Wissensbasierte Planung

Jürgen Dorn

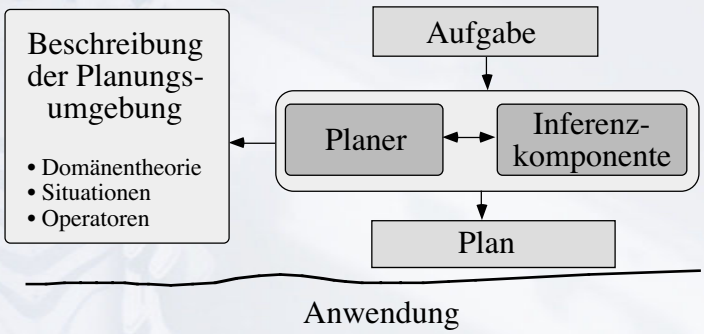
Inhalt

- Wissensrepräsentation
- lineare Planung
- Erweiterungen

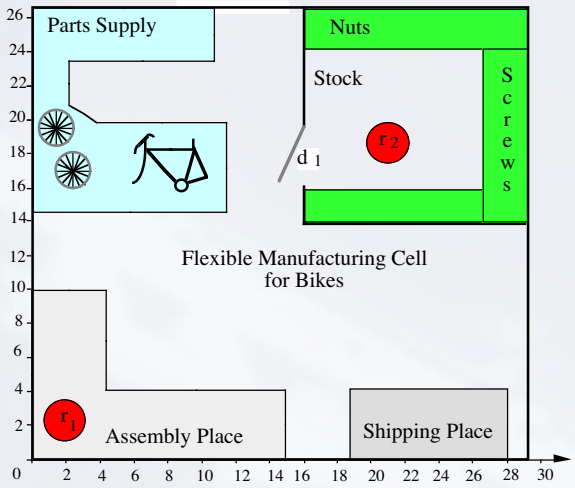
Wissensbasierte Planung

- Handlungsplanung
- was muss getan werden, damit in der Zukunft eine Aussage gilt (z.B. ein Fahrrad produziert ist)
- hier Einschränkung auf statische Probleme
- Dynamische Probleme (Echtzeitplanung)
  - Echtzeitreaktion
  - während der Planung verändert sich die Umgebung
  - Adaptieren von alten Plänen
  - Zuverlässigkeit und Sicherheit

Architektur



Beispiel



Objekte der Umwelt

- Objektklassen und Instanzen

  frame(robot, agent, [capacity, load]).
  frame(assembly_robot, robot, []).
  exist(r_2, mobile_robot, [capacity : 1500]).
  frame(place, self, [region, [x, y]]).
  place(parts_supply, [cell, [3, 29]]).

- Zugriffsfunktionen

  r_2 is_a robot. r_2 at parts_supply.
  r_2 has bike_frame. r_2 in cell.
  bike parts [bike_frame, front_wheel, rear_wheel].
  screws contains screw.

Domänentheorie

- Domänentheorie

  impossible(A_1 at O_1, A_2 at O_2) :- A_1 == A_2, O_1 \= O_2.
  is_a(O, object) :-
    frame(O, object, _),
    exist(_, container, A),
    member(contains : O, A).

Situationskalkül

- Menge von Aussagen, die gleichzeitig existieren
- Startsituation
- Zielsituation
- Situationsübergänge
- Planen als Suche nach adäquaten Situationsübergängen
- Reihenfolgeproblem

Beispiel einer Situation

```
r1 at assembly_place.
r2 at stock.
bike_frame at parts_supply.
front wheel at parts_supply.
rear_wheel at parts_supply.
open d1.
```

Planungsoperatoren

- beschreiben einen Situationsübergang
- repräsentiert durch vier Listen plus Name:
  - Einschränkungen (Bedingungen an die Instantiierung der Variablen)
  - Vorbedingungen (was muss gelten, damit der Operator ausgeführt werden kann)
  - Hinzufügungen (was gilt nach der Ausführung)
  - Aufhebungen (was gilt nicht mehr nach der Ausführung)

Beispiel

```
operator(
  move(R, P1, P2) ::
  [R is_a mobile_robot, P2 is_a place, P2 in Ro,
     Ro is_a room, P1 in Ro, P1 is_a place] ::
  [R in Ro, R at P1] ::
  [R at P2] ::
  [R at P1]).
```

Sieben Operatoren für Beispiel

- move(R, P_1, P_2)
- enter(R, R_1, R_2)
- open(R, D)
- take_from(R, C, O)
- take(R, O, P)
- put(R, O, P)
- assemble(R, O)

Planungstrategie

- Means-End-Analyse
  1. wende Operator an, der die Differenz zwischen Ziel und Startsituation verringert

     welcher Operator fügt Ziele ein
  2. wende Operator an, der die Differenz zwischen Vorbedingungen des Operators und Startsituation verringert
- Zielsituation: `r1 has bike_frame`
- gefundenener Operator: `take(r1, bike_frame)`
- Vorbedingungen: `exist bike_frame, r1 at P,bike_frame at P`

Endgültiger Plan

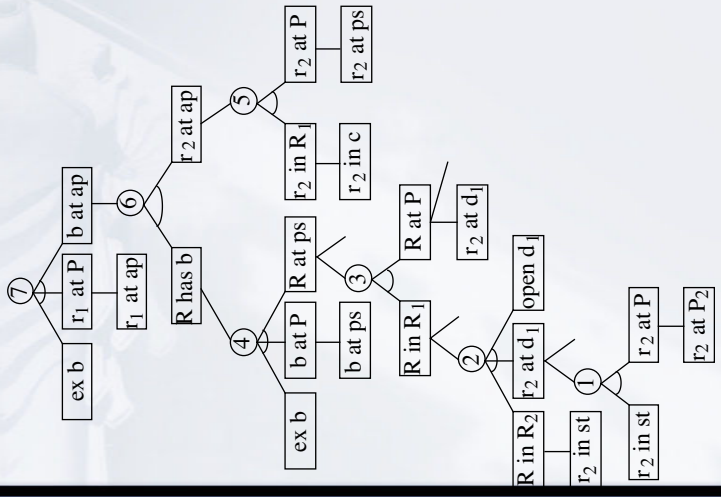
```
move(r2,stock,d1) =>
enter(r2,stock,cell) =>
move(r2,d1,parts_supply) =>
take(r2,bike_frame) =>
move(r2,parts_supply,assembly_place) =>
put(r2,bike_frame) =>
take(r1,bike_frame)
```


„nicht-deterministische" Entscheidungen

1. welches Teilziel einer Liste wird zuerst gelöst
   bisher Reihenfolge der Liste, dann Permutation
2. unterschiedliche anwendbare Schemata
   Heuristik: jeweils spezifischeren Operator auswählen
3. Unifikation der Literale (Zielaussagen)
   möglichst zuerst die instantiieren, die weniger Alternativen bieten

- die Stellen der „Nichtdeterminiertheit" sind potentielle Punkte für Backtracking
- Entscheidung durch Heuristiken

Planen als lineare Tiefensuche



Verbesserungen der linearen Planung

- Nichtlineare Planung
  - einzelne Teilpläne werden ineinander kombiniert
  - regressive Planung (Einschub von Operatoren in geeignete Stellen des Plans)
- Einschränkungsbasierte Planung (constraint-based)
  - frühzeitige Reduzierung der Wertebereiche von Variablen
  - Strategie der geringsten Festlegung (least-commitment strategy)
- Hierarchische Planung
  - Situationsabstraktion durch Bewertung von Aussagen von Situationen und Vorbedingungen
  - Operatorabstraktion