Sokoban

Knowledge Engineering und Lernen in Spielen

TECHNISCHE UNIVERSITÄT DARMSTADT

Mark Sollweck

Überblick

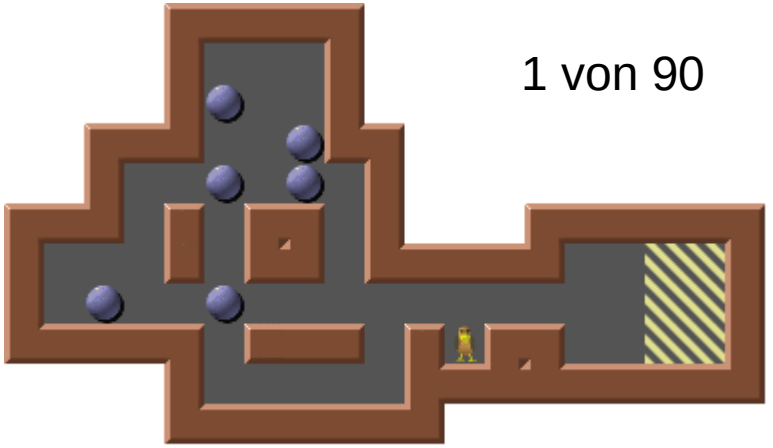
TECHNISCHE UNIVERSITÄT DARMSTADT

- Sokoban
  - Spielregeln
  - Eigenschaften
- Lösungsansatz
  - IDA* Suche
  - Anwendungsspezifische Verbesserungen
  - Ergebnisse
- Zusammenfassung
- Quellenangabe

Sokoban
Spielregeln

TECHNISCHE UNIVERSITÄT DARMSTADT

- Spielfeld
  - Freie Felder
  - Blockierte Felder
  - 1 Spielfigur
  - „Kisten“
  - Zielfelder



1 von 90

- Ziel: Bewegen aller Kisten auf Zielfelder

- Kisten können nur geschoben werden (nicht gezogen)

Sokoban Eigenschaften

TECHNISCHE UNIVERSITÄT DARMSTADT

- Züge sind nicht immer umkehrbar
- NP schwer
- PSPACE vollständig
- Suchbaum wird auf 20 Millionen Knoten begrenzt
- Minimierung der Kistenbewegungen

Property	Specifics	24-Puzzle	Rubik's Cube	Sokoban
Branching factor	Average	2.37	13.35	12
	Range	1–3	12–15	0–136
Solution length	Average	100+	18	260
	Range	1-unknown	1–20	97–674
Search-space size	Upper bound	10^{25}	10^{19}	10^{98}
Calculation of lower bound	Full	$O(n)$	$O(n)$	$O(n^3)$
	Incremental	$O(1)$	$O(1)$	$O(n^2)$
Underlying graph		Undirected	Undirected	Directed

Lösungsansatz IDA* Suche

TECHNISCHE UNIVERSITÄT DARMSTADT

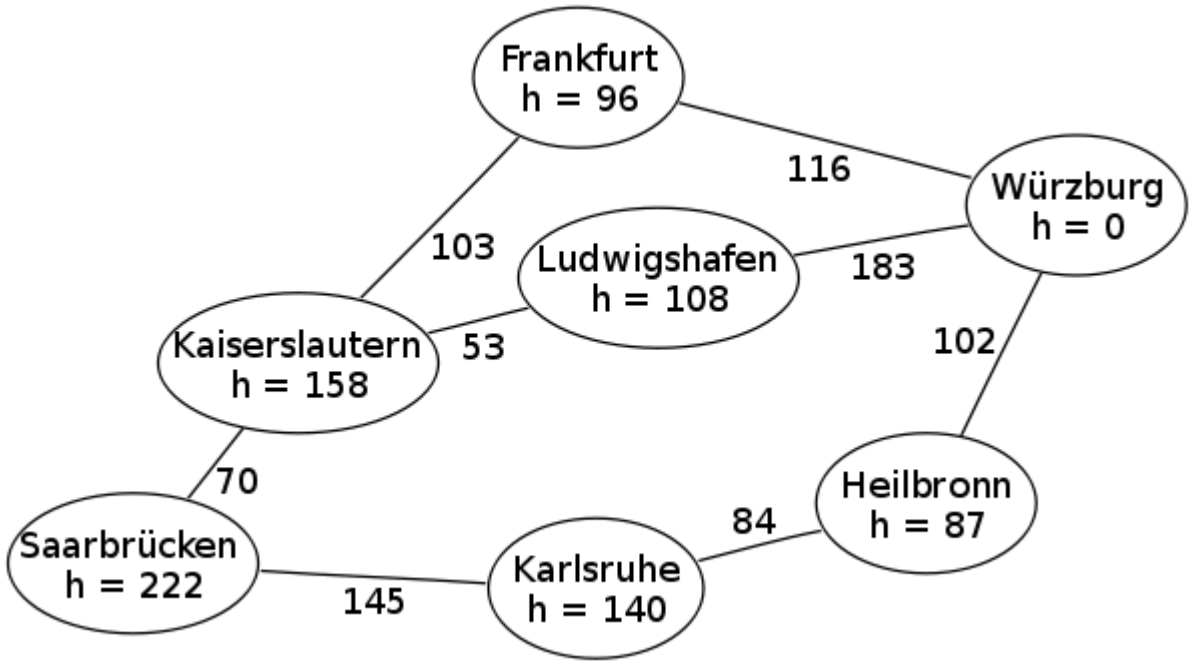
A* Suche

- Berechnung des kürzesten Pfades von Startknoten zu Zielknoten im Graphen
- Best-first Suche
- In jedem Schritt wird der Knoten mit der kleinsten Heuristik expandiert
- Heuristischer Wert eines Knoten: f(n) = g(n) + h(n)
  - g(n) = Kosten vom Startknoten zum Knoten n
  - h(n) = geschätzte Kosten des billigsten Pfades von n zum Zielknoten
- h(n) muss *zulässige Heuristik* sein damit optimale Lösungen gefunden werden
  - h(n) darf die tatsächlichen Kosten h*(n) des Pfades nie überschätzen z.B. euklidische Distanz

Lösungsansatz
A* Suche

TECHNISCHE UNIVERSITÄT DARMSTADT

Pfad: Saarbrücken → Würzburg
h(n) = Luftliniendistanz

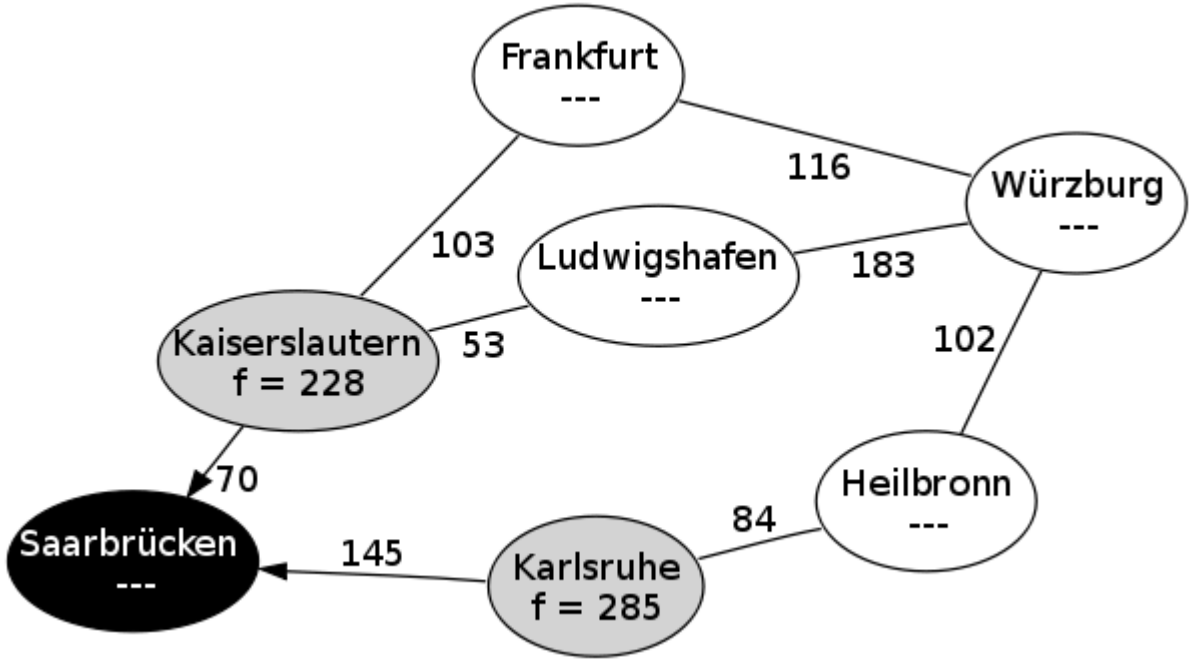


Quelle: Wikipedia

Lösungsansatz
A* Suche

TECHNISCHE UNIVERSITÄT DARMSTADT

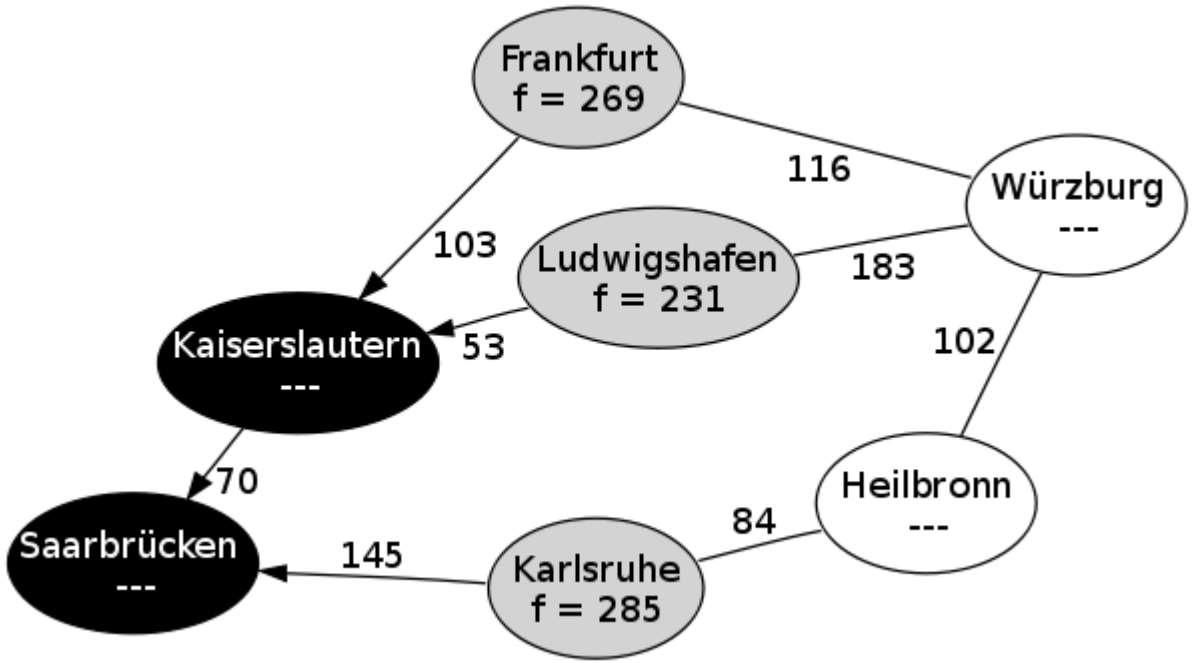
f(Kaiserslautern) = 70 + 158 = 228



Quelle: Wikipedia

Lösungsansatz
A* Suche

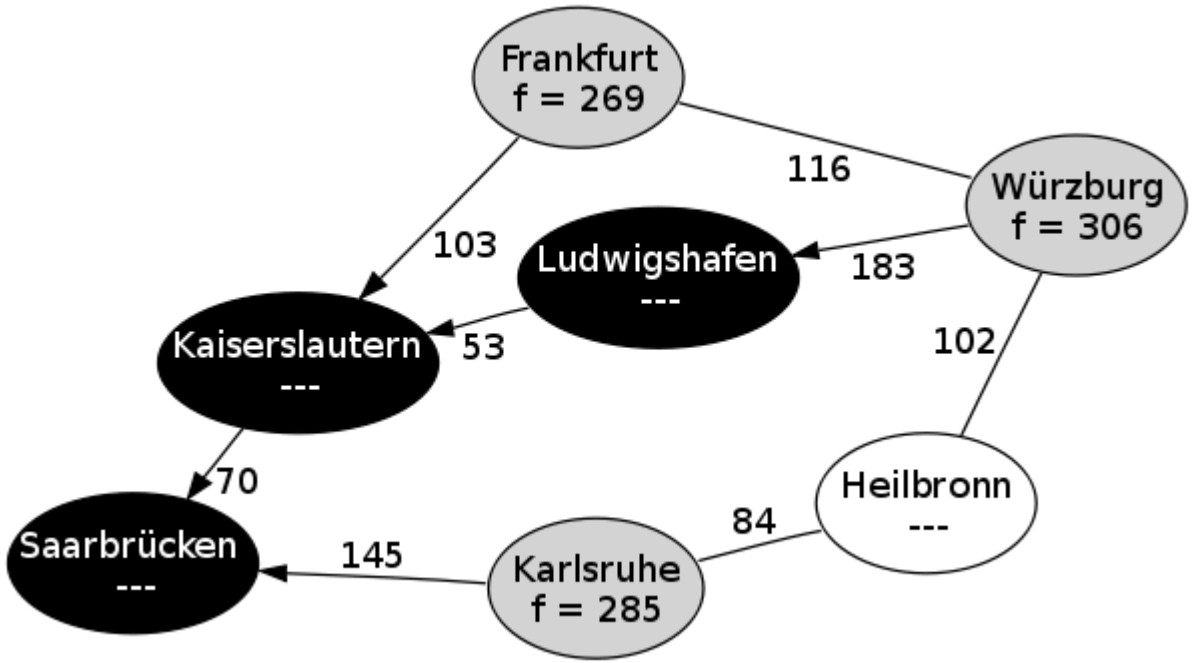
TECHNISCHE UNIVERSITÄT DARMSTADT



Quelle: Wikipedia

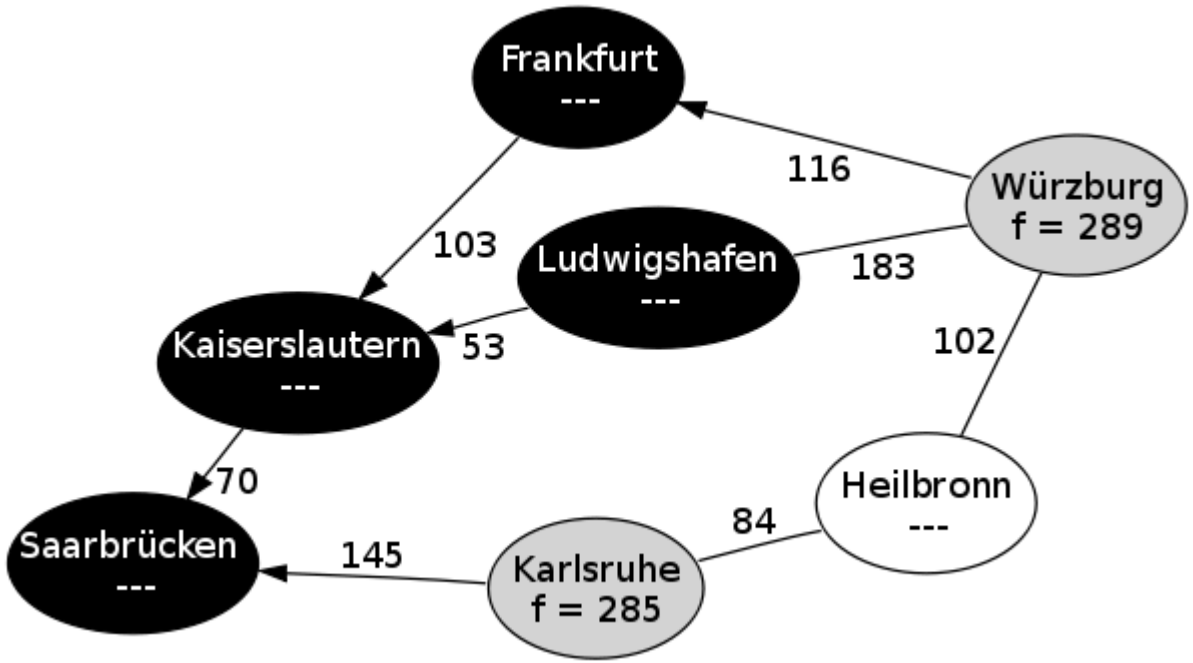
Lösungsansatz
A* Suche

TECHNISCHE UNIVERSITÄT DARMSTADT



Quelle: Wikipedia

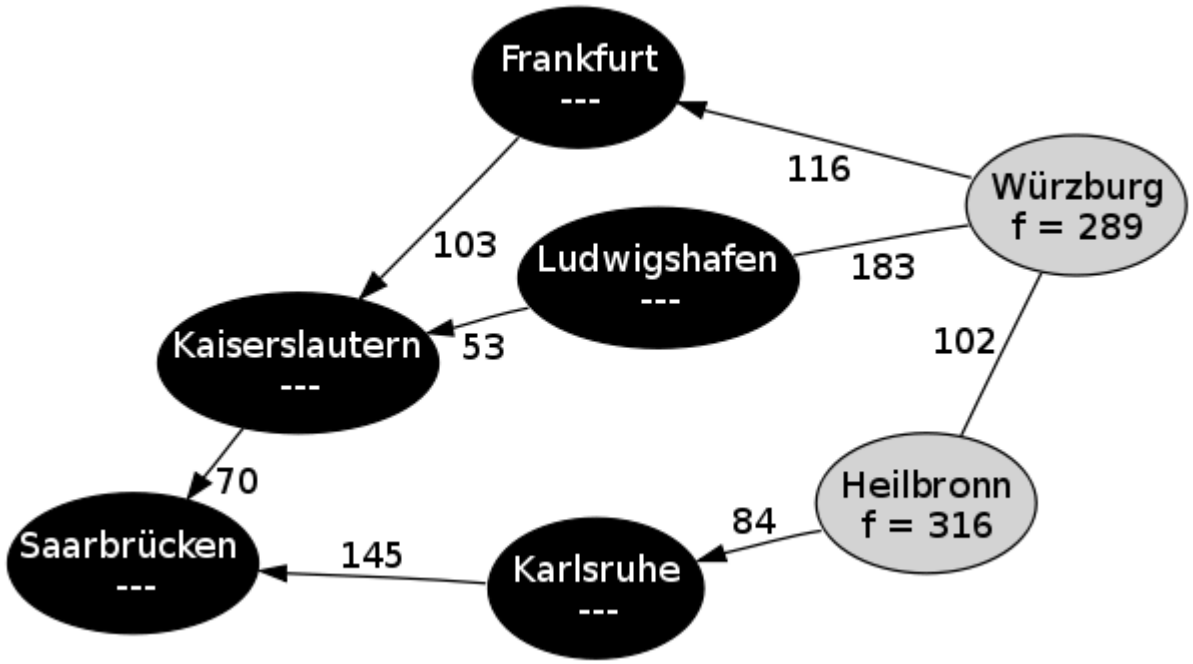
Lösungsansatz
A* Suche



Quelle: Wikipedia

Lösungsansatz
A* Suche

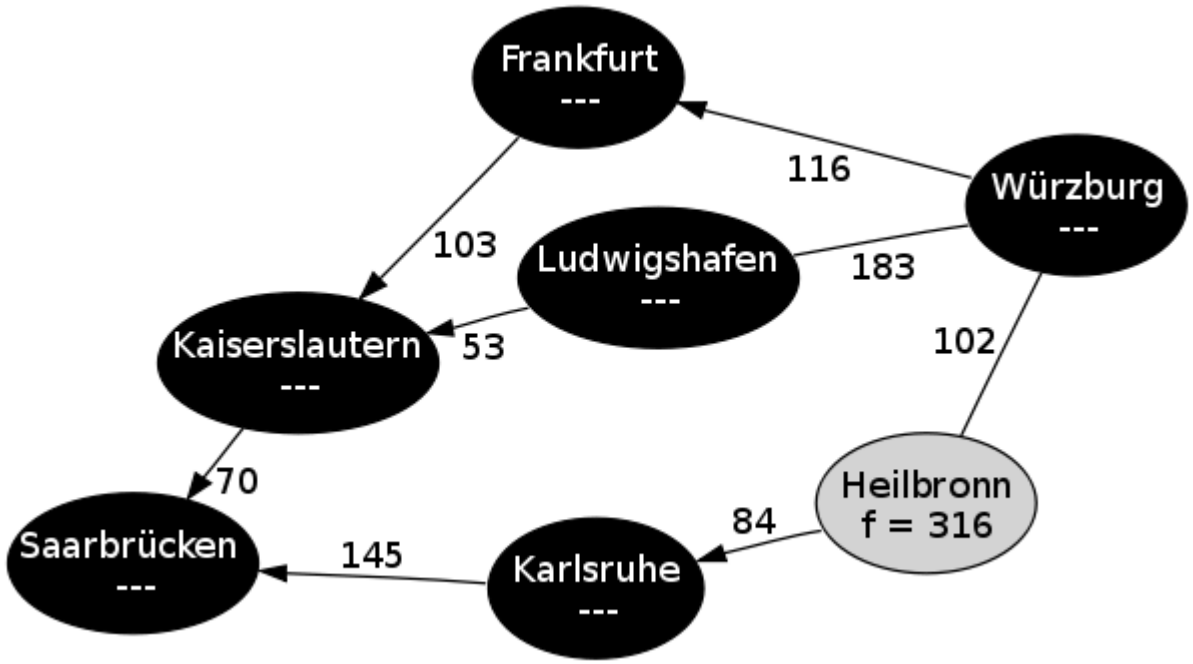
TECHNISCHE UNIVERSITÄT DARMSTADT



Quelle: Wikipedia

Lösungsansatz
A* Suche

TECHNISCHE UNIVERSITÄT DARMSTADT



Quelle: Wikipedia

Lösungsansatz
IDA* Suche

TECHNISCHE UNIVERSITÄT DARMSTADT

Problem:

- Hoher Speicherplatzbedarf von A*

IDA*

- Iterativ-deepening A*
- In jeder Iteration wird obere Schranke für Kosten (f(n)) festgelegt
- Übersteigt die Suche diese Schranke wird Iteration abgebrochen
- Neustart mit erhöhter Schranke
- Geringerer Speicherbedarf
- Wiederholtes Erzeugen von Knoten unproblematisch
- Findet optimale Lösungen (bei zulässiger Heuristik)

Lösungsansatz Verbesserungen (R0)

TECHNISCHE UNIVERSITÄT DARMSTADT

Naiver Ansatz: h(n) = Manhattan Distanz

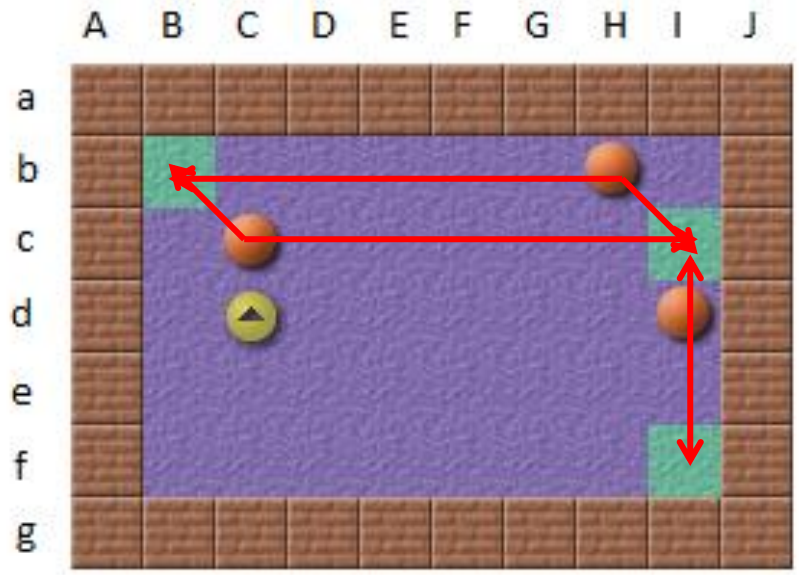
- Keine Lösung zu finden
- Zu großer Unterschied zwischen h(n) und Lösungslänge

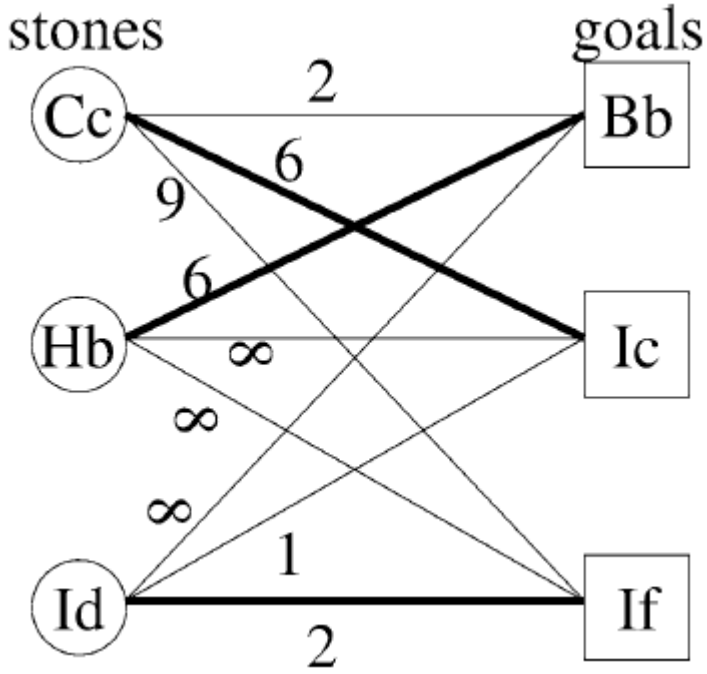
Minimum matching lower bound (R0)

- Bessere untere Schranke für Anzahl der Züge
- $O(N^3)$
- Distanz aller Kisten zu allen Zielfeldern berechnen
- Jede Kiste wird einem Zielfeld zugewiesen, sodass:
  - Keinem Zielfeld sind mehr als eine Kiste zugewiesen
  - Gesamtzuweisung ist minimal bzgl. Distanz

→ 0 Level gelöst

Lösungsansatz Verbesserungen (R0)



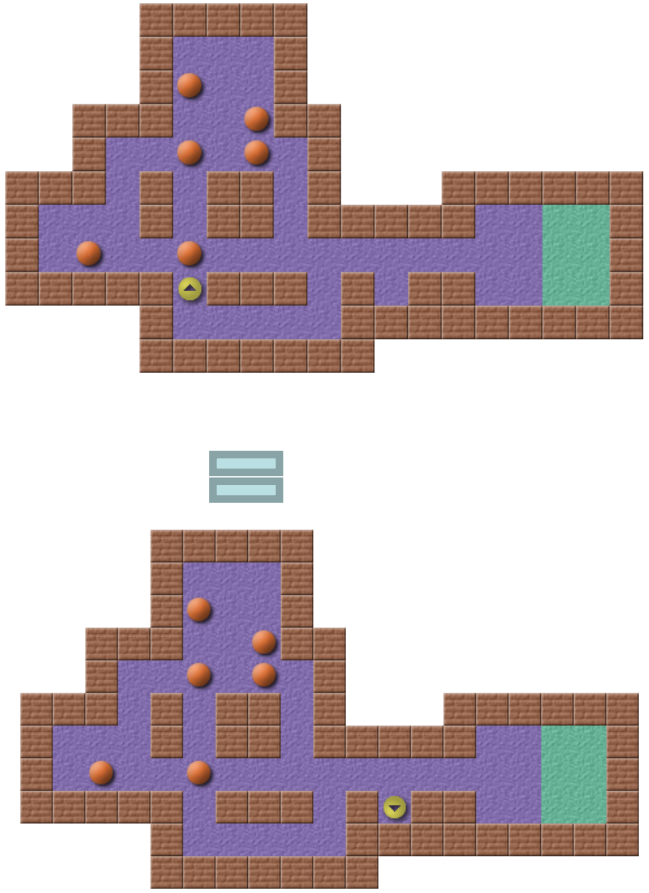


Lösungsansatz Verbesserungen (R1)

TECHNISCHE UNIVERSITÄT DARMSTADT

Transposition tables (R1)

- Verhindert, dass gleiche Zustände wiederholt erzeugt werden
- Speicherung von bis zu 2^{18} Knoten in Hashtabelle
- Zugriff über:
  - Position der Kisten (muss exakt stimmen)
  - Position der Spielfigur (muss erreichbar sein)



→ 5 Level gelöst (+5)

Lösungsansatz Verbesserungen (R2)

TECHNISCHE UNIVERSITÄT DARMSTADT

Move ordering (R2)

- Sortierung der Züge, sodass „gute" Züge zuerst versucht werden
- Sortierung nach:
  1. Züge, die wiederholt die gleichen Kisten bewegen
  2. Optimale Züge (Verringerung von h(n))
  3. Restliche Züge

- Nur in der letzten Iteration effektiv

→ 4 Level gelöst (-1)

Lösungsansatz Verbesserungen (R3)

TECHNISCHE UNIVERSITÄT DARMSTADT

Deadlock table (R3)

- Erkennen von Deadlocks
- Off-line Suche aller Kisten-Wand Positionen im 4x5 Bereich
- Testen auf Deadlock
- Speichern in Deadlock table

- Jeder Zug wird mit der Deadlock table getestet

→ 5 Level gelöst (+1)

Lösungsansatz Verbesserungen (R4)

TECHNISCHE UNIVERSITÄT DARMSTADT

Tunnel marcos (R4)

- Zusammenfassen zusammenhängender Züge zu Makros
- Zug wird durch ein Tunnel-Makro ersetzt, wenn
  - Kiste in einen *oneway* Tunnel bewegt wird
- Makro bewegt die Kiste durch den gesamten Tunnel

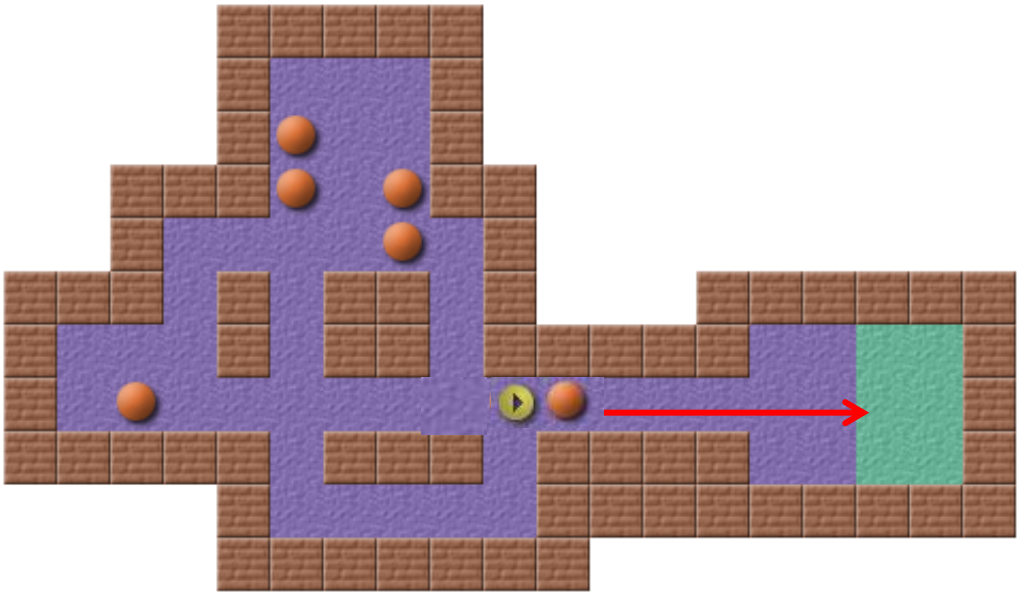
oneway Tunnel

- Ein Feld breit
- Einzige Verbindung von zwei Bereichen im Level

→ 6 Level gelöst (+1)

Lösungsansatz Verbesserungen (R4)

TECHNISCHE UNIVERSITÄT DARMSTADT



Lösungsansatz Verbesserungen (R5)

TECHNISCHE UNIVERSITÄT DARMSTADT

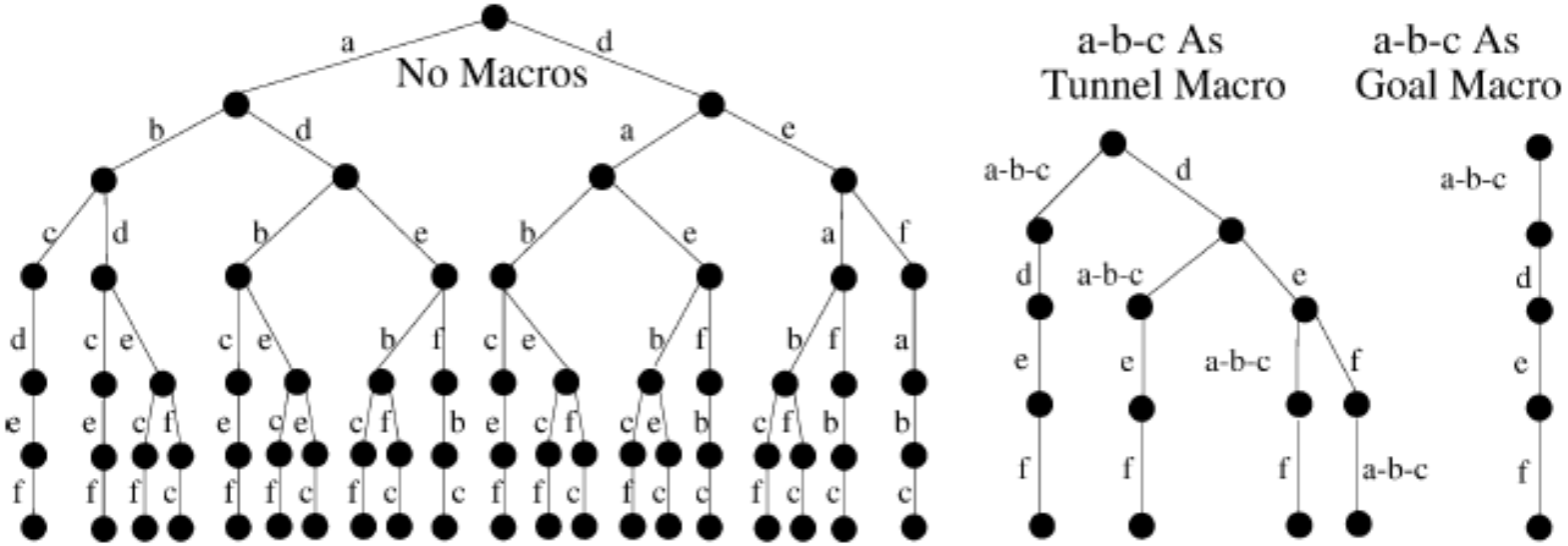
Goal macros (R5)

- Zielbereiche sind meistens durch kleinen Eingang erreichbar
- Problem zerlegen:
  - Bewegen einer Kiste zum Eingang des Zielbereichs
  - Bewegen einer Kiste vom Eingang zum Zielfeld
- Unabhängig lösbar
- Vorberechnung der Reihenfolge zum Platzieren der Kisten auf Zielfeldern (ohne Deadlock)

→ 17 Level gelöst (+11)

Lösungsansatz Verbesserungen (R5)

TECHNISCHE UNIVERSITÄT DARMSTADT



Lösungsansatz Verbesserungen (R6)

TECHNISCHE UNIVERSITÄT DARMSTADT

Goal cuts (R6)

- Wenn ein Goal macro erreicht werden kann
  - Bewege Kiste zum Goal macro
  - Alternative Züge werden nicht beachtet

- Deutliche Verbesserung
  - Suchbaum fast halbiert
  - Für gelöste Level um Faktor 6 verkleinert

→ 24 Level gelöst (+7)

Lösungsansatz Verbesserungen (R7)

TECHNISCHE UNIVERSITÄT DARMSTADT

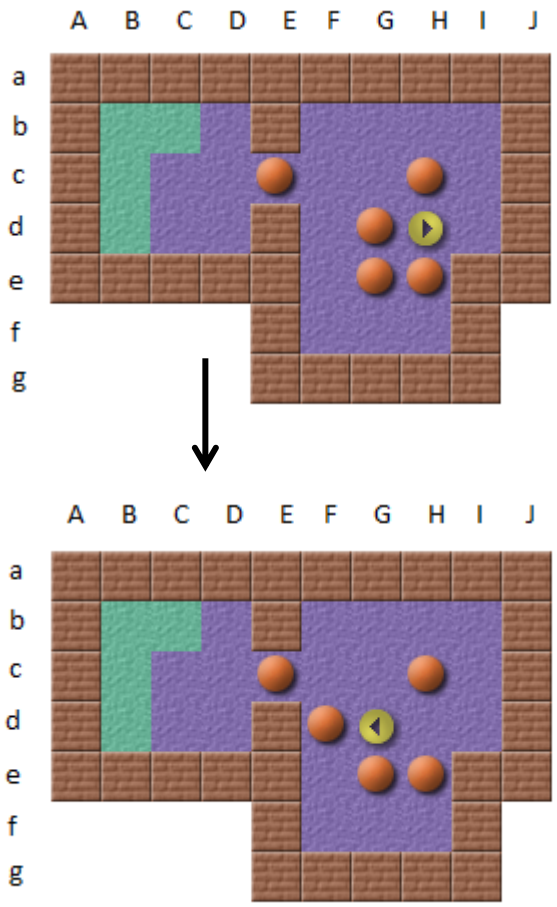
Deadlocks

- Kann von einer bis zu allen Kisten erzeugt werden
- Schwierig zu finden
- Erzeugt ein Muster von Kisten einen Deadlock, haben alle Zustände die dieses Muster enthalten einen Deadlock

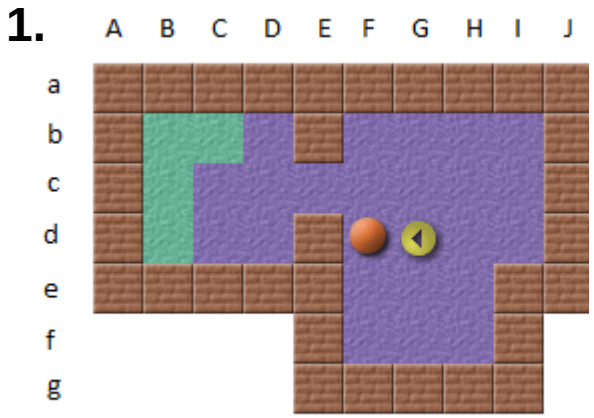
Pattern search (R7)

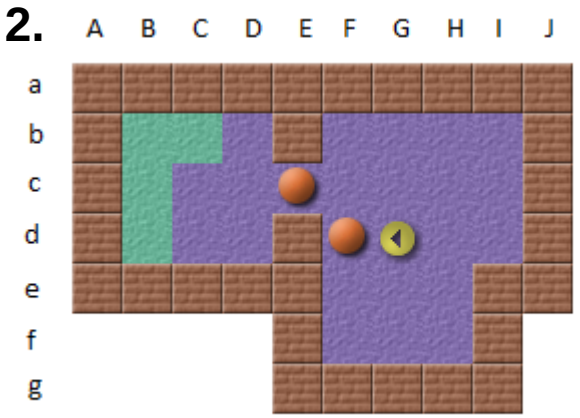
- Findet Muster die zeigen, dass untere Schranke des Zustands falsch ist
- Speicherung der Muster
- Korrektur der unteren Schranke (1 bis ∞)
- Suche findet auf Testfeld statt

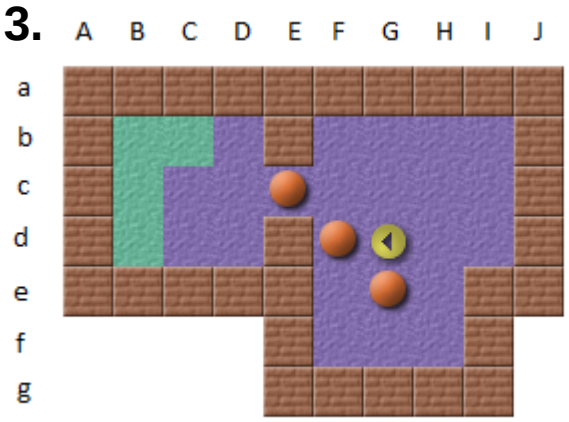
Spielfeld

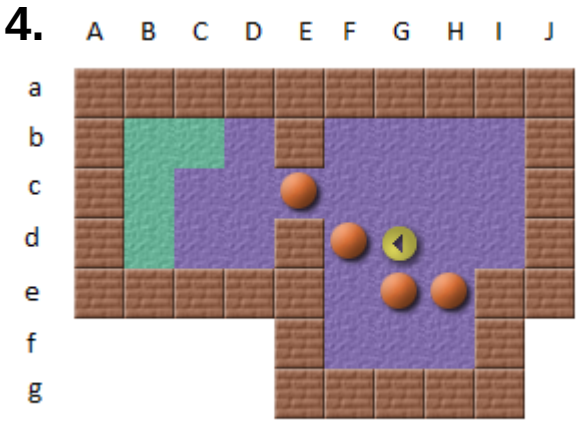


Testfeld









Lösungsansatz Verbesserungen (R7)

TECHNISCHE UNIVERSITÄT DARMSTADT

PIDA*

- Optimiert für Suche von Mustern
- Sucht nach Lösungen im Testfeld

- 2 Suchen notwendig
  - ➢IDA*
  - ➢PIDA*
- Deutliche Verbesserung
- Aber Großteil der Suche nicht mehr für direkte Lösung verwendet

#	R7 = R6 + Pattern search		R6 = R5 + Goal cuts
	IDA* nodes	Total nodes	IDA* nodes
1	50	1,042	53
2	82	7,532	316
3	94	13,445	2,493
4	187	50,369	597
5	436	59,249	1,275,146
6	85	5,119	283
7	1,704	28,561	48,209
8	317	339,255	> 20,000,000
9	704	168,412	659,972
10	1,909	1,480,115	> 20,000,000
11	14,048	4,691,929	> 20,000,000
12	162,129	4,373,802	> 20,000,000
17	2,473	30,111	11,910
19	59,433	> 20,000,000	> 20,000,000
21	1,853	154,593	10,643,971

→ 48 Level gelöst (+24)

Lösungsansatz Verbesserungen (R8)

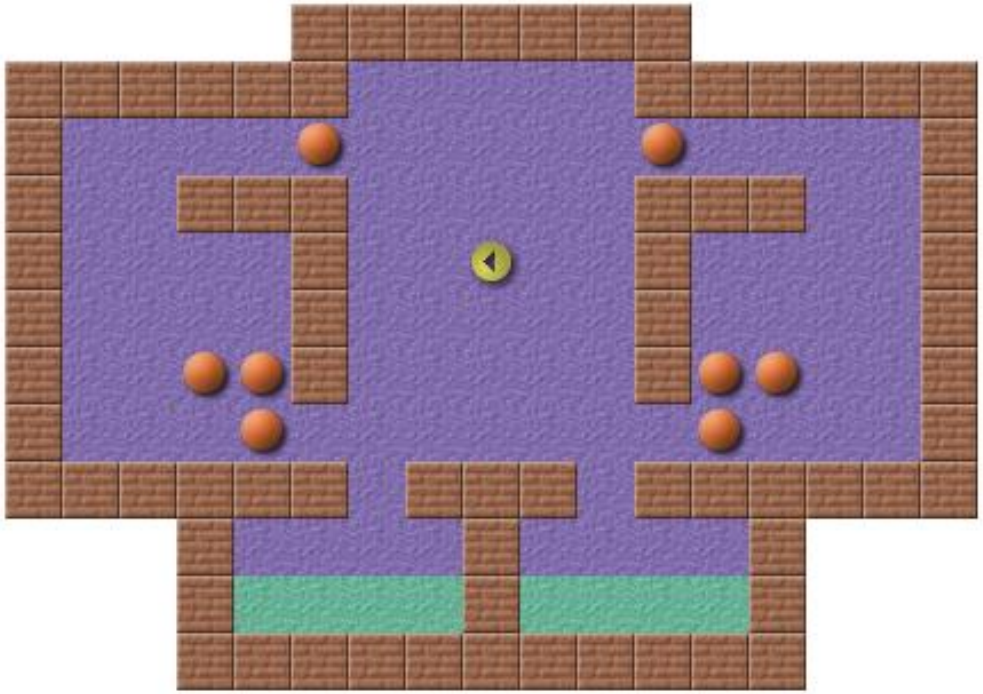
TECHNISCHE UNIVERSITÄT DARMSTADT

Relevance cuts (R8)

- Manche Züge sind unwichtig für vorhergehende Zugfolge
- Relevance cuts löschen Züge aus der Suche die unwichtig sind
- Berechnung des gegenseitigen Einflusses von je 2 Spielfeldern
  - ➢Gewichtung nach 4 Parametern
    - Mehr Alternativen - weniger Einfluss
    - Felder auf dem Zielpfad haben hohen Einfluss
    - Verbindung (Manpath(-), Stonepath(+))
    - Tunnel
  - ➢Finden des Pfades mit dem größten Einfluss (shortest-path Algorithmus)
- Speichern des gegenseitigen Einflusses von allen Spielfeldpaaren
- Ein nicht-lokaler Zug alle 9 Züge erlaubt

Lösungsansatz Verbesserungen (R8)

TECHNISCHE UNIVERSITÄT DARMSTADT



→ 50 Level gelöst (+2)

Lösungsansatz Verbesserungen (R9)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Heuristik h für A* Algorithmen muss *zulässig* sein, damit optimale Lösungen gefunden werden
- Ziel: bessere Annäherung von h an h* (tatsächliche Distanz)

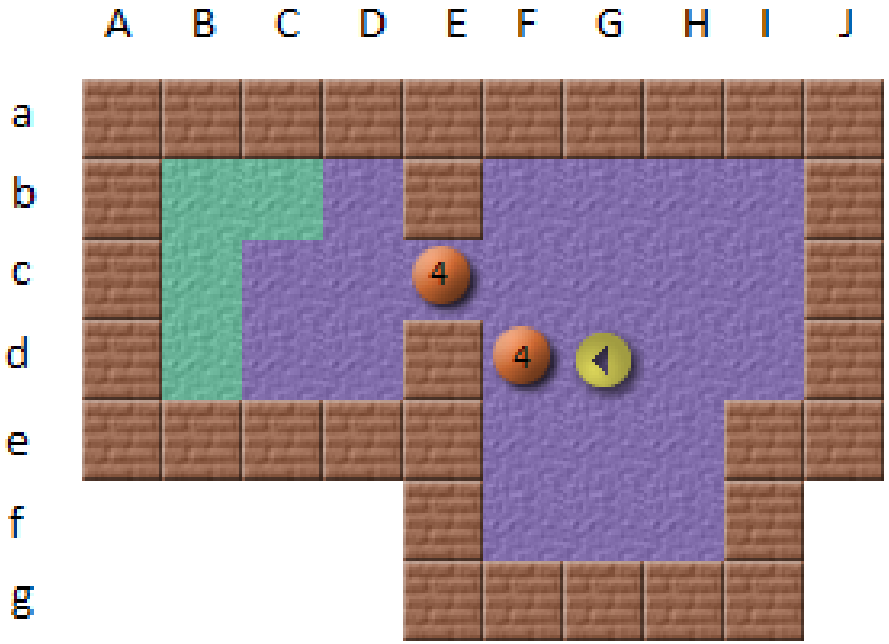
Overestimation (R9)

- Benutzt pattern-search Muster für bessere Annäherung von h an h*
  - ➢Durchsuchen aller gespeicherten Muster für aktuellen Zustand
  - ➢Untere Schranke eines Musters wird auf alle Kisten aufgeteilt
  - ➢Jeder Kiste wird Maximum dieser aufgeteilten Heuristik zugewiesen
  - ➢Heuristik des aktuellen Zustands ist Summe aller Werte der Kisten

→ 54 Level gelöst (+4)

Lösungsansatz Verbesserungen (R9)

TECHNISCHE UNIVERSITÄT DARMSTADT



h=8

Lösungsansatz Verbesserungen (R10)

TECHNISCHE UNIVERSITÄT DARMSTADT

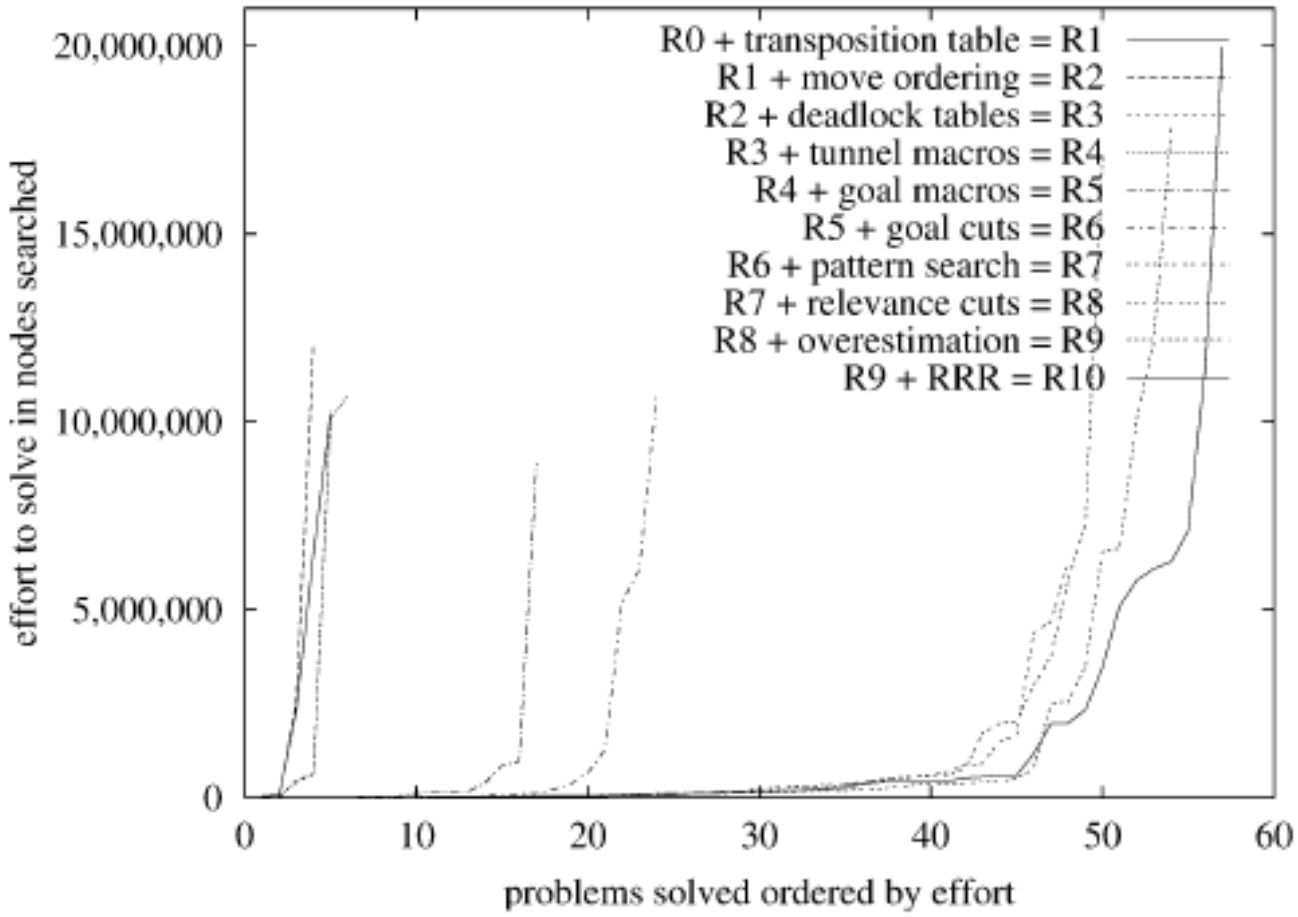
Rapid random restart (R10)

- Gesamten Rechenaufwand nicht nur auf einen Versuch konzentrieren
- Neustart mit neuen Parametern
- Randomisieren der Zug-Sortierung

→ 57 Level gelöst (+3)

Zusammenfassung

TECHNISCHE UNIVERSITÄT DARMSTADT



Zusammenfassung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Anwendungsunabhängige Algorithmen sind (noch) nicht in der Lage Lösungen für Sokoban zu finden
- Stufenweise Anpassung des Programms (R0-R10) an die Domäne notwendig
  - ➢Reduktion des Suchraums durch:
    - Techniken aus anderen Suchproblemen (Rubic‘s Cube, 15-Puzzle)
    - Spielerfahrung
    - Analyse des Suchbaums
- Diese Anpassungen lassen sich auch auf andere Domänen übertragen

Quellenangabe

TECHNISCHE UNIVERSITÄT DARMSTADT

- *Sokoban: Enhancing general single-agent search methods using domain knowledge*
  Andreas Junghanns, Jonathan Schaeffer
- *Pushing Blocks is NP-Complete for Noncrossing Solution Paths*
  Erik D. Demaine, Michael Hoffmann
- *Sokoban: Evaluating Standard Single-Agent Search Techniques in the Presence of Deadlock*
  Andreas Junghanns, Jonathan Schaeffer
- *Artificial Intelligenz: A Modern Approach*
  S. Russel, P.Norvig
- *Enhanced Iterative-Deepening Search*
  Alexander Reinefeld, T.A. Marsland
- *http://de.wikipedia.org/wiki/A*-Algorithmus*