Universität Bielefeld

2 Problemlösen und Suche

5. Vorlesung: Means-Ends-Analysis, Diskriminations-netze; Goal Trees

Methoden der Künstlichen Intelligenz

Ipke Wachsmuth WS 2008/2009

Problemlösestrategien

REASONING FORWARD

("data driven",
"bottom up")

REASONING BACKWARD

("goal directed",
"top down")

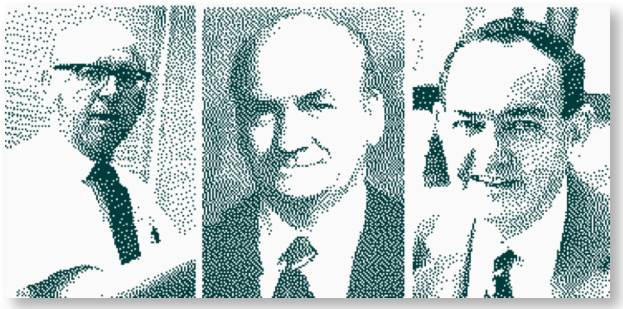
als Kombination von beiden:

MEANS-ENDS-ANALYSIS

General Problem Solver – GPS

(Newell, Shaw und Simon)

- berühmtes, aber heute „entmythisiertes“ Suchprogramm der frühen KI
- intendiert zur Modellierung menschlicher kognitiver Fähigkeiten bei Problemlösungen wie Puzzles und symbolischer Integration
- „allgemeine psychologische Theorie des Problemlösens“ (umstritten – oder eher unumstritten, dass nicht):
- eine Variante des allgemeinen Suchalgorithmus, die sowohl *Zustands-bewertung* als auch *Operatorordnung* zur Suchsteuerung verwendet.



Haupttechniken:
- **Means-Ends Analysis und „matching“**
- **Diskriminationsnetze**

Means-Ends Analysis

(„Mittel-Ziel-Analyse“)

Wie lässt sich die Anwendung von Operatoren "vorplanen", auch wenn sie in einem aktuellen Zustand (noch) nicht anwendbar sind?

- Grund für die Auswahl eines nächsten Operators kann sein, dass dadurch ein anderer Operator anwendbar wird („planning ahead“)
- wird benutzt, um nach und nach aus einer *partiellen* Anordnung von Operatoren (die von Zwischenzuständen zu anderen Zuständen führen) eine *totale* Anordnung von Operatoren (zwischen Startzustand und Zielzustand) zu erhalten.

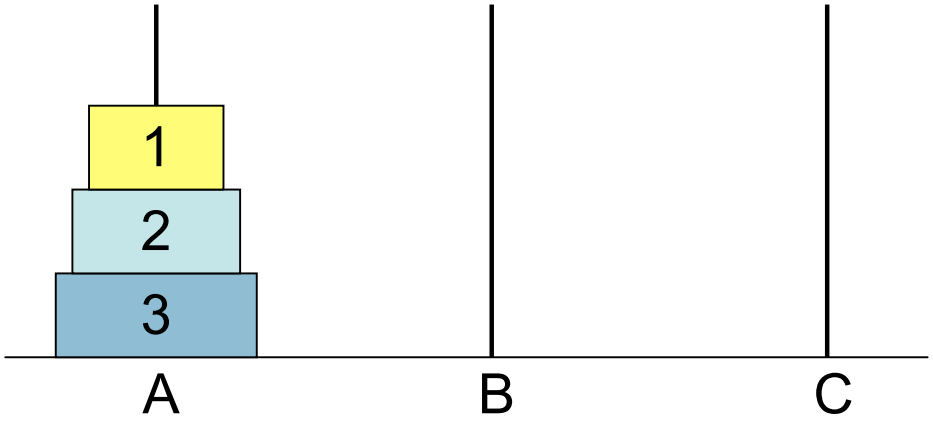
Differenzen-Abgleich

- **Aktuell explorierter Zustand im Suchraum wird mit Zielzustands-beschreibung abgeglichen (matching), um eine *Differenz* fest-zustellen**
  - keine Differenz: Zielzustand erreicht.
  - falls Differenz: Reduziere sie.
- **Differenzen**
  - dienen als grobes Maß für den Abstand vom Ziel
  - werden zur Auswahl erfolgversprechender Operatoren benutzt
- **Jeder Operator wird in drei Komponenten konzipiert:**
  - Vorbedingungen
  - Überführungsfunktion
  - Differenzen, die er reduziert

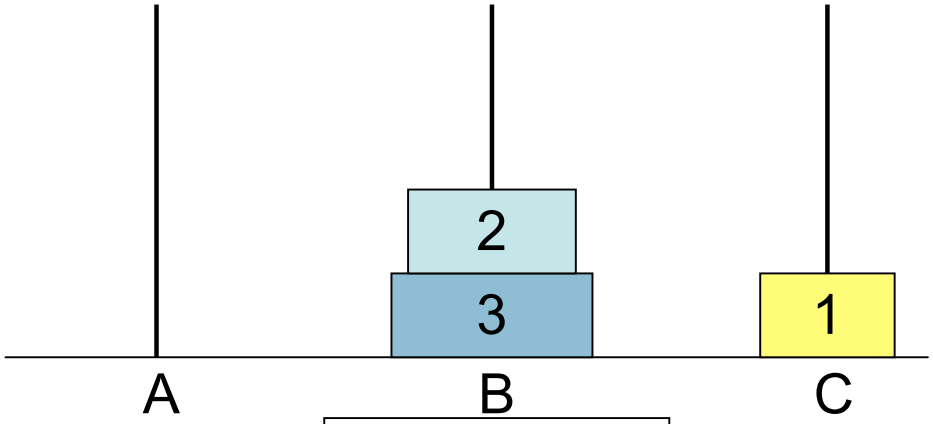
Beispiel:
„Die Türme von Hanoi“

„Towers of Hanoi“

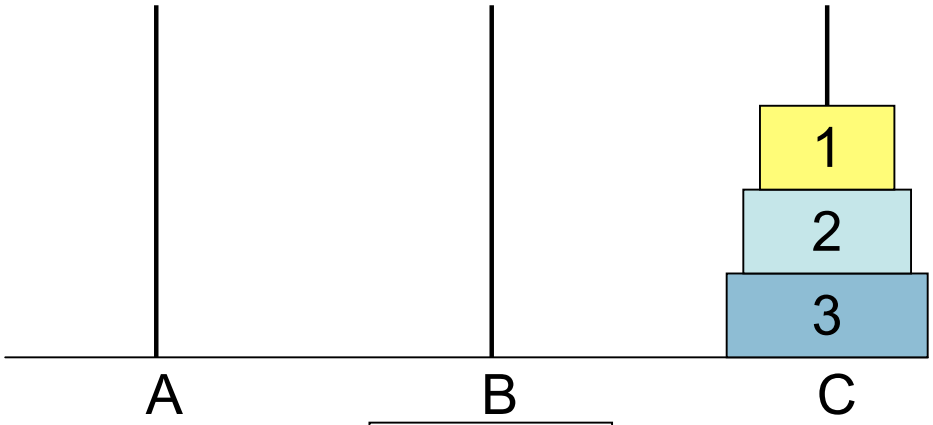
18 Operatoren des Typs move- iXY



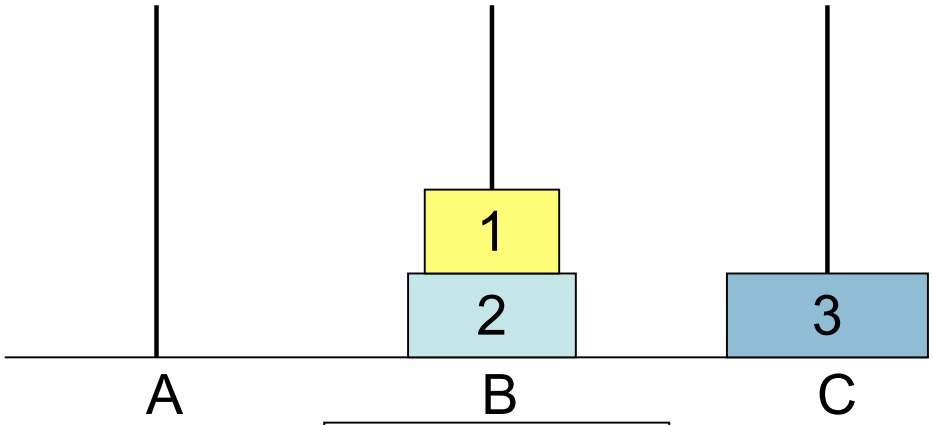
Startzustand



Zwischenzustand



Zielzustand



Zwischenzustand

Beispiel für „planning ahead“

GPS

Start: ((1 2 3) ( ) ( )) Ziel: (( ) ( ) (1 2 3))

Startzustand
- Alle drei Scheiben auf A

Zielzustand
- Alle drei Scheiben auf C

Differenz (matching) u.a.:
- Scheibe 3 auf A, nicht C

Differenztabelle:
- wird reduziert durch **move-3AC**

Vorbedingung von **move-3AC**:
- Scheiben 1 und 2 beide auf B: nicht erfüllt

Unterziel:
- Differenz des Startzustands zur Vorbedingung verringern

neue Differenz:
- Scheibe 1 auf A, nicht auf B
- Scheibe 2 auf A, nicht auf B

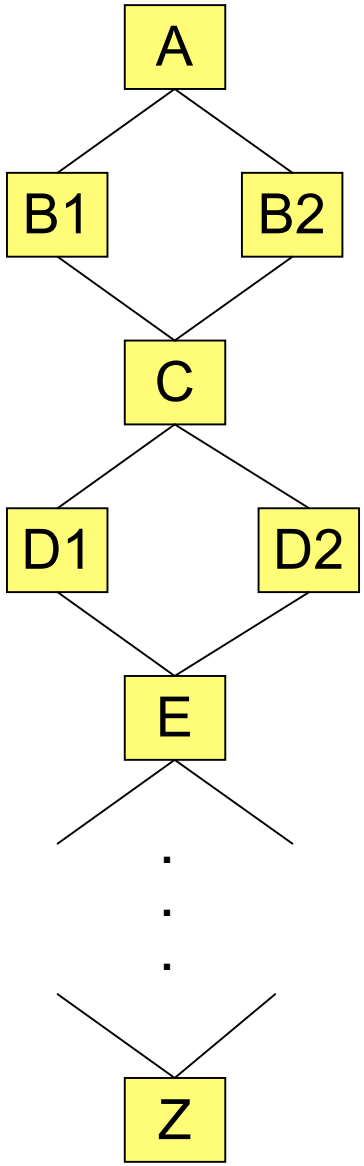
Differenztabelle:
- Scheibe 1 auf A, nicht auf B: wird reduziert durch **move-1AB**

Vorbedingung von **move-1AB**
- keine: erfüllt.

Führe **move-1AB** aus.

Wiederholte Zustände

Wie läßt sich eine Suche geschickt steuern, um wiederholte Zustände im Suchraum (wiederholt besuchte Knoten im Suchgraph) zu vermeiden?



Problem: **Bei einer Suche können verschiedene Operatorsequenzen zum gleichen Zustand im Suchraum führen.**

Betrachte **das Beispiel mit einer Depth-first Suche (Backtracking). Die Häufigkeit, mit der ein Knoten besucht wird, kann mit der Tiefe des Suchraums exponentiell wachsen.**

Konsequenz: **Der Knoten Z wird 2^n - mal besucht !**

Vermeiden wiederholter Zustände

- **alle explorierten Zustände in einer Tabelle abspeichern**
- **für jeden neu generierten Zustand nachsehen, ob er schon in der Tabelle ist**
- **falls ja: von diesem Zustand aus nicht (erneut) den Suchraum explorieren**

Der Aufwand für dieses Verfahren kann durch *O (log n)* beschränkt werden (n = Anzahl der tabellierten Zustände), und zwar mit Hilfe von

Diskriminationsnetzen.

Diskriminationsnetze

(erstmals so verwendet von Newell, Shaw und Simon im GPS)

Idee: Während der Suche wird simultan mit dem Suchbaum ein weiterer Baum aufgebaut, mit dem explorierte Zustände tabelliert ("indexiert") werden:

ANALOGIE: Bestimmungs-schlüssel in der Botanik

ACHTUNG: Ein Diskrimi-nationsnetz ist *keine* Repräsentation eines Suchraums!

(1) in den Blättern werden generierte Zustände eingetragen

(2) in nicht-terminalen Knoten werden Unterscheidungs-*gesichtspunkte* (Fragen) eingetragen

(3) die Kanten werden mit den *Unterscheidungsmerkmalen* (Antworten) gelabelt

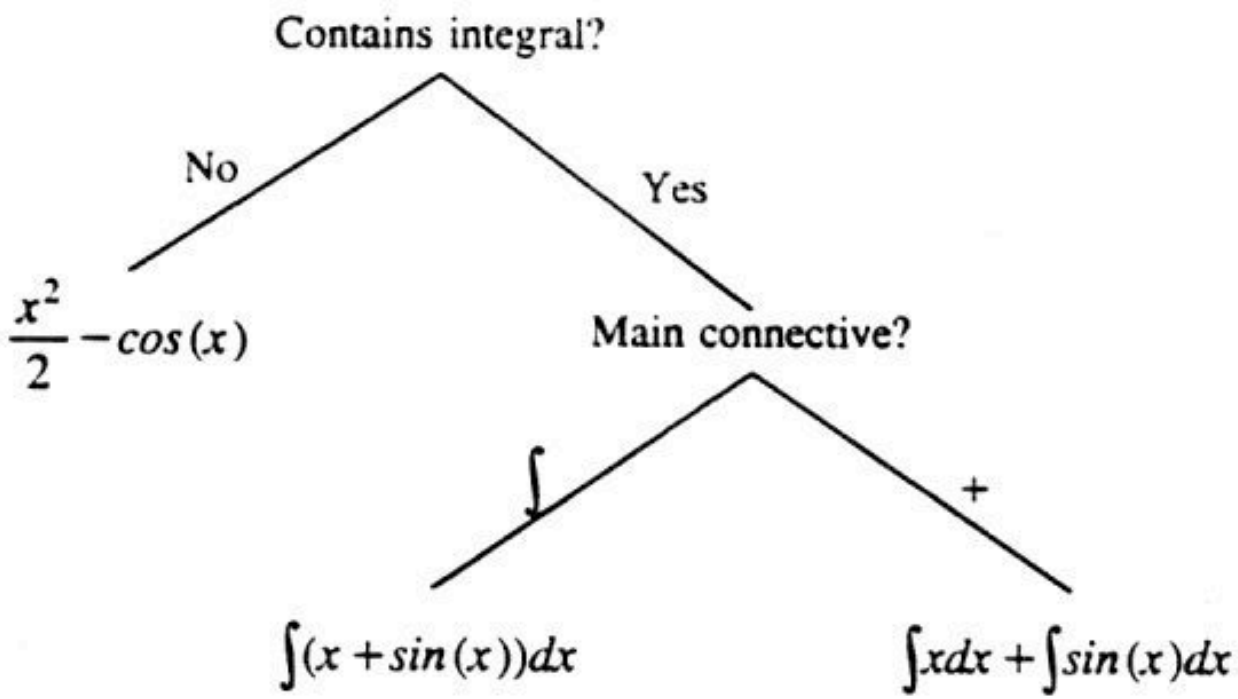
(4) kommt ein neuer Zustand hinzu, der sich von einem Blattknotenzustand in einem Merkmal unterscheidet, wird der Baum gemäß (2) und (3) erweitert.

Bsp.: Ein Diskriminationsnetz

Wird ein terminaler Knoten erreicht, vergleicht eine *Matching*-Prozedur den dort tabellierten Zustand mit dem fraglichen aktuellen Zustand des Suchraums:

falls = : Zustand nicht weiter explorieren

falls ≠ : Unterscheidungsmerkmal feststellen, Baum erweitern, aktuellen Zustand weiter verfolgen

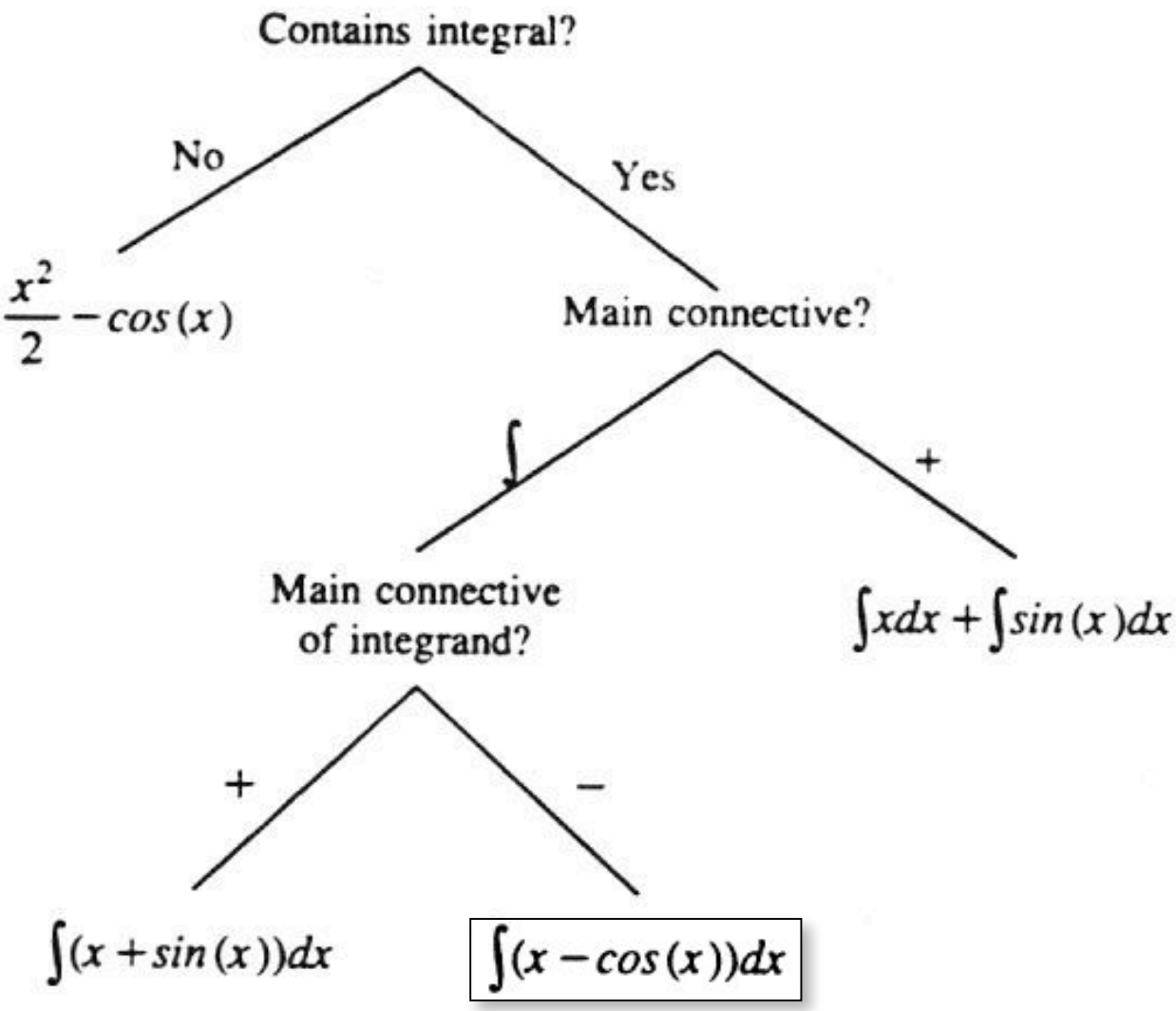


$$\int(x - cos(x))dx$$

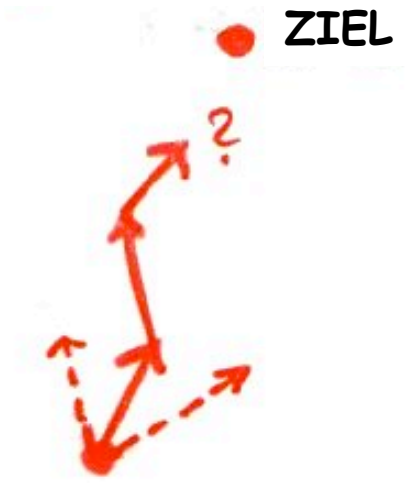
Diskriminationsnetz erweitert

Bemerkungen:

1. Jede neue Unterscheidung betrifft immer nur ein diskriminierendes Merkmal (das "nächstzugängliche").
2. Erforderlich ist eine Art Parser für die Zustandsbeschreibung (etwa für algebraische Ausdrücke).



Problem-Repräsentation: 2 Arten



- **Vorwärtssuche**: bottom-up

  jede Operator-Anwendung erzeugt nur ein neues Objekt (einen neuen Zustand im "state space")

  ZUSTANDSRAUM-REPRÄSENTATION

- **Rückwärtssuche**: top-down (goal-directed)
  - 1. Fall: jeweils ein neues Objekt ("subgoal")
    Beispiel: LOGIC THEORIST
  - 2. Fall: eine Operator-Anwendung erzeugt mehrere Unterziele, die in Konjunktion zu lösen sind

  PROBLEMREDUKTIONS-REPRÄSENTATION

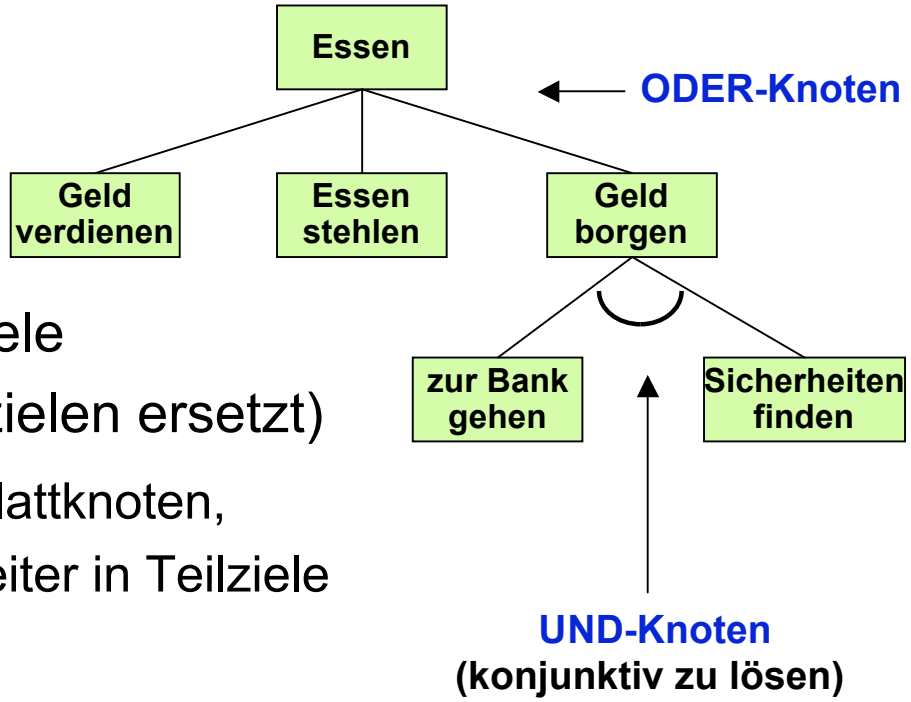
Suche in UND-ODER-Bäumen

Spezialfall des allgemeinen Suchproblems:

- Erfüllung eines Ziels, das sich in Teilziele zerlegen lässt.
- Gibt es mehrere Methoden, die Erfüllbarkeit von (Teil-)zielen zu prüfen, suche eine erfolgreiche.

Zwei Knotentypen:

- ODER-Knoten: alternative Methoden
- UND-Knoten: simultan zu erfüllende Ziele
  (Ziel wird durch Konjunktion von Teilzielen ersetzt)
  - SUCCESS-Knoten („Erfolgsknoten“): Blattknoten, das sind UND-Knoten, die sich nicht weiter in Teilziele aufspalten lassen



Erweiterung: Goal Tree (Zielbaum)

Goal Tree
=
UND-ODER-Baum
+
CONSTRAINTS

Dieser Typ von Suche ist ein Spezialfall der Problemreduktions-Suche (problem reduction search)

In Goal Trees formulieren sog. CONSTRAINTS

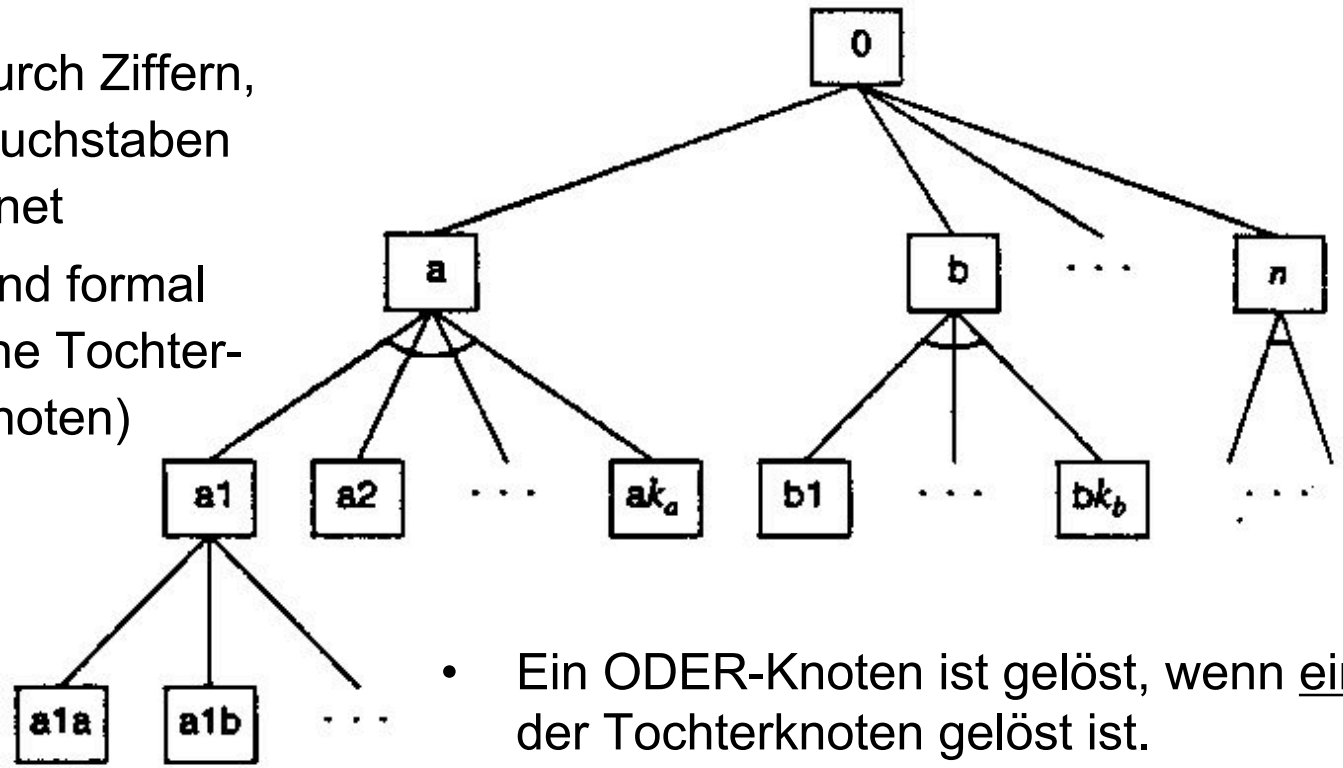
- **Randbedingungen an die Lösung eines (nonsuccess-)UND-Knotens, die**
- **von den in Konjunktion zusammenzusetzenden Teillösungen des UND-Knotens gemeinsam erfüllt werden müssen.**

Zum Beispiel könnte gefordert werden:

- **dass die Summe der Kosten der Teillösungen durch einen Wert beschränkt ist**
- **dass die Variablenbindungen gleich benannter Variablen in den Teillösungen auch tatsächlich gleich sind, etc.**

Goal Tree – allgemeine Struktur

- ODER-Knoten hier durch Ziffern, UND-Knoten durch Buchstaben (hinten) gekennzeichnet
- SUCCESS-Knoten sind formal UND-Knoten, die keine Tochter-knoten haben (Blattknoten)
- Jeder nonsuccess-UND-Knoten kann einen zugeordneten CONSTRAINT haben (sonst: „purer“ UND-ODER-Baum)

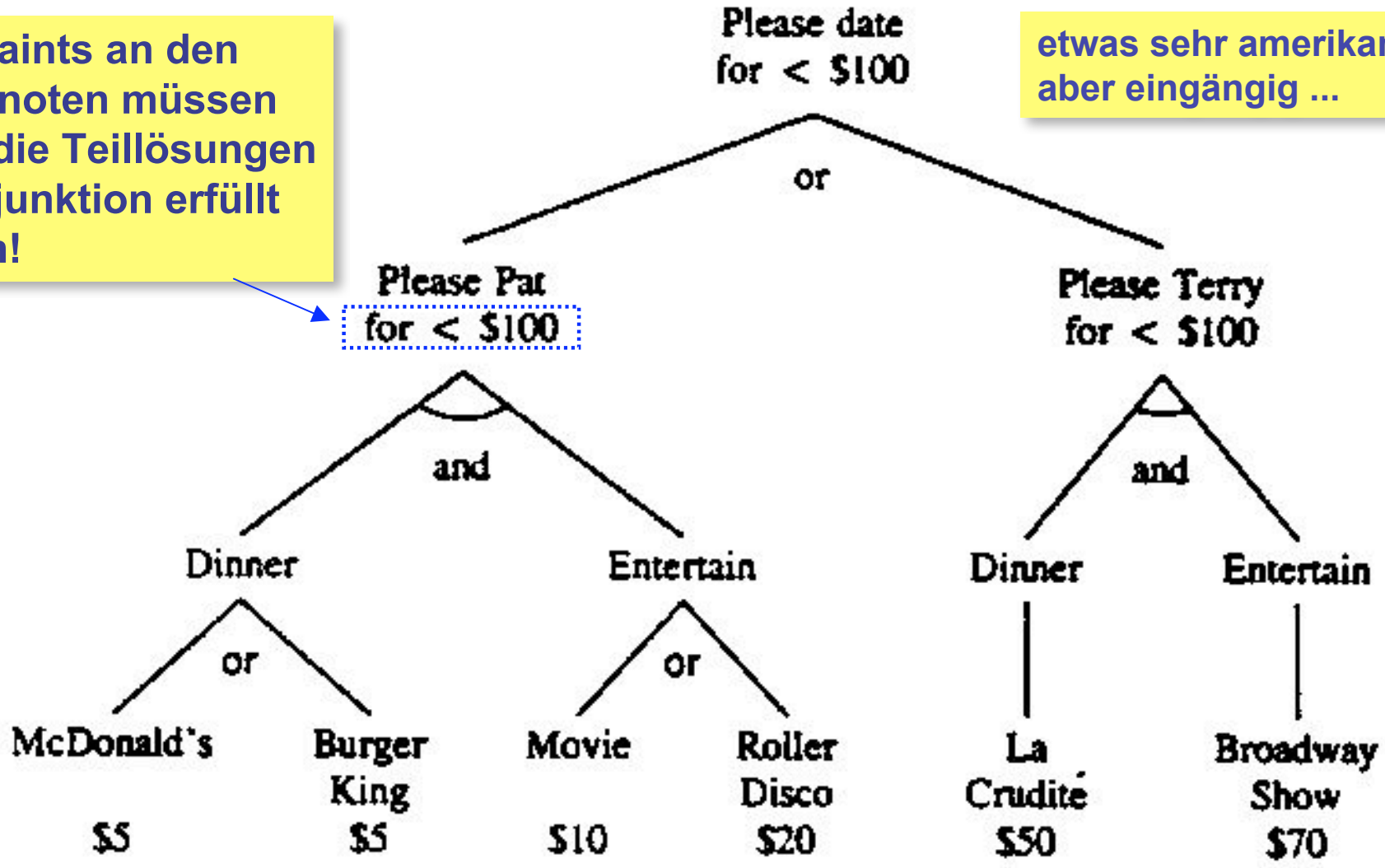


- Ein ODER-Knoten ist gelöst, wenn <u>einer</u> der Tochterknoten gelöst ist.
- Ein UND-Knoten ist gelöst, wenn <u>alle</u> Tochterknoten so gelöst sind, das sein CONSTRAINT insgesamt erfüllt ist.
- Der Goal Tree ist gelöst, wenn sein Wurzelknoten (das Ziel) gelöst ist.

Ziele und Teilziele am Beispiel

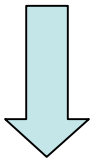
Constraints an den UND-Knoten müssen durch die Teillösungen in Konjunktion erfüllt werden!

etwas sehr amerikanisch aber eingängig ...



Suchverfahren für Goal Trees

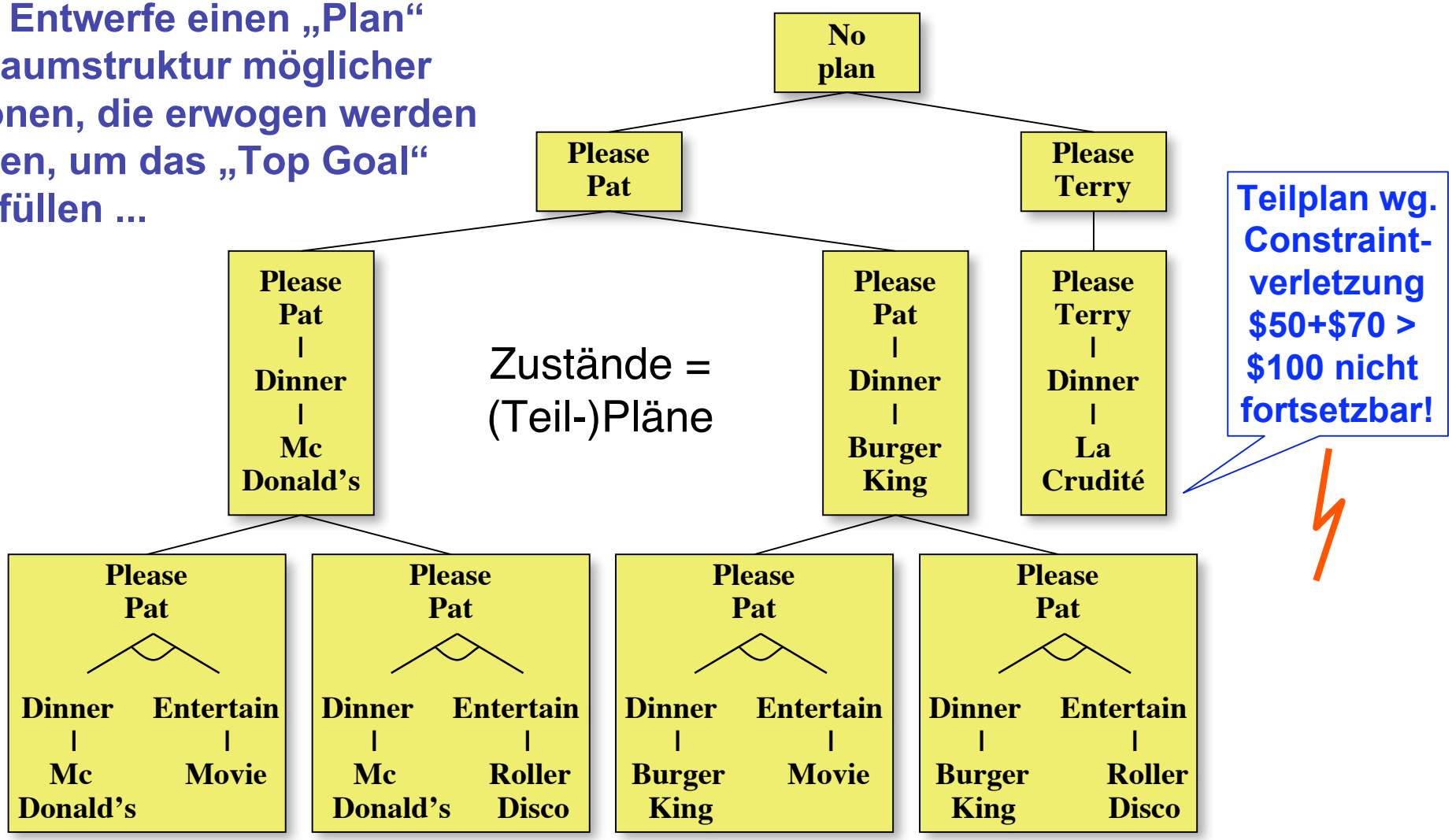
- Problem:
  - Nach dem bisher Gesagten ist Goal-Tree-Suche nicht ohne weiteres ein Spezialfall des allgemeinen Suchalgorithmus:
  - dieser untersucht nur Alternativen (ODER-Knoten), keine simultan zu erfüllenden Teilziele (UND-Knoten)!
- Idee:
  - Zu einem gegebenen Goal Tree wird ein Suchraum als Raum von Teillösungen des Goal Trees definiert.
  - Die Töchter eines Knotens sind „Fortsetzungen“ des Knotens, bei denen genau eine weitere Festlegung getroffen wird.



am Beispiel...

Pläne für Goal-Tree-Lösungen

Idee: Entwerfe einen „Plan“ als Baumstruktur möglicher Aktionen, die erwogen werden können, um das „Top Goal“ zu erfüllen ...

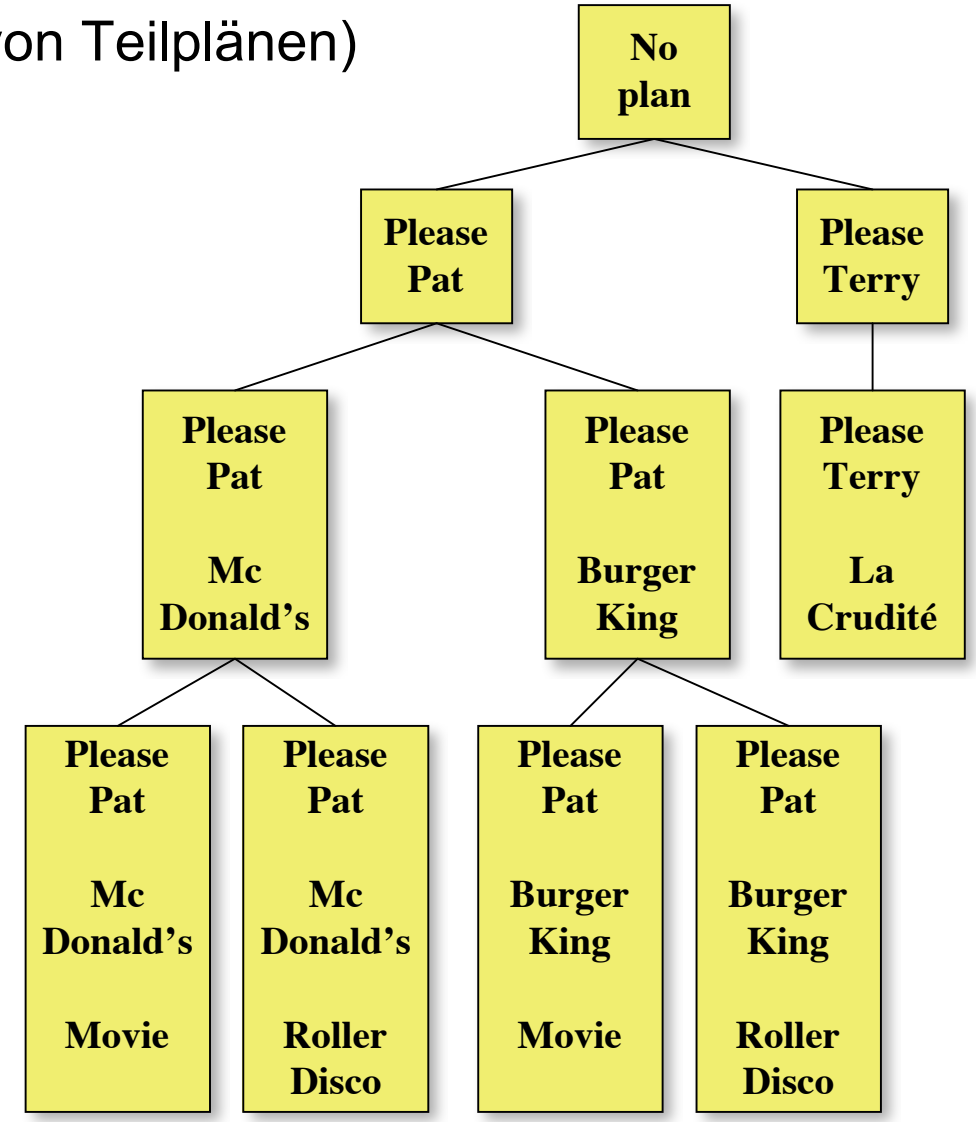


und redundanzfrei:

(Suchraum von Teilplänen)

ACHTUNG:

- Es gibt alternative mögliche Extensionen der UND-Knoten, die aber keine echt verschiedenen Pläne erbringen.
- Die Reihenfolge der Generierung von Teilplänen kann jedoch die Größe des Suchraums beeinflussen!



Lösungen eines Goal Trees

Teillösung eines Knotens *n*:

– ein Teilbaum mit Wurzel *n*, der, falls er einen ODER-Knoten *m* enthält, auch genau einen der Tochterknoten von *m* enthält

Vollständige Lösung eines Knotens *n*:

– eine Teillösung von *n*, die, falls sie einen UND-Knoten *m* enthält,
  (1) auch alle Tochterknoten von *m* enthält und
  (2) deren Lösungen ferner *C(m)*, dem Constraint an *m*, genügen

Lösung eines gesamten Goal Trees:

– eine vollständige Lösung seiner Wurzel *0*

Fortsetzung *s2* (Extension) einer Teillösung *s1*:

– falls *s2* ein Baum mit allen Knoten von *s1* ist und zusätzlich 0 oder mehrere Knoten mit weiteren Festlegungen enthält

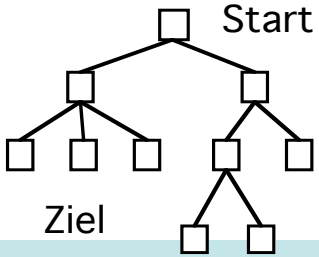
Bemerkungen zu Goal Trees

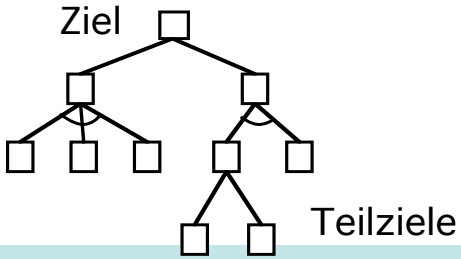
- Jeder Goal Tree kann in einen normalen Suchraum alternativer Möglichkeiten umformuliert werden!
- Ein Goal Tree ist ein abstraktes Objekt, das einen Alternativenraum „aufspannt“ und in der Regel nicht explizit im Computer repräsentiert ist.
- Denkweise für Teillösungen im zugeordneten Suchraum:
  - bisher gemachte Festlegungen
  - noch mögliche Festlegungen (eingeschränkt durch die bisherigen)

Beispiel: Färben einer Landkarte

Bei jedem Problemlösungszustand hat man eine Menge bereits gefärbter und eine Menge noch zu färbender Länder, deren Färbemöglichkeit durch bereits getroffene Wahlen eingeschränkt ist.

Übersicht: Suchbaum vs. Goal Tree





Suchbaum

- Suchproblem: charakterisiert durch Startzustand und Zielzustandsbeschreibung (goal-state description)
- Operatoren: können einen Zustand in einen anderen transformieren
- Suchraum: Menge aller Zustände, die man durch Anwendung einer Folge von Operatoren erreichen kann

Goal Tree

- beschreibt eine Situation, in der ein Ziel durch das Lösen von Teilzielen erreicht werden kann
- jedes Teilziel kann aus mehreren Teilen bestehen, die konsistent zu lösen sind
- Zwei Knotentypen:
  ODER: für Alternativen
  UND: für simultane Teilziele
- jeder nonsuccess-UND-Knoten kann eine Randbedingung an die Lösung haben (constraint)

Bemerkungen zu Spielbäumen

- eine weitere Anwendung von Goal Trees
- Spielbaum = Suchraum alternativer Spielverläufe

Aufgabe: Finde vollständige Gewinnstrategie

Grundlegende Ideen:

- Bewertungsfunktion (static board evaluator)
- look-ahead-depth
- Minimax search
- Alpha-Beta-Pruning

(Frage jeweils: Was kann bei der Suche unexploriert bleiben, da garantiert irrelevant?)

Leseempfehlung heute:

- Charniak & McDermott, Kapitel 5, Seite 270-306

zu Gameplaying auch:

- Russell & Norvig, Kapitel 5, Seite 122 ff