Universität Bielefeld

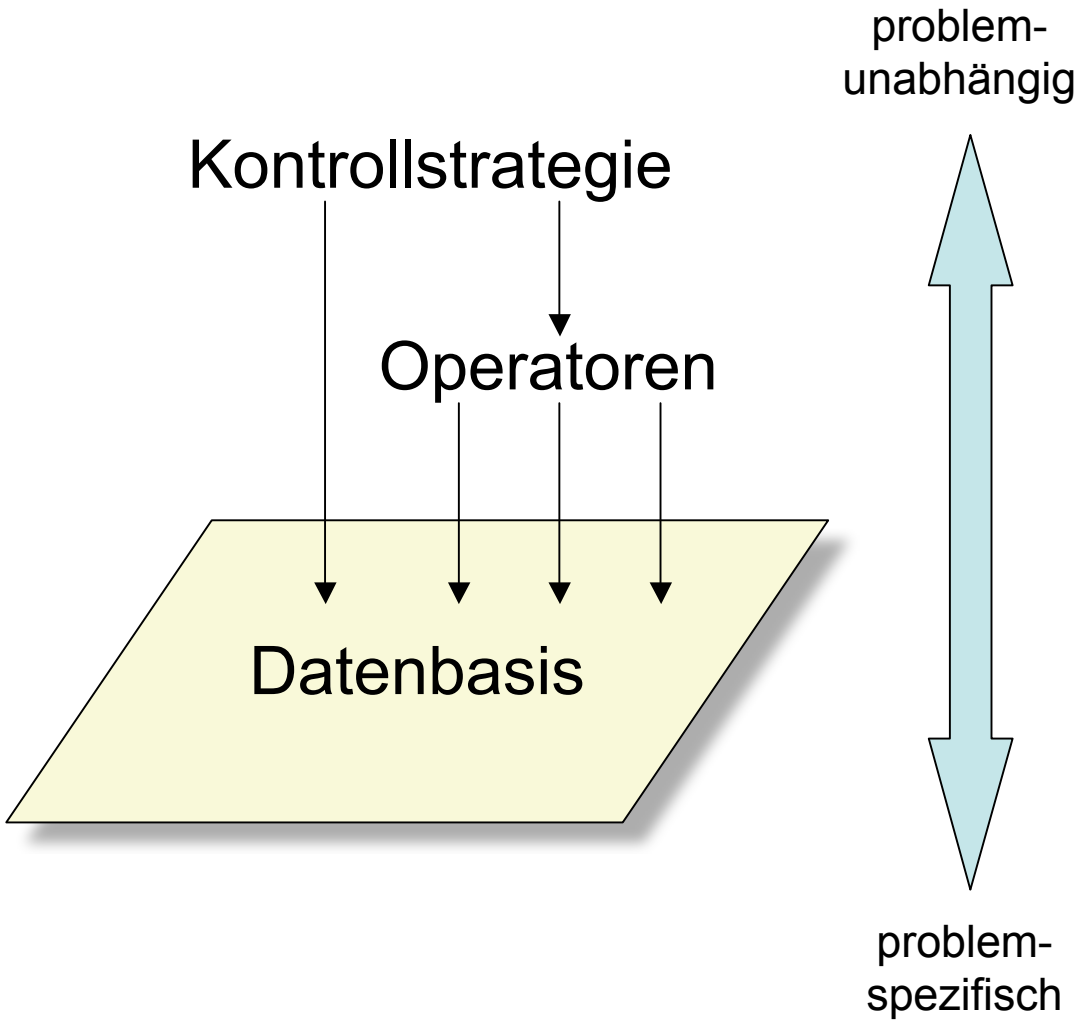
2 Problemlösen und Suche

6. Vorlesung: Constraint Satisfaction, Planen, wissensgestützte Suche

Methoden der Künstlichen Intelligenz

Ipke Wachsmuth WS 2008/2009

Kontrollstrategien für die Suche



(Teil-)Übersicht:

- Breitensuche
- Tiefensuche
- Operatorordnung
- Zustandsbewertung
- Bestensuche, A*-Algorithmus
- Means-Ends Analysis
- Problemreduktion (Goal Trees)
- constraint satisfaction
- wissensgestützte Suche

Constraint Satisfaction Problem

- Constraints („Beschränkungen", „Randbedingungen") sind ein häufig verwendetes Darstellungsmittel, vor allem in den Bereichen Suche und →Planung.
- Ein Constraint Satisfaction Problem (CSP) ist ein spezieller Problemtyp:
  - die **Zustände** des Suchraums sind durch Werte einer Menge von Variablen definiert
  - der **goal-test** enthält eine Menge von Constraints, denen die Werte genügen müssen

Eine einfache Form der Constraint Satisfaction (Constraint-Erfüllung) wurde bereits bei den Goal Trees betrachtet.

Constraints

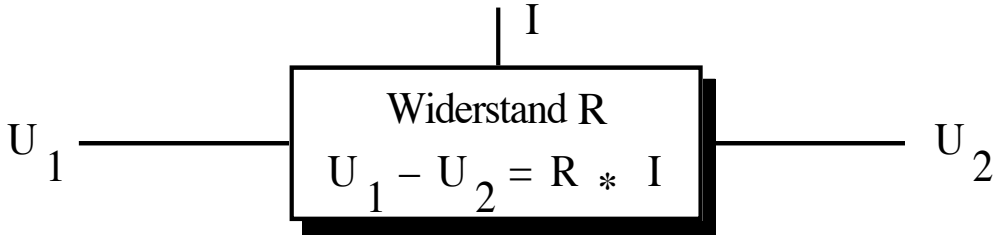
- dienen zur Repräsentation von Relationen, das heißt Beziehungen zwischen Variablen
- besonders zur Darstellung lokaler Randbedingungen, die eine Problemlösung erfüllen muss
  - Beispiel: Wunsch eines Lehrers bei der Stundenplan-Planung: einen Tag pro Woche frei
  - durch jedes neue Constraint wird der Lösungsraum weiter eingeschränkt
- Ziel des CSP: Finden einer Lösung unter Beachtung aller Constraints

Lesehinweis hierzu:

- Puppe, Einführung in Expertensysteme, Kap. 6

Uridee: Schaltkreis-Simulation

Im Constraint-System EL wird z.B. elektrischer Widerstand als Constraint zwischen Eingangs-, Ausgangsspannung und Strom repräsentiert:



- Alle Komponenten eines Schaltkreises sind durch Constraints repräsentiert und durch gemeinsame Variablen zu einem Constraint-Netz verbunden.
- Bekannte Werte werden durch das Constraint-Netz propagiert, so dass für jede Variable ein Wert berechnet wird.

Vergleich Constraint/Operator

- Constraints repräsentieren ungerichtete Zusammenhänge zwischen Variablen (z.B. $U = R * I$), Operatoren gerichtete.
- Constraints sind mächtiger als Operatoren (schließen die Möglichkeiten von mehreren Operatoren ein), aber in der Realisierung aufwändiger.
- Constraints sind vorzüglich zur Modellierung vernetzter Systeme geeignet und beschleunigen dort die Suche (Abbruch bei Constraint-Verletzung).

Constraint-Propagierung

Ein Constraint kann häufig als mathematische Gleichung (oder Ungleichung) betrachtet werden und ein Constraint-Netz als (Un-) Gleichungssystem, auch für nichtnumerische Zusammenhänge.

- Ausrechnen des Systems durch Constraint-Propagierung:

  Beschränkungen der Wertemengen von Variablen werden über die mit ihr verbundenen Constraints an andere Variablen weitergegeben, bis keine Einschränkung des Wertebereichs mehr möglich ist.

  – Genau ein Wert pro Variable: eindeutige Lösung
  – Wertemengen für Variablen: multiple Lösungen
  – Leere Wertzuweisung für eine Variable: Inkonsistenz

Propagierungsalgorithmen

- EINGABE: Constraint-Netz und Teilbelegung von Variablen mit Werten
- AUSGABE: eine mit den Constraints konsistente Wertzuweisung an weitere Variablen

- Propagierungsalgorithmen unterscheiden sich danach, was entlang einer Variablen propagiert werden kann:
  - nur feste Werte, z.B. X = 5
  - Wertemengen, z.B. {3,4,5,6}
  - symbolische Ausdrücke, z.B. X = 2Y
- Wichtig ferner: ob Maßnahmen für Fallunterscheidungen und Revision von Ableitungen zu treffen sind.

CSP: Ein nettes Beispiel

(Newell & Simon)

```
  SEND
+ MORE
 MONEY
```

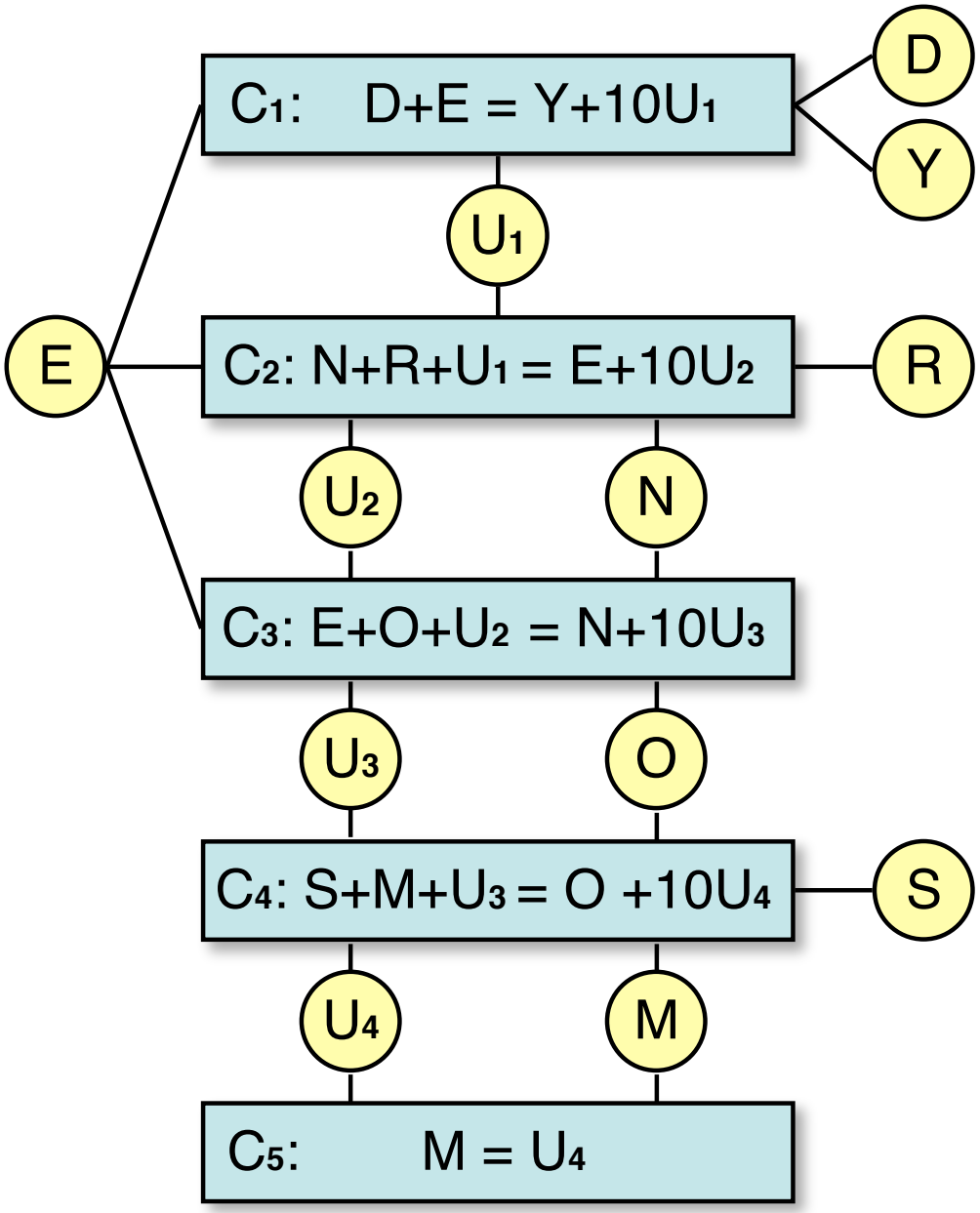
- Belegung aller Variablen mit verschiedenen Werten finden, so dass die Gleichung aufgeht.
- Elegante Formulierung des Problems als Constraint-Netz benutzt zusätzliche Variablen U_1, U_2, U_3, U_4 für Überträge der Spaltensummen.
- Wertebereiche der Variablen:

$D,E,N,O,R,Y \in \{0,1,2,3,4,5,6,7,8,9\}$

$M,S \in \{1,2,3,4,5,6,7,8,9\}$

$U_1, U_2, U_3, U_4 \in \{0,1\}$

Constraint-Netz mit 5 Constraints (C_i)



SEND
+ MORE
MONEY

CSP: Formale Charakterisierung

- Constraint als Tripel: (Name, Variablenmenge, Relation)

  z.B. $(C_1 , \{D,E,Y,U_1\} , D+E = Y +10U_1)$

- CSP gegeben durch:
  1. Menge von Constraints, die durch gemeinsame Variablen verbunden sind
  2. Anfangsbelegung von einigen Variablen

- Lösung des CSP:

  maximale Einschränkung aller Variablen

Propagierungs-algorithmen für symbolische Ausdrücke sind allerdings sehr aufwändig!

Propagierungsalgorithmus

(ein einfacher: nur für feste Werte und Wertemengen)

Eingabe: Constraint-Netz und Anfangsbelegung
Ausgabe: Maximale konsistente Wertzuweisung zu allen Variablen

1. Weise allen Variablen ihre Anfangsbelegung zu;
2. Aktiviere alle Constraints, in denen die Variablen vorkommen, und setze sie auf die Liste AKTIV;
3. Wähle ein Constraint aus AKTIV aus und deaktiviere es; falls AKTIV leer: STOPP (Ergebnis = aktuelle Variablenbelegung);
4. Berechne den neuen Wertebereich für die an dem ausgewählten Constraint beteiligten Variablen. Falls die Wertemenge einer Variable leer ist: STOPP (Inkonsistenz);
5. Falls die Wertemenge beschränkt wurde, setze alle Constraints, die auf die Variablen Bezug nehmen, auf AKTIV;
6. GOTO 3;

Definition von Planung

- Problembereich: Zustandsraum und Operatoren
- Planungsproblem: Anfangs- und Zielzustand
- Lösung: Folge von Operatoren zur Überführung des Anfangszustandes in den Zielzustand

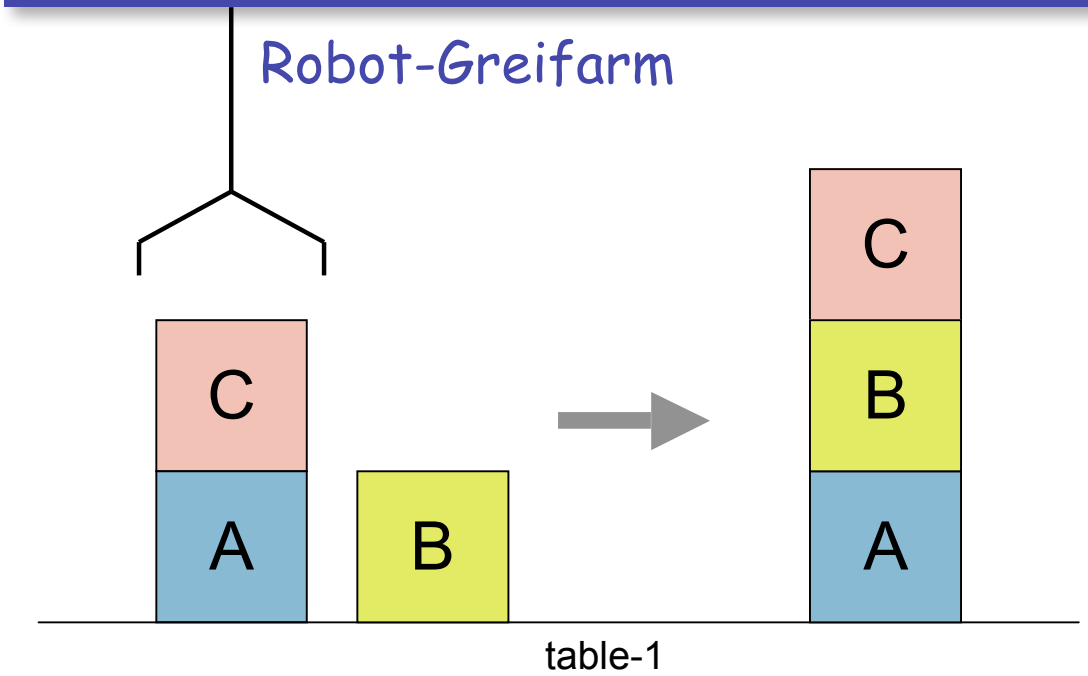
im Grundansatz zunächst:
klassisches Suchproblem

zu Suche und Planung:

- eigentlich alle KI-Bücher im Semesterapparat

Zum Nachschlagen auch:
Handbook of Artificial Intelligence, Volume 1, Kap. II – „Search“ oder Russell & Norvig, Teil II

Planung: Beispielproblem



Einfache Klötzchenwelt

Gegeben: Anordnung von Klötzchen, Zielanordnung und mögliche Roboteraktionen.

Gesucht: Aktionenfolge zur Herstellung der Zielanordnung.

Anfangszustand	→	Zielzustand
On(A,table-1) On(B,table-1) On(C,A)		On(A,table-1) On(B,A) On(C,B)

Operatoren:
(1) pickup (*block*)
(2) putdown (*block*)
(3) stack (*block1*,*block2*)
(4) unstack (*block1*,*block2*)

Weitere Beispielprobleme

Stundenplanerstellung

- Gegeben: Räume, Lehrer, Schulklassen, Fächer, Zuordnungen
- Gesucht: Stundenplan

Computer-Konfigurierung

- Gegeben: Hauptkomponenten
- Gesucht: Layout und Hilfs-komponenten

Planung von Experimenten in der Molekulargenetik

- Gegeben: Substanzen und Organismen, durchführbare Aktionen, Ziel des Experiments (z.B. Insulin herstellen)
- Gesucht: Sequenz von Aktionen und Spezifikation der benötigten Substanzen und Organismen

Ein- und mehrstufiges Planen

Einstufiges (nichthierarchisches) Planen:

- systematisch alle Sequenzen von Operatoren durchprobieren, bis der Zielzustand erreicht ist
- Gefahr der kombinatorischen Explosion!

siehe hierzu: Konstruktion

- Puppe, Einführung in Expertensysteme, Kap.11

Mehrstufiges (hierarchisches) Planen:

- zunächst Planung einer Sequenz von abstrakten Operatoren (Grobplan), die dann schrittweise verfeinert werden.
- Hauptproblem: Nebeneffekte eines Operators können Anwendung eines anderen verhindern:

  Interaktion von Operatoren!

Interaktion von Operatoren

Beispiel: „Streiche die Leiter und die Decke“

Decke-Leiter-Problem: Aufgabe ist, die Decke und die Leiter zu streichen. Dabei muss die Decke zuerst gestrichen werden, da die ungestrichene Leiter Voraussetzung dafür ist, dass man die Decke ohne längere Wartezeit streichen kann.

Nichtlineares Planen:

- explizite Repräsentation von Interaktionen als Constraints zwischen Operatoren und Objekten
- Festlegung der Operator-Reihenfolge erst dann, wenn alle Constraints erfüllbar sind

Verfeinern von Grobplänen

- Differenzanalyse (means-ends analysis)
  - Grobplan direkt aus Aufgabenstellung generiert durch Bewertung der Differenz zwischen Ausgangs- und Zielzustand; Auswahl von Operatoren zur Verringerung der Differenz
- Phasenaufteilung
  - Benutzung eines einzigen Standardplans für einen gegebenen Problembereich (Beispiel: Computer-Konfigurierung)
- Skelettplanen (wissensbasiertes Planen)
  - allgemeinen Plan aus einer Bibliothek vorgegebener Skelettpläne auswählen; Beispiel folgt ...

Skelettplanen in MOLGEN

„Molecular Genetics“

Ziel: Planung eines Experiments zur Feststellung von „Intrastrang-Komplementarität“ in einer DNA (d.h. ob ein DNA-Strang komplementäre Teilstücke enthält)

1. Auswahl des Skelettplans: Für „Intrastrang-Komplementarität“ ist der angegebene Skelettplan vorhanden
2. Verfeinerung: Die einzelnen Operatoren werden stufenweise präzisiert (Ergebnis auf der rechten Seite)

	1. Skelettplan	2. Verfeinerung
(1)	Denaturiere (spalte die Stränge der DNA)	Erhitze die Probe
(2)	Renaturiere (Umkehrung der Denaturierung)	Schnelles Abkühlen der Probe
(3)	Zerstöre alle einsträngige DNA	Benutze Enzym „S1-Nuclease“
(4)	Finde die restliche DNA	Benutze Technik „Gel-Filtration“

[Quelle: Puppe]

Einordnende Bemerkungen

Suche ist eine Grundmethode für KI-Programme und wird in der einen oder anderen Form häufig benötigt.

Suche ist nicht alles:

- **"Allgemeinste" Suchprogramme haben sich in der Praxis als nicht fruchtbar erwiesen, außer für "genuine" Such-probleme (wie Puzzles).**
- **Die meisten Such-Heuristiken sind in ihrer Nützlichkeit auf bestimmte Problemklassen beschränkt.**

Ein sehr wichtiges Einsatzfeld für Suche ist der Bereich "Planen".

Das große Gewicht, das dem Feld "Suche" (genauer: explorative Suche) in der KI zugeschrieben wird, hängt mit den frühen Themenstellungen der KI zusammen: Problem Solving, GPS etc.

Jedoch gilt:

Warum Puzzles und ähnliche Aufgaben nicht so spannend für das "Problemlösen" sind

- **Sie sind bereits als Suchaufgaben gestellt**
  (die dann zum Teil auch extrem schwierig zu lösen sind).

- **Wichtiger ist es aber, zu <u>erkennen</u>, wie sich die mit einem Problem gestellten Fragen auf Suchaufgaben abbilden lassen.**

Rückblick auf Teil 2

1. Allgemeine Suchprobleme mit Suchraum, Operatoren, Zustands-bewertungsfunktion und weiteren Heuristiken
2. Means-Ends-Analysis, wo man die Zustände eines Suchraums mit Zielbeschreibungen abgleicht, um anhand der Differenz nächst-anwendbare Operatoren zu "raten"
3. Goal-Tree-Suche, wo Zustände in Bezug auf zu machende Fest-legungen analysiert werden, die bestimmten Constraints genügen.
4. Minimax-Suche, wo spezielle Goal-Trees zur Modellierung von Zwei-Personen-Spielen benutzt werden.
5. Constraint Satisfaction und planende Suche
6. [kontinuierliche Suchprobleme (Optimierungsaufgaben, Suche nach lokalen/globalen Minima und Maxima, "hill climbing" etc.) ...]

Ausblick: Wissensgestützte Suche

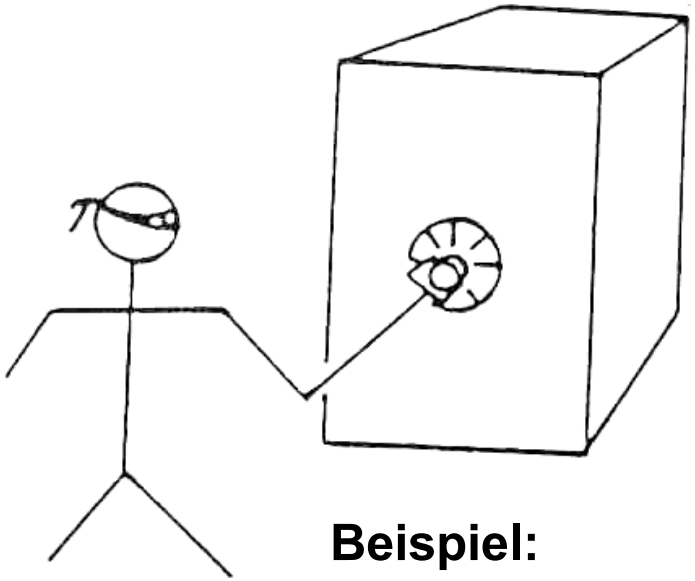
Problemlösung basiert auf Suche --

- "Schwache Methoden" für Suche wie generate-and-test sind für komplexe Suchräume nicht adäquat;
- Menschen benutzen *heuristisches Wissen,* um die Suche einzuschränken (aus Wissen über den Bereich abgeleitetes Kontrollwissen für die Suche);
- Ein wissensbasiertes System ist ein Problemlösesystem, in dem <u>explizites</u> Wissen* eines menschlichen Problemlösers eingebettet ist, um die Komplexität des Suchraums zu kontrollieren.

**Das Wissen muss man aber erst einmal haben!*

Idee wissensbasierter Systeme

Idee: Explizites Wissen über einen Aufgabenbereich verkürzt Problemlösesuche.



Beispiel:
Safe knacken

69–32–99

Blinde Suche: Alle Kombinationen probieren
=> 1 000 000 (100x100x100) Versuche

Wissensgestützte Suche:

Wenn man weiß, dass für einen Safetyp 2 Pos. links oder rechts auch noch klappt
=> 8000 (20x20x20) Versuche

Wenn man weiß, dass die Safes der Firma X bei Auslieferung 25–0–25 oder 50–25–50 eingestellt haben und dass 20% der Käufer versäumen, die Kombination zu ändern
=> in 20% reichen zwei Versuche!

(nachzulesen bei R.P. Feynman: "Sie belieben wohl zu scherzen, Mr. Feynman!")