Technische Universität Berlin
Fakultät IV-Elektrotechnik und Informatik

Grundlagen der Künstlichen Intelligenz WS 04/05

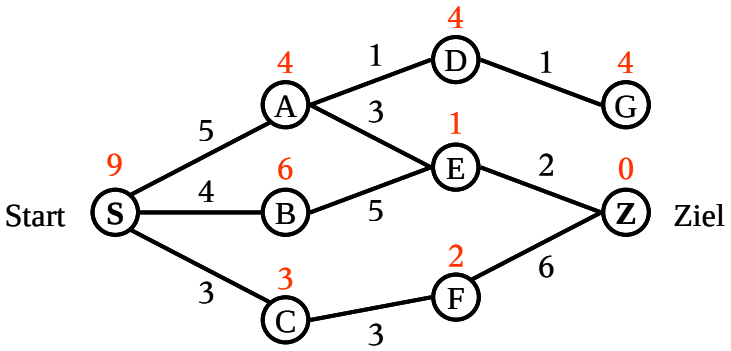
Albayrak, Fricke, Jain (AOT) - Obermayer, Martin (NI)

2. Übungsblatt

Abgabe: 16.11.2005

Aufgabe 1 **(3 Punkte)**

Betrachte den folgenden Problemgraphen mit Anfangszustand S und Zielzustand Z:



Aktionskosten sind schwarz an den Kanten und Restwegkostenschätzungen rot an den Knoten annotiert.

Finden Sie eine Lösung mit `Best-First Search`, `Branch & Bound` und `A*`. Besitzen mehrere Teilpfade eine minimale Bewertung, so wird nach lexikographischer Ordnung expandiert. Die Algorithmen sind im Anhang beschrieben.

a) Simulieren Sie die Suche mit Best-First Search nach folgendem Schema:

Iteration	Liste	Nachfolger
0	<S-9>	<SA-4>, <SB-6>, <SC-3>
1	<SC-3>, <SA-4>, <SB-6>	…

Geben Sie dabei die Liste immer vor Beginn eines Schleifendurchlaufs (vor Schritt 2a) und am Schluss (nach Schritt 3) an. Geben Sie in der Spalte

„*Nachfolger*“ die Expansionen des ersten Teilpfades der Liste in der jeweiligen Zeile an. Nachfolger sind alle Expansionen, die in die Queue gemäß Schritt 3d) eingefügt werden. Die Nachfolger dürfen unsortiert angegeben werden.

b) Simulieren Sie die Suche mit Branch & Bound wie in Aufgabenteil a).

c) Simulieren Sie die Suche mit A* wie in Aufgabenteil a).

Aufgabe 2 **(7 Punkte)**

Das Ziel beim 8-Damen Problem ist es, acht Damen auf dem Schachbrett so zu platzieren, dass keine Dame eine andere attackiert. Eine Dame attackiert eine andere Dame, wenn beide in derselben Spalte, Zeile oder Diagonalen stehen.

Eine Verallgemeinerung des 8-Damen Problems ist das n-Damen Problem. Wie beim 8-Damen Problem geht es darum n Damen auf ein Brett mit n × n Feldern so zu platzieren, dass keine Dame eine andere schlägt.

Für die Bearbeitung der Aufgabe sei die **Problemformulierung** für das n-Damen Problem von der folgenden Form:

- **Problemraum:** Jede Konfiguration von n Damen auf dem Brett, so dass sich in jeder Spalte genau eine Dame befindet.

- **Startzustand:** Zufällige Konfiguration. Platziere in jeder Spalte eine Dame auf ein zufällig ausgewähltes Feld. Innerhalb einer Spalte wird jedes Feld mit Wahrscheinlichkeit 1/n ausgewählt.

- **Zielzustand:** Konfiguration von n Damen, sodass keine Dame eine andere attackiert.

- **Aktionen:** Verschiebe eine Dame innerhalb einer Spalte.

Die **heuristische Funktion** `h` gibt für jeden Zustand `S` die Anzahl `h(S)` aller Paare von Damen, die sich direkt und indirekt attackieren. Zwei Damen attackieren sich indirekt, wenn zwischen ihnen mindestens eine weitere Dame in der „*Schlaglinie*“ steht.

Löse das n-Damen Problem für `n = 5, 10, 15, 20` mit den Algorithmen `A = Best-First Search, Steepest Descent, Simulated Annealing`. Die Algorithmen sind im Anhang des Übungsblatts beschrieben und sollen in Java oder C/C++ implementiert werden.

Wiederhole für jedes oben genannte `n` das Experiment mit einer zufälligen Startkonfiguration 100-mal.

a) Bestimme für jedes oben genannte `n` und jeden oben genannten Algorithmus `A` den Mittelwert $(h(S_{A,1})+\ldots+h(S_{A,100}))/100$, wobei $S_{A,i}$ den Zustand nach Termination von Algorithmus `A` im i-ten Experiment bezeichne.

b) Bestimme für jedes oben genannte n und jeden oben genannten Algorithmus A den prozentualen Anteil der korrekten Lösungen.
c) Bestimme für jedes oben genannte n und jeden oben genannten Algorithmus A die mittlere „CPU“ Zeit.
d) Vergleiche die Algorithmen und diskutiere die Resultate bzgl. Qualität (Wert der heuristischen Funktion) und Zeitaufwand in Abhängigkeit der Problemgröße.

Abzugeben sind

- Der Quellcode und eine README Datei an bjj@dai-labor.de. Die README Datei beschreibt, wie das Programm kompiliert und gestartet wird.
- Ein Ausdruck des Quellcodes.
- Tabellarisch die Ergebnisse von a)-c).
- Diskussion (max. 5-7 Sätze).

Bemerkung: Benötigt ein Algorithmus A auf dem zur Verfügung stehenden Rechner für gegebenes $n = m$ mehr als 3 min für ***eine*** Lösung des m-Damen Problems, so darf auf die Auswertung von Algorithmus A für alle $n \geq m$ verzichtet werden.

Hinweise:

- **Abgabe der Lösungen:** Bitte Seite 6 ausgefüllt abgeben.
- Beschreibungen der Algorithmen befinden sich auf den Seiten 4-5.

Anhang: Algorithmen

- **Algorithmus A* [Winston 93]**

1. Form a one-element queue consisting of a zero-length path that contains only the root node.
2. Until the first path in the queue terminates at the goal node or the queue is empty
   a) Remove the first path from the queue; create new paths by extending the first path to all neighbors of the terminal node.
   b) Reject all paths with loops.
   c) If two or more paths reach a common node N, delete all those paths except the one that reaches N with the minimum cost. (*Dynamic Programming*)
   d) Add the remaining new paths, if any, to the queue.
   e) Sort the entire queue by ascending order of the evaluation function f = g + h
3. If the goal node is found, announce success; otherwise announce failure.

- **Best-First Search** und **Branch & Bound** können von A* wie folgt abgeleitet werden:
  - **Best-First Search:**
    - Keine Pfadkosten (g = 0)
    - Keine dynamische Prorgammierung (Schritt 2c entfällt)
  - **Branch & Bound:**
    - Keine Restwegkostenschätzung (h = 0)
    - Keine dynamische Prorgammierung (Schritt 2c entfällt)

- **Steepest Descent**

Definition: Sei S ein Zustand. Ein *Nachbarzustand* von S ist ein Zustand S′, der von S durch Anwendung einer Aktion erreichbar ist.

Notation:

S	Zustand des Problemraums
N(S)	Menge aller Nachbarzustände von Zustand S
h	heuristische Funktion

Algorithmus:

S = random initial state.
while `true` **do**
  select candidate state S′ ∈ N(S) with lowest value h(S′)
  if h(S′) ≥ h(S) **then return** h(S)
  S = S′

- **Simulated Annealing**

Notation:

T	Temperatur
T_0	Anfangstemperatur
S	Zustand des Problemraums
ε	Schwelle für Abbruchkriterium
t	Zähler für Iterationen
h	heuristische Funktion
$lower(T, t)$	Funktion mit $lower(T, t) \leq T$

Algorithmus:

$T = T_0$
$S =$ random initial state
set $\varepsilon \geq 0$
$t = 1$
while $T > \in$ **do**
  draw an action a from a uniform distribution
  apply action a to the current state S to obtain candidate state S'
  compute $\Delta E = h(S) - h(S')$
  draw random number ρ from a uniform distribution over $[0, 1[$
  if $\rho < \exp(\Delta E/T)$ **then** $S = S'$
  $T = lower(T, t)$
  $t = t+1$
return $h(S)$

Annealing Schedule für das n-Damen Problem:

$T_0 = n^2$
$\varepsilon = 0.1$
$k = n^2$
$\alpha = 0.9$
$lower(T, t) = \alpha T$, wenn $t \bmod k = 0$
$lower(T, t) = T$, sonst

Um ein Gefühl für das Verhalten von Simulated Annealing zu erhalten, können die Parameter T_0, ε, α, k variiert werden.

Übungsblatt 2

Name & Matrikelnummer 1:

Name & Matrikelnummer 2:

Bitte Aufgabenteile ankreuzen, die vollständig bearbeitet wurden. Kommentare zur Bearbeitung können in der entsprechenden Spalte oder unterhalb der Tabelle eingetragen werden.

Aufgabe	Bearbeitet	Kommentar
1a		
1b		
1c		
2a		
2b		
2c		
2d		