Planungssystem *Fast Downward* gelöst werden. Auf der Vorlesungsseite finden Sie ein entsprechendes Softwarepaket von Fast Downward für Linux. Zur Ausführung von Fast Downward benötigen Sie Python in einer Version ≥ 2.7 (diese sollte in Ihrer Distribution standardmässig mitgeliefert sein). Verwenden Sie Fast Downward mit A*-Suche und der LM-cut-Heuristik, um das Beispielproblem von der Webseite zu lösen. Führen Sie hierzu `./plan bridges.pddl koenigsberg-strips.pddl --search "astar(lmcut())"` aus. Geben Sie die Laufzeit sowie die Anzahl der expandierten Zustände an. Ist das Problem lösbar? Wenn ja, geben Sie auch den gefundenen Plan an.

(c) Modifizieren Sie die Beschreibung der Domäne, sodass es erlaubt ist, dass Brücken auch mehrmals benutzt werden dürfen. Modifizieren Sie hierzu die Datei `bridges.pddl`. Lösen Sie das resultierende Problem mit Fast Downward analog zu b). Geben Sie die Laufzeit sowie die Anzahl der expandierten Zustände an. Ist das Problem lösbar? Wenn ja, geben Sie auch den gefundenen Plan an.

(d) Formalisieren Sie die folgende Instanz des Königsberger Brückenproblems (siehe nächste Seite) in PDDL und lösen Sie es mit Fast Downward analog zu b) für die Variante, in der Brücken nur einmal benutzt werden dürfen. Der Anfangsknoten ist Knoten 1. Geben Sie den gefundenen Plan, die Laufzeit und die Anzahl der expandierten Zustände an.



Bemerkung: Bei Interesse finden Sie weitere und detailliertere Hinweise zu Fast Downward auf der Webseite `http://www.fast-downward.org/`.

Aufgabe 9.2 (1+2+2 Punkte)

Betrachten Sie die STRIPS-Planungsaufgabe $\Pi = \langle V, I, G, A\rangle$ mit $V = \{a, b, c, d, e, f\}$, $I = \{a, b\}$, $G = \{e, f\}$, und $A = \{a_1, a_2, a_3\}$ mit $cost(a_1) = 8$, $cost(a_2) = 10$, und $cost(a_3) = 12$. Hierbei sei a_1 durch $pre(a_1) = \{a\}$ und $add(a_1) = \{c\}$ definiert, a_2 durch $pre(a_2) = \{b\}$ und $add(a_2) = \{d\}$ definiert, sowie a_3 durch $pre(a_3) = \{c, d\}$ und $add(a_3) = \{e, f\}$ definiert. Die Delete-Effekte seien bei allen drei Aktionen leer.

(a) Berechnen Sie $h^+(I)$. Geben Sie den Rechenweg an.

(b) Berechnen Sie den relaxierten Planungsgraphen für Π bis zur Tiefe 2.

(c) Berechnen Sie $h^{\max}(I)$, $h^{\text{add}}(I)$ und $h^{\text{FF}}(I)$. Geben Sie jeweils den Rechenweg an.

Aufgabe 9.3 (2 Punkte)

Betrachten Sie folgende STRIPS-Kodierung für das 8-Puzzle-Problem, bei der eine 8-Puzzle-Planungsaufgabe $\Pi = \langle V, I, G, A\rangle$ gegeben ist durch

- Variablenmenge V bestehend aus
  - $\text{tile-is-in-cell}_{t,c}$ für $t \in \{1, \ldots, 8\}$ und $c \in \{(1,1), \ldots, (3,3)\}$
  - cell-is-empty_c für $c \in \{(1,1), \ldots, (3,3)\}$
- Ziel $G = \{\text{tile-is-in-cell}_{1,(1,1)}, \ldots, \text{tile-is-in-cell}_{8,(3,2)}\}$
- Operatoren

$$A = \{move_{t,c,c'} \mid t \in \{1, \ldots, 8\}, c \in \{1,2,3\} \times \{1,2,3\}, \\ c' \in \{1,2,3\} \times \{1,2,3\} \text{ und } c, c' \text{ benachbart.}\}$$

  Die Operatoren haben alle Kosten 1 und sind folgendermassen definiert:

  - $pre(move_{t,c,c'}) = \{\text{tile-is-in-cell}(t, c), \text{cell-is-empty}(c')\}$
  - $add(move_{t,c,c'}) = \{\text{tile-is-in-cell}(t, c'), \text{cell-is-empty}(c)\}$

  - $del(move_{t,c,c'}) = \{\text{tile-is-in-cell}(t, c), \text{cell-is-empty}(c')\}$

- Wir nennen einen Zustand *legal*, wenn jedes Plättchen an einer Position ist, sich keine zwei Plättchen an einer Position befinden und die verbleibende Position als einzige leer ist. Der Anfangszustand I ist dann ein beliebiger legaler Zustand.

Zeigen Sie für diese Formalisierung die Aussage aus der Vorlesung: Für das 8-Puzzle wird die Heuristik h^{MD} basierend auf der Manhattan-Distanz von der h^+-Heuristik dominiert (formal: $h^{\text{MD}}(s) \leq h^+(s)$ für alle legalen Zustände s).

Die Übungsblätter dürfen in Gruppen von zwei Studierenden bearbeitet werden. Bitte schreiben Sie beide Namen auf Ihre Lösung.