Konzepte der AI: Moderne Heuristiken

Nysret Musliu
Database and Artificial Intelligence Group
Institut für Informationssysteme, TU-Wien

Gliederung

- Heuristiken-Einführung
- Lokale Suchmethoden
  - Hill-Climbing Strategien
  - Simulated Annealing
  - Tabu-Suche
- Genetische Algorithmen

Heuristiken

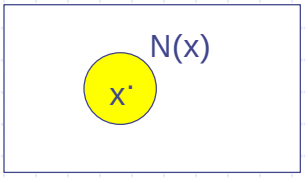
- Polya (1945) definierte Heuristiken als
  „das Studium der Methoden und Regeln von Entdeckung und Erfindung“
- Das griechische Wort „heuriskein“ bedeutet: „Ich entdecke“.
- Newell, Shaw, and Simon (1963) definierten Heuristiken als
  „a process that may solve a given problem, but offers no guarantees of doing so“

Heuristiken

- Heuristiken werden hauptsächlich in folgenden Fällen angewendet:
  - Ein Problem besitzt eine exakte Lösung, doch die Rechenkosten für die Suche nach dieser Lösung sind unerschwinglich
    - Schah
    - Traveling Salesman Problem
  - Ein Problem hat keine Lösung, da die Problembeschreibung oder die verfügbaren Daten zweideutig sind:
    - Medizinische Diagnose
    - Das Sehvermögen

Lokale Suchmethoden

- Basieren auf der Nachbarschaft der aktuellen Lösung.



- Die Lösung wird schrittweise (iterativ) mit Reparatur-basierten Schritten (`moves') geändert, bis man eine akzeptable oder optimale Lösung findet.

Lokale Suchmethoden

- Hill-Climbing
- Simulated Annealing
- Tabu-Suche
- ...
- Unterscheiden sich hauptsächlich in der Art und Weise wie sie:
  - die Nachbarschaft erkunden.
  - die nachfolgende Lösung der aktuellen Lösung akzeptieren.

Hill-Climbing

- Erzeuge eine anfängliche Lösung s.
- Erzeuge Nachbarschaft N(s) von Lösung s,
- Selektiere in Nachbarschaft die Lösung mit bester Fitneß.
- Wenn die selektierte Lösung besser als die aktuelle Lösung ist, akzeptiere diese als aktuelle Lösung für die nächste Iteration und fahre mit Schritt 2 fort.
- Wenn die selektierte Lösung schlechter als die aktuelle Lösung ist, beende die Suche.

Hill-Climbing

- Stoßt auf das lokale Optimum
- Random Restart Hill-Climbing
  - Beginnt von verschiedenen Anfangspunkten.
- Stochastic Hill-Climbing
  - Nur eine Lösung aus der Nachbarschaft wird selektiert.
  - Diese Lösung wird für die nächste Iteration mit einer Wahrscheinlichkeit akzeptiert, die von der Evaluierungsdifferenz zwischen dieser Lösung und der aktuellen Lösung abhängig ist.

Stochastic Hill-Climbing

```
Prozedure Stochastic Hill-Climbing ([1])
 begin
   t=0
   select a current string v_c at random
   evaluate v_c
   repeat
    select the string v_n from the neighborhood of v_c
    select v_n with probability 1 / (1 + e^((eval(v_c) - eval(v_n)) / T))
    t=t+1
   until t=MAX
 end
```

- Je größer der Wert von T, desto kleiner die Bedeutung der Evaluierungsdifferenz zwischen den Lösungen v_c and v_n.

Simulated Annealing

- Während der Suche wird auch eine schlechtere Lösung mit einer bestimmten Wahrscheinlichkeit akzeptiert.
- Unterscheidet sich von der Stochastic-Hill-Climbing-Methode dadurch, dass der Parameter T während der Suche verändert wird.
- Am Anfang der Suche hat T hohe Werte, während sie am Ende sehr klein sind.

Simulated Annealing

```
Prozedure Simulated Annealing ([1])
 begin
    t=0
    Intialize T
    select a current string v_c at random
     evaluate v_c
    repeat
      repeat
      select a new point v_n in the neighborhood of v_c
      if eval(v_c) < eval(v_n) then
         v_c =v_n
         else if random[0,1) < e^((eval(v_n)-eval(v_c))/T)  then v_c=v_n
       until (termination-condition)
       T=g(T,t)
       t=t+1
    until (halting-criterion)
 end
```

Simulated Annealing

- Wichtige Fragen für SA:
  - Wie bestimme ich die Anfangs-„Temperatur“ T?
  - Wie bestimme ich die Funktion g(T,t)?
  - Wie bestimme ich die „termination condition“?
  - Wie bestimme ich das „halting criterion“?

Tabu-Suche

- Durch die Verwendung einer Tabu-Liste werden während der Suche Zyklen (Besuchen von gleichen Lösungen) vermieden.
- In der Tabu-Liste werden für eine fixe Anzahl von durchlaufenen Iterationen bestimmte Informationen zur Suche gespeichert (z.B. bereits besuchte Lösungen).
- Die Akzeptanz der Lösung für die nächste Iteration hängt ab von:
  - Fitneß der Lösung, Tabu-Liste.

Tabu-Suche

```
Procedure Tabu-Suche
begin
 Initialize tabu list
 Generate randomly Initial Solution s_c
 Evaluate s_c
 repeat
   Generate all neighborhood solutions of the solution s_c
   Find best solution s_x in the neighborhood
   if s_x is not tabu solution then s_c= s_x
     else if 'aspiration criteria' is fulfilled then
        s_c= s_x
     else
      find best not tabu solution in the neighborhood s_nt
      s_c = s_nt
   Update tabu list
 until (terminate-condition)
end
```

Tabu-Suche

- „aspiration criteria“ erlaubt das Akzeptieren von Tabu-Lösungen unter bestimmten Umständen, z.B. wenn die gefundene Lösung die bisher beste ist.
- Tabu-Suche ist deterministisch.

Genetische Algorithmen

- Basieren auf biologischen Prozessen
- Bei Verwendung von Genetischen Algorithmen zur Problemlösung werden drei Phasen durchlaufen:
  - Repräsentation von potenziellen Lösungen der Problemdomäne (z.B. in Bit-Strings).
  - Erzeugung einer neuen Generation von Individuen, die Merkmale ihrer Eltern in sich vereinen, mit Paarungs- und Mutationsalgorithmen.
  - Selektion der besten Individuen für die nächste Generation basiert auf einer Fitnessfunktion.

Genetische Algorithmen

P(t) sei definiert als eine Population mit Lösungskandidaten x_i zum Zeipunkt t

Prozedur genetischer Algorithmus ([2])
begin
  setze zeit *t:=0*;
  initialisiere die Population *P(t);*
  While der Abbruchbedingung ist nicht erfüllt **do**
    begin
      evaluiere die Fitness der Mietglieder der Population *P(t);*
      wähle die geeignetsten Mitglieder aus der Population P(t) aus;
      produziere mit genetischen Operatoren Nachkommen diese Paare;
      ersetze die schwächsten Kandidaten von *P(t)* durch die Nachkommen;
      setze Zeit *t:=t+1*
    end
end.

Genetische Algorithmen

- Genetische Operatoren:
  - Kreuzung: Der Kreuzungsmechanismus teilt zwei Lösungskandidaten auf und tauscht Komponenten gegeneinander aus, um neue Kandidaten zu produzieren.
  - Mutation: Ein Aspekt eines Kandidaten wird in zufälliger Weise verändert.

Literatur

1. Z. Michalewicz, D. B. Fogel. *How to Solve It: Modern Heuristics*. Springer Verlag, 2000.
2. G.F. Luger. *Künstliche Intelligenz: Startegien zur Lösung komplexer Probleme*. Pearson Studium, 2001.