Übungen zur Vorlesung
Grundlagen der Künstlichen Intelligenz
Sommersemester 2016

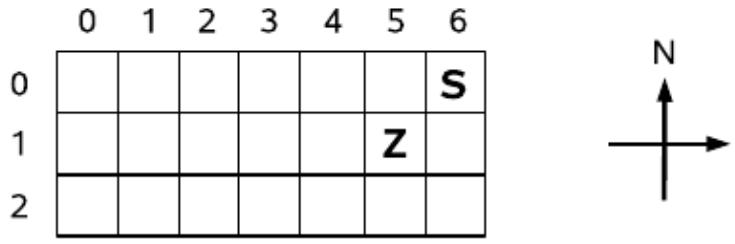
Institut für Informatik 4
Priv.-Doz. Dr. V. Steinhage
Friedrich-Ebert-Allee 144
53113 Bonn
Email: steinhage@cs.uni-bonn.de
WWW: http://net.cs.uni-bonn.de/ivs/

# Blatt 2 (9 Punkte)

Abgabe durch Hochladen auf der eCampus-Seite bis Sonntag, 24.04.2016, 23:59 Uhr, in Gruppen von 2-3 Personen.

## Aufgabe 2.1: Uninformierte Suche $(2 + 1 = 3)$

Wenden Sie bei den folgenden Aufgaben den Suchalgorithmus GENERAL-SEARCH (Vorlesung 3, Folie 22) an. Beachten Sie bitte genau, an welcher Stelle des Algorithmus der GOAL-TEST angewendet wird.
Ein Roboter soll in einem Raum einen Weg von seinem ihm bekannten Startzustand $\mathsf{S} = (6, 0)$ zu einem ihm unbekannten Zielfeld $\mathsf{Z}$ suchen. Er hat keinerlei Sensoren, um die Richtung des Ziels zu bestimmen. Er besitzt lediglich einen Sensor um festzustellen, ob er sich aktuell im Zielfeld befindet (GOAL-TEST). Mit einem Schritt kann der Roboter entweder horizontal oder vertikal in ein angrenzendes Feld wechseln, diagonale Bewegungen sind nicht erlaubt, d.h. seine möglichen OPERATORS sind *Ost, Nord, West, Süd* (in genau dieser Reihenfolge). Die Roboterumgebung sei durch folgende Abbildung definiert:




Die Zahlen geben die Feldkoordinaten der Umgebung an. Der Roboter weiß, wo sich die Wände des Raums befinden, und wird nie versuchen, die Umgebung zu verlassen.

a) Suchen Sie in diesem Aufgabenteil das Zielfeld mittels *Breitensuche*. Gehen Sie davon aus, dass der Roboter keine Möglichkeit hat, sich bereits besuchte Felder zu merken.

   i) Beschriften Sie die Felder der Umgebung nach der Reihenfolge ihrer (evtl. mehrfachen) Expansion bis ersichtlich ist, wann das Ziel entdeckt wird.

   ii) Notieren Sie auch die Entwicklung der Warteschlange (QUEUE).

   iii) Wieviele Knoten werden insgesamt expandiert?

   iv) Wieviele Einträge umfasst die Warteschlange maximal?

   v) Zeichnen Sie auch den resultierenden Pfad ein.

   vi) Wie lang ist der resultierende Pfad?

b) Führen Sie dieselbe Aufgabenstellung wie in a) mit *Tiefensuche* durch.

**Aufgabe 2.2: Bewertung von Suchstrategien** **(1)**

Bewerten Sie anhand der Kriterien *Vollständigkeit, Optimalität, Zeit-* sowie *Platzkomplexität* eine *Bidirektionale Suche*, die für eine Suchrichtung *Breitensuche* und für die andere *Tiefensuche* verwendet.

**Aufgabe 2.3: Verwandtschaft von Suchstrategien** **$(0.5 + 0.5 = 1)$**

Zeigen Sie:

a) Die *Breitensuche* ist ein Sonderfall der *uniformen Kostensuche*.

b) *Uniforme Kostensuche* ist ein Sonderfall der *A*-Suche*

**Aufgabe 2.4: Programmieraufgabe: Routenplanung** **$(2 + 2 = 4)$**

**Vorbereitung**. Laden Sie bitte das ZIP-Archiv AIMA-Py-Routenplanung.zip herunter von unserer eCampus-Seite unter Kursunterlagen » Python und AIMA Python » AIMA-Py Routing. Das ZIP-Archiv enthält die beiden Ordner *aima* und *routen* sowie ein Handbuch.
Die gesamte AIMA-GUI wird gestartet mit dem Skript AIMA.py im Ordner *aima*. AIMA.py importiert für verschiedene Zeichenoperationen zeichne.py und für die Bearbeitung von Graphen GRAPH.py, das wiederum Node.py und zeichne.py importiert. Die restlichen Skripte im Ordner *aima* sind noch nicht von Bedeutung.
Im Ordner *routen* bietet Aimaqueue.py verschieden Queue-Klassen an. Die Klasse FiFoQueue wird von breitenqueue.py importiert, das die die Breitensuche umsetzt.
Eine Straßenkarte is vorgegeben in der ASCII-Datei *7Cities12Streets.map*. Für die Aufgabe ist nur diese Karte nötig und die Datei nicht zu bearbeiten.
Wenden Sie zunächst die Breitensuche an, um den kürzesten Weg von Berlin nach Stuttgart zu finden: Start von Skript AIMA.py erzeugt die GUI als neues Fenster. Dort *Select Application » Routefinding » Openmap*. Auswahl von *7Cities12Streets.map*. Die Karte erscheint. Die Zahlen geben die wahren Straßenlängen an. Auswahl *Load Algorithm » Load* » Ordner *routen » breitenqueue.py » Run*. Das gesamte GUI-Fenster wird am besten maximiert. Die Suche terminiert mit der Melding *Done*. Das Ergebnis und der Weg dazu stehen im rechten Anzeigeteil der GUI. Für Berlin-Stuttgart wird z.B. eine Route mit 370 km gefunden.

**Aufgabe**: Implementieren Sie die A*-Suche zur Lösung der Routenplanung.

a) Ergänzen Sie dazu zunächst die Queue-Klasse *SortedQueue* in Aimaqueue.py, welche die zu expandierenden Knoten nach ihren f-Kosten heraussucht (ca. 5 - 6 Codezeilen).

b) Gehen Sie vom Skript Breitensuche.py (Breitensuche zur Lösung der Routenplanung) aus. Kopieren Sie dies in ein neues Skript AStar.py und modifizieren Sie die gegebene Breitensuche nun geeignet zur A*-Suche. Es handelt sich dabei um lediglich ca. 5 - 6 Codezeilen, die zu erstellen bzw. zu ändern sind. Als Heuristik ist die Luftliniendistanz zu verwenden. Diese kann über graph.luftlinie(nameA, nameB) erfragt werden.
Hinweis: Die A*-Suche sollte die Route Berlin-Stuttgart mit 360 km finden.