DAI-Labor
TU Berlin

Grundlagen der Künstlichen Intelligenz

Übungen

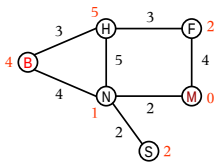
09.11.2005

Brijnesh J Jain
bjj@dai-labor.de

A|O|T
Agententechnologien in betrieblichen Anwendungen und der Telekommunikation

Beispiel 1: Routen finden

B	Berlin (Start)
F	Frankfurt
H	Hannover
M	München (Ziel)
N	Nürnberg
S	Stuttgart



A* Algorithmus [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.
   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.

Branch and Bound

Algorithmus Branch and Bound [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) Add the remaining new paths, if any, to the queue.
   d) Sort the entire queue by ascending order of the evaluation function **f = g**
3. If the goal node is found, announce success; otherwise announce failure.

Best First Search

Algorithmus Best First Search [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) Add the remaining new paths, if any, to the queue.
   d) Sort the entire queue by ascending order of the evaluation function **f = h**
3. If the goal node is found, announce success; otherwise announce failure.

Lokale Suche

- ⇨ Einleitung
- ⇨ Steepest Descent / Hill Climbing
- ⇨ Simulated Annealing

Einleitung

- Suchbaumverfahren suchen nach einem Pfad von Start zum Ziel
- Für viele Probleme ist
  - der Pfad zum Ziel irrelevant
  - Das Ziel ist die Lösung
- Beispiel:
  - 8-Damen Problem
  - Minimiere Anzahl der 1, 2, 5 und 10 Cent Münzen für 3, 23 €
- Frage: Kann man für die Suche nach einer Lösung die Irrelevanz der Pfade zum Ziel ausnutzen?

Einleitung

- Lokale Suche
  - Betrachtet nur aktuellen Zustand S
  - Merkt sich keine Teilpfade
  - Geht ausgehend von S in einen benachbarten Zustand S‘ über
- Vorteile:
  - Konstante Speicherkomplexität
  - Findet oftmals „*gute*“ Lösungen in „*akzeptabler*“ Zeit

Einleitung

- Beispiel: 4-Damen Problem
  - Ziel: Platziere 4 Damen auf ein 4×4-Schachbrett, sodass sich keine 2 Damen in der gleichen Spalte, Zeile, Diagonalen befinden
  - Problemraum: Zustände sind alle Konfigurationen mit je einer Dame in jeder Spalte
  - Anfangszustand: Zufällig
  - Zielzustand: 4 Damen auf dem Brett, sodass sich keine 2 Damen direkt oder indirekt attackieren
  - Aktionen: Bewege eine Dame innerhalb einer Spalte

Einleitung

- Lösung des 4-Damen Problem mit lokaler Suche
  - Wähle geeignete Aktion aus, um Nachbarzustand zu erzeugen
  - Verschiedene lokale Suchverfahren unterscheiden sich in der Strategie, mit der man zu einem Nachbarzustand übergeht
    - Definiere geeignete heuristische Funktion h
    - Maximiere/minimiere heuristische Funktion

Steepest Descent

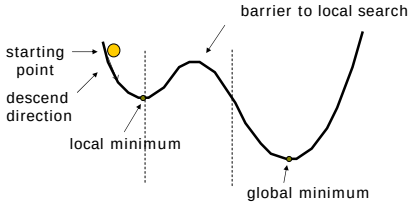
Algorithmus:

```
define heuristic function h to minimize
initialize current state S
while true do
    move to neighbouring state S' of S with minimum value h(S')
    if h(S') ≥ h(S) then return h(S)
    S = S'
end while
```

- Hill Climbing: Maximierung der heuristischen Funktion h mit Hill Climbing entspricht Minimierung von –h mit Steepest Descent

Steepest Descent

- Problem:
  - Lokale Minima (auch Plateaus, Grate)

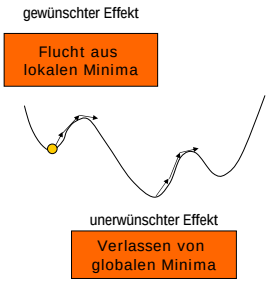


Simulated Annealing

- Simulated Annealing (simulierte Abkühlung)
  - Nachbildung eines physikalischen Abkühlungsprozesses zur Optimierung von Funktionen
  - Erlaubt Flucht aus lokalen Minima
  - Analogie: Schmieden eines Schwerts mit Abkühlen
    - Zu schnelles Abkühlen des Metallgemischs formt ein Flickenteppich von verschiedenen Metallen.
      - Schwert ist brüchig, entspricht lokalem Energieminimum
    - Langsames Abkühlen des Metallgemischs führt zu einer gleichmäßigen Vermischung der Metalle
      - Schwert ist hart, entspricht im Idealfall einem globalem Energieminimum

Simulated Annealing

- Idee: Flucht aus lokalen Minima durch gelegentlichen Aufstieg der zu minimierenden Funktion
- Konsequenzen: Gute bzw. optimale Zustände können wieder verlassen werden
- Ausweg: Merke immer den Zustand mit dem bisher kleinsten Wert der Energiefunktion



Simulated Annealing

Informelle Beschreibung des Algorithmus

1. Initialisiere Anfangszustand S
2. Initialisiere Temperaturparameter T
3. Wiederhole bis Abbruchkriterium erfüllt
   a) Erzeuge zufälligen Nachbarzustand S‘ von S
   b) Wenn $h(S') < h(S)$, dann akzeptiere den Zustandsübergang. Setze $S = S'$.
   c) Wenn $h(S') > h(S)$, dann akzeptiere den Zustandsübergang mit einer Wahrscheinlichkeit, die umso kleiner ist, je größer $(h(S') - h(S))/T$ ist
   d) Verringere Temperatur T

Simulated Annealing

- Eigenschaft: Wenn man T ausreichend langsam abkühlen lässt, dann findet SA ein globales Minimum mit Wahrscheinlichkeit 1.
- Formale Beschreibung von SA für das n-Damen Problem siehe Anhang von Übungsblatt 2

Simulated Annealing

- S. Kirkpatrick et al.: Optimization by Simmulated Annealing http://www.fisica.uniud.it/~ercolessi/MC/kgv1983.pdf
- Weitere Referenzen:
  - Citeseer http://citeseer.ist.psu.edu/
  - Google-Scholar http://scholar.google.com/

    Suchbegriff: „simulated annealing“ liefert

    http://scholar.google.com/scholar?q=%22simulated+annealing%22&ie=UTF-8&oe=UTF-8&hl=en&btnG=Search