

Heuristische/Informierte Suche

- **Ziel:** Verringerung der Suchdauer
- **Idee:** Einbauen von zusätzlichem Wissen
- 2 Gruppen von Algorithmen bewerten
 - Lösungsschritte
 - Lösungen

Übersicht

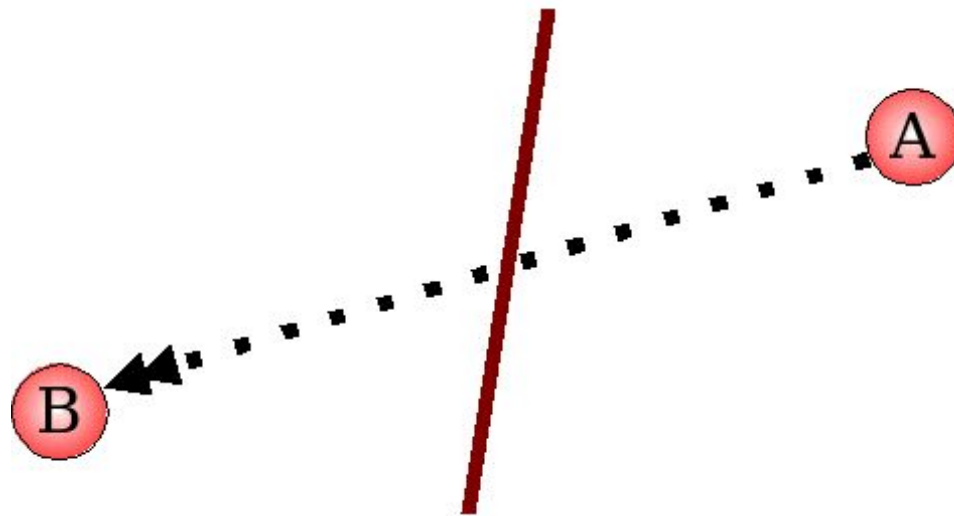
- Greedy-Search
 - (direktes Anlaufen der Lösung)
- Relaxed Problem
 - (Finden einer Heuristik)
- A* Search
 - (Greedy Search + uniform cost search)

Übersicht

- Iterative Algorithmen
 - (Verbesserung von fertigen Lösungen)
- Hill-climbing
 - (bessere Lösungen verwenden, schlechtere verwerfen)
- Simulated Annealing
 - (u.U. Schlechtere verwenden/bessere verwerfen)
- Ausblick Genetic Algorithms
 - (Gedächtnis, Breitensuche, unabhängige Datendarstellung
⇒ Anwendung in sehr komplexen Problemen)

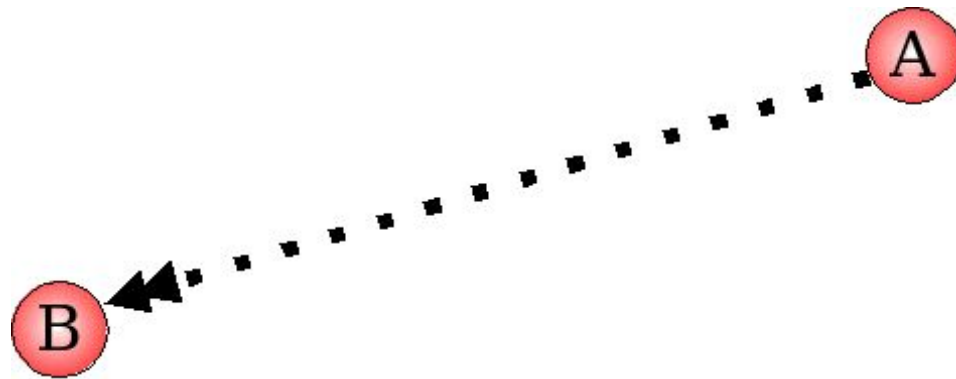
Einstieg (1)

“Schwieriges” Pfadsuche-Problem mit Wand



Einstieg (2)

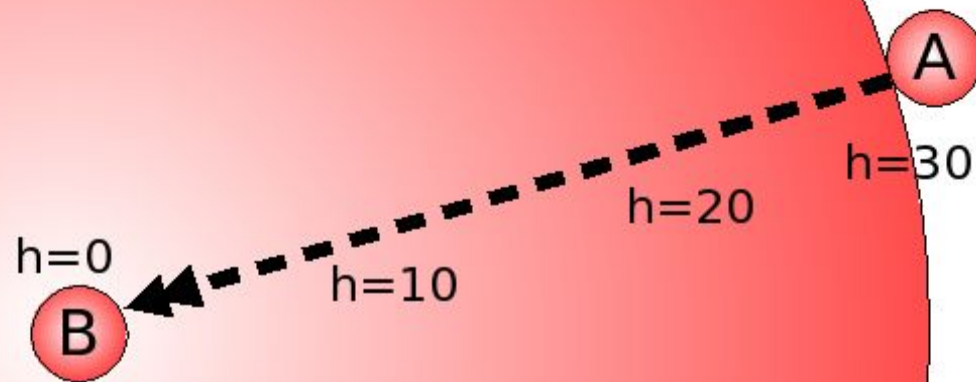
Vereinfachtes Pfadsuche-Problem ohne Wand



Einstieg (3)

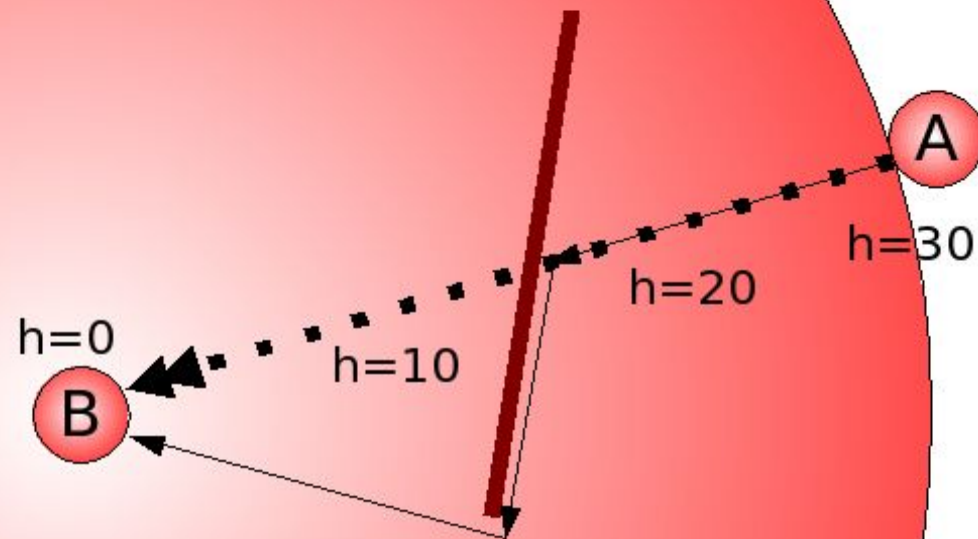
Darstellung mit heuristischer Funktion h

Hier: $h \triangleq$ Abstand zu B



Einstieg (4)

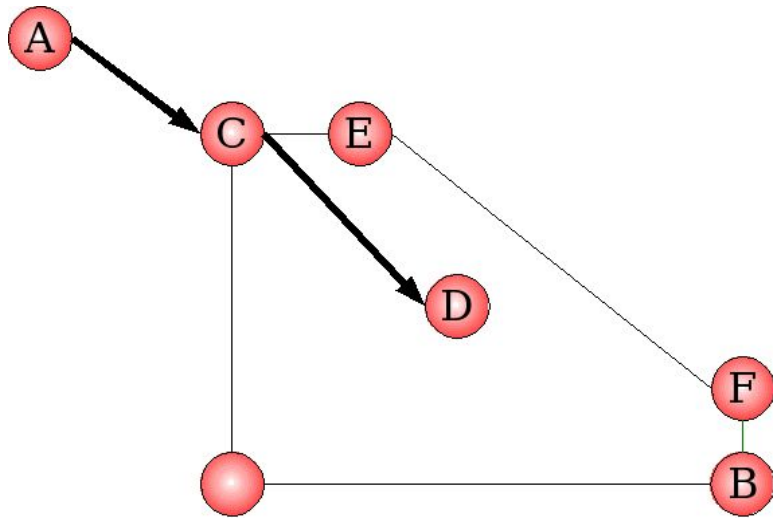
Greedy Search



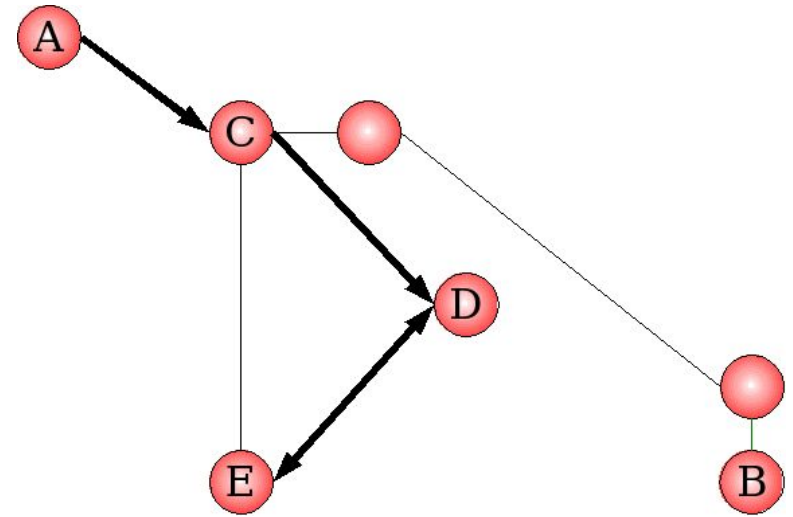
h : heuristische Funktion

Greedy search (1)

(Sackgasse und Schleife)



Sackgasse

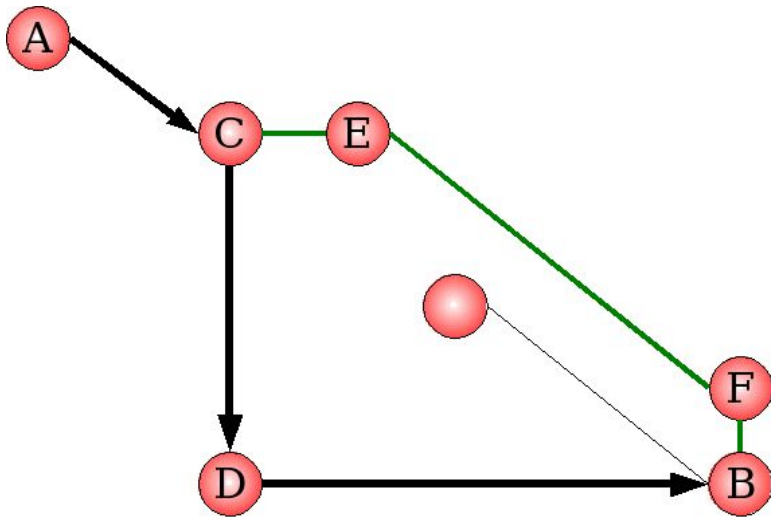


Schleife

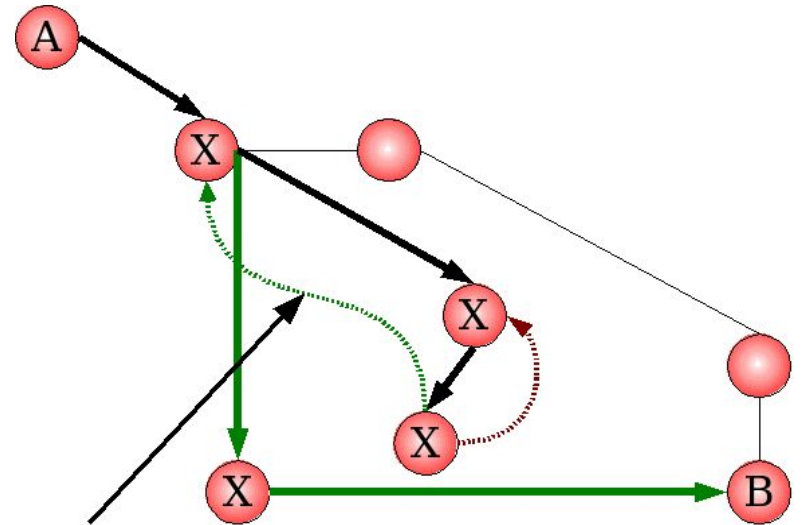
⇒ **GREEDY SEARCH nicht vollständig**

Greedy search (2)

(ungünstiger Start und Gedächtnis)



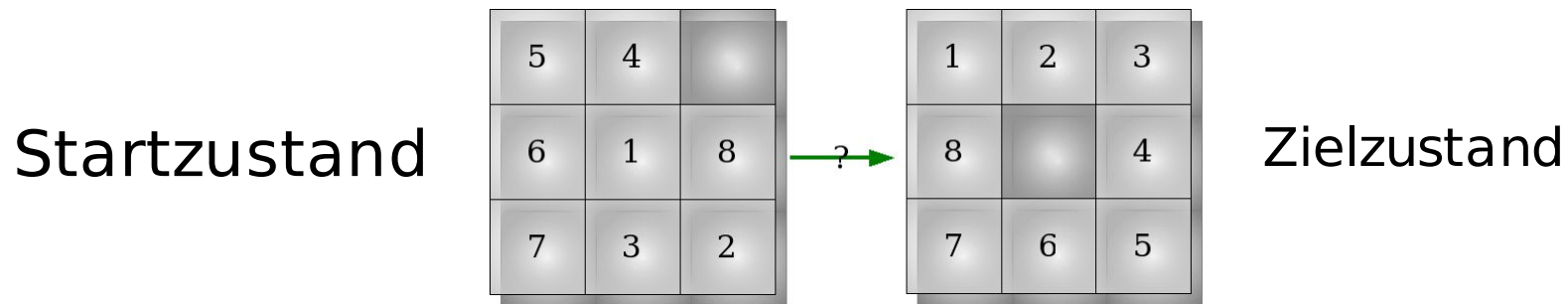
⇒ **GREEDY SEARCH nicht optimal**



⇒ **vollständig aber nicht optimal**

Heuristikenentwicklung

8 Puzzle Problem



Ein Teil darf

- a) pro Schritt nur 1 Feld
- b) nicht schräg
- c) nur in ein freies Feld verschoben werden

Heuristik 1:
korrekte Positionen
Ignorierung der
Regeln a), b) und c)

| | | |
|---|---|---|
| 5 | 4 | |
| 6 | 1 | 8 |
| 7 | 3 | 2 |

Heuristik 2:
Abstand zum Ziel
Ignorierung der
Regel c)

| | | |
|-----|-----|-----|
| (4) | (2) | |
| (2) | (2) | (2) |
| (0) | (2) | (4) |

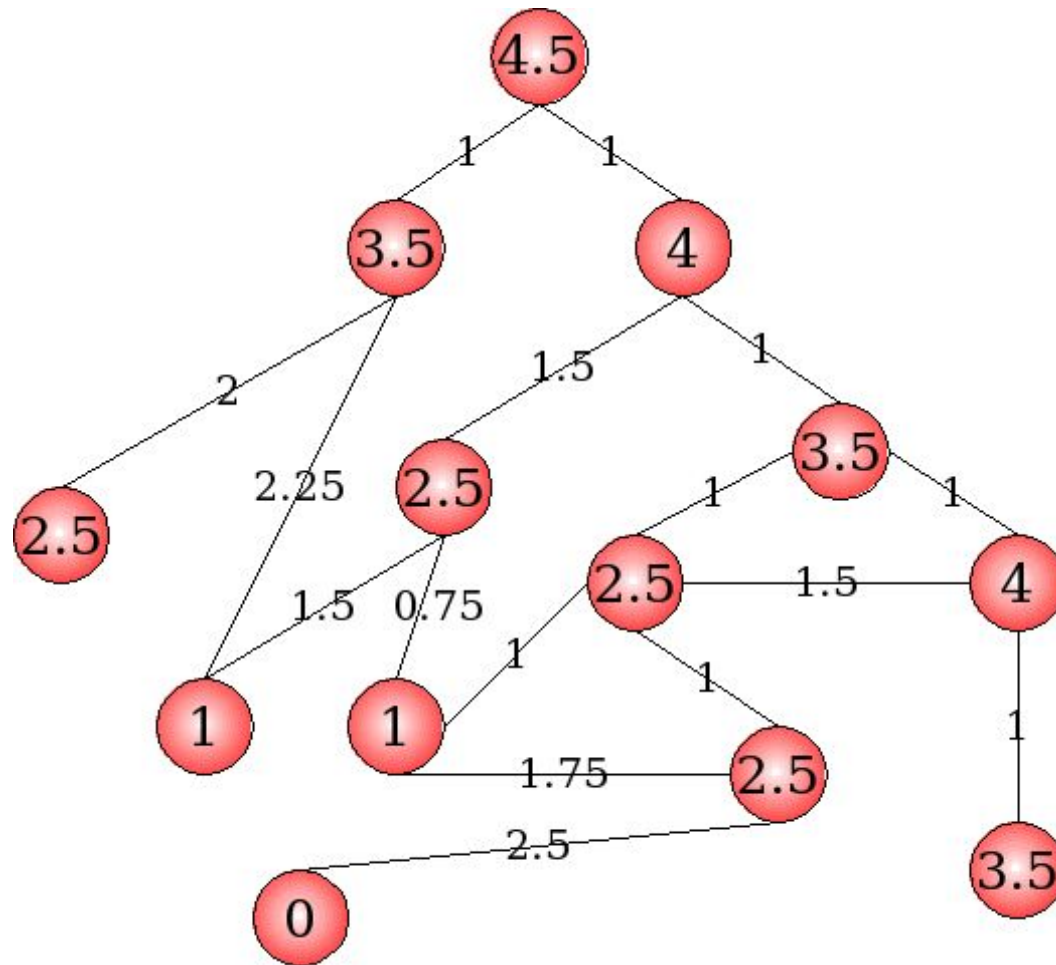
A* Search (1)

Übersicht

- Kombination aus Greedy und uniform-cost search
- Vollständig und optimal
- Meist Luftlinie (Greedy Search) + bisherige Kantenlänge (uniform cost search)
- Begrenzender Faktor ist Speicher
- Im schlechtesten Fall Speicher- und Suchzeit $O(b^m)$,
- Im Durchschnitt schneller als allgemeine Suche
- Variante 1: Iterative Deepening A* Search
 - (wie Iterative deepening Algorithmus)
- Variante 2: Simplified Memory-Bounded A* Search
 - (mit Knoten vergessen)

A* Search (2)

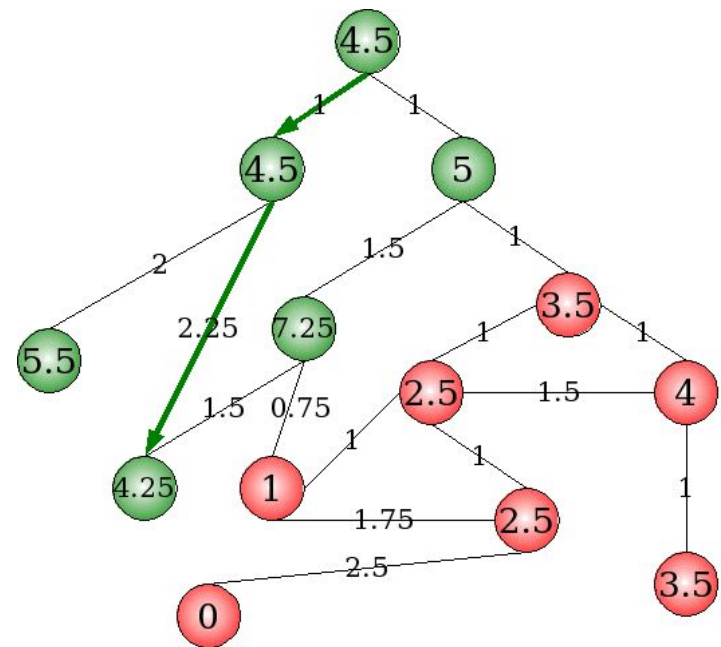
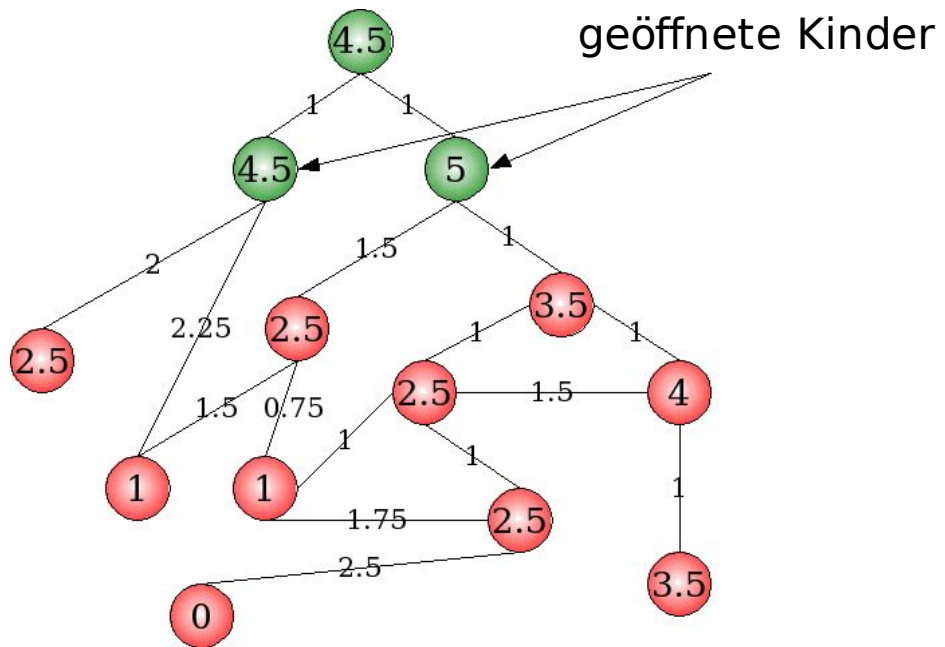
Beispiel Pfadsuche



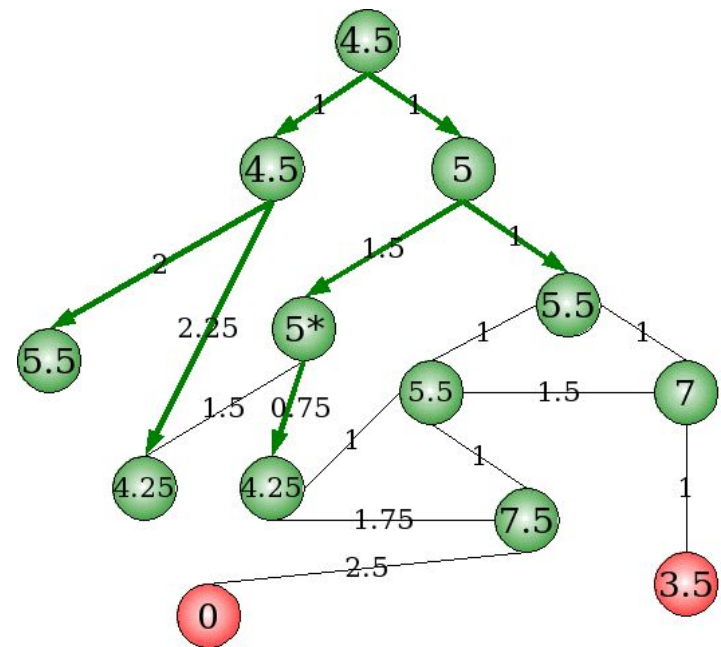
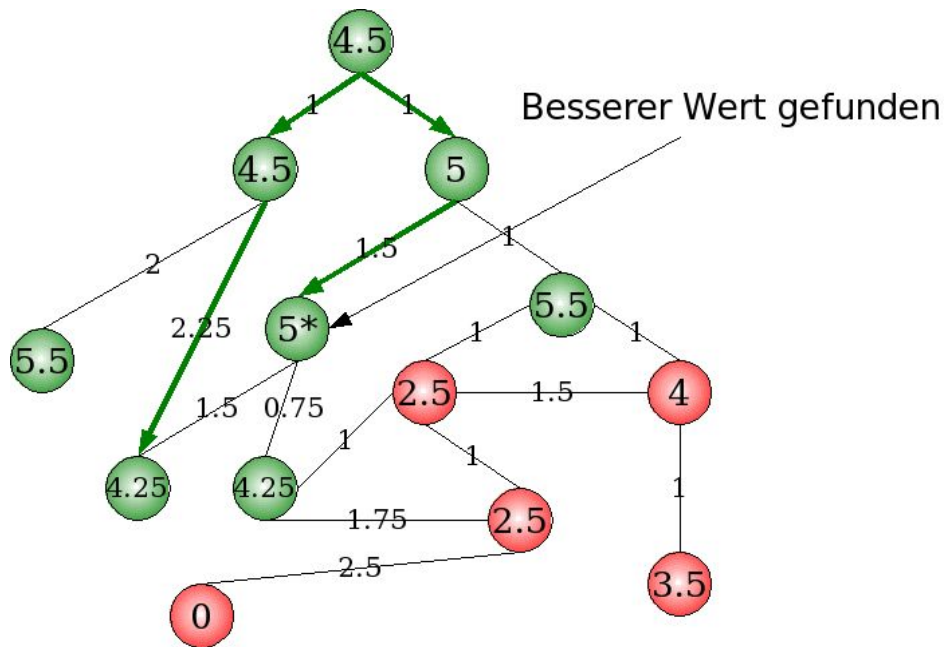
Zahlen an Kanten: Kantenlänge

Zahlen in Kreisen: Luftlinie zum Ziel

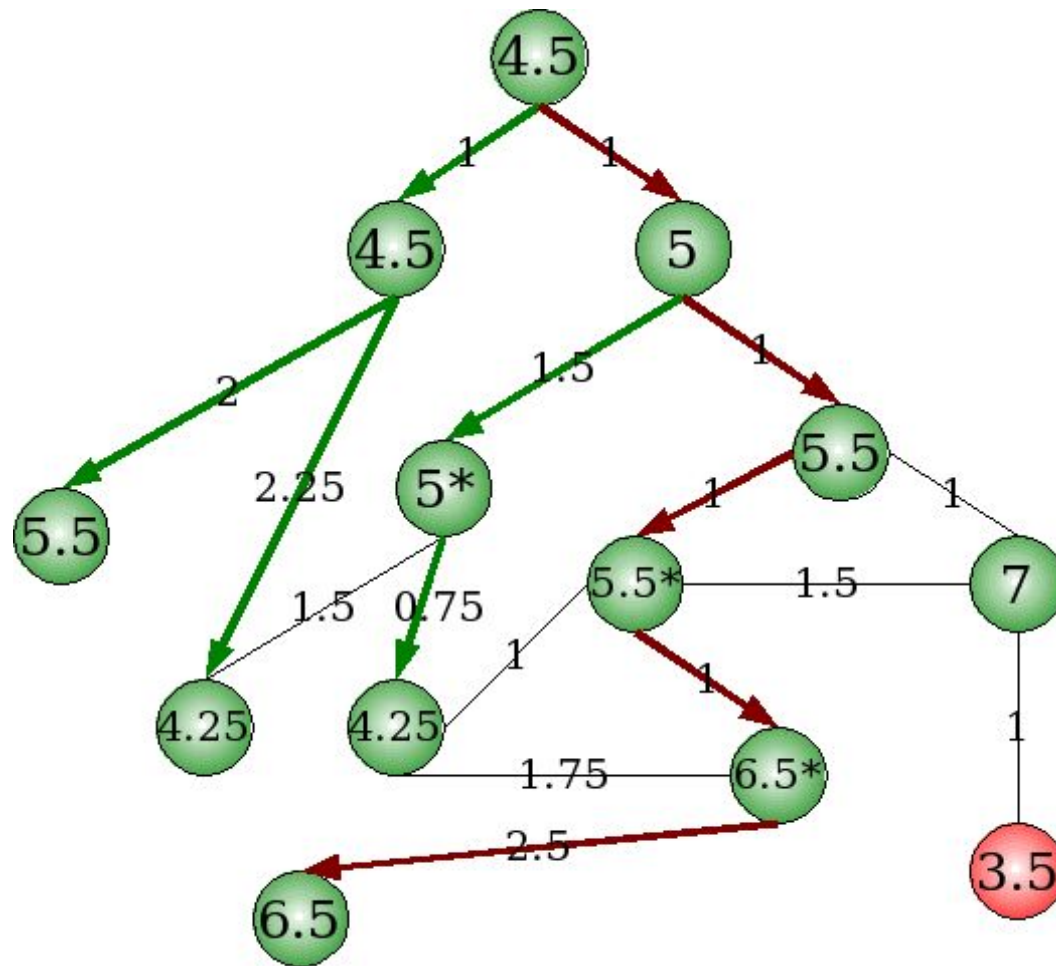
A* Search (3)



A* Search (4)



A* Search (5)



⇒ Optimaler Weg gefunden

Iterative Algorithmen

- Basis: **zufällige Suche**
- komplette Lösung **schrittweise angepasst**
- **Grundsätzlicher Ablauf:**
 - 1. Zufällige Lösung wird erstellt
 - 2. Lösung wird zufällig verändert und mit der alten verglichen
 - 3. besser \Rightarrow Neue Lösung fuer 2 verwenden
 - 4. schlechter \Rightarrow alte Lösung fuer 2 verwenden
- Optimum oft unbekannt:
 - \Rightarrow Suche endet sobald eine gültige Lösung “ausreichend” gut ist

Hillclimbing Algorithmus (1)



Fitness $\hat{=}$ Bewertungsfunktion

Fitnesslandschaft $\hat{=}$ Graph der Fitnessfunktion

Ziel: beste Fitness, z.B. Berggipfel A erreichen

Hier: Von E und F wird direkt auf Gipfel A gelaufen

Hillclimbing Algorithmus (2)

Beispiel

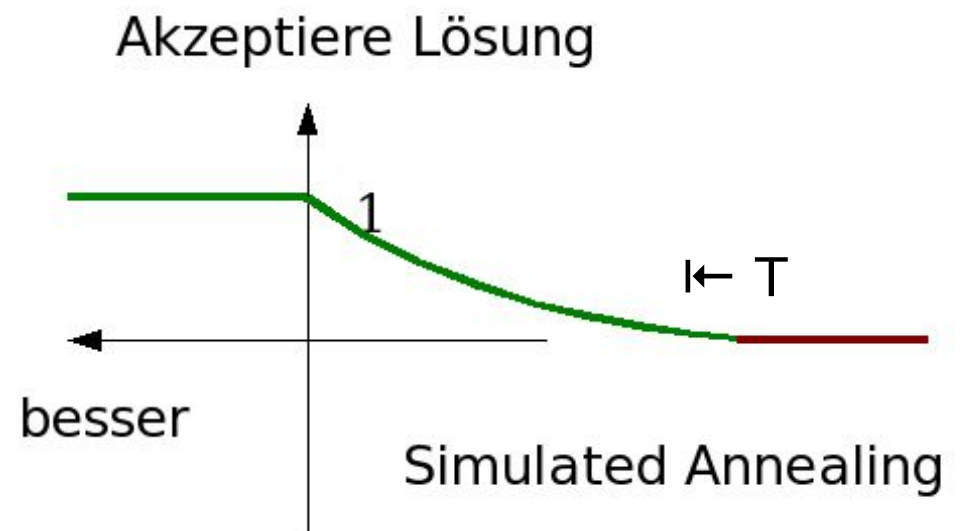
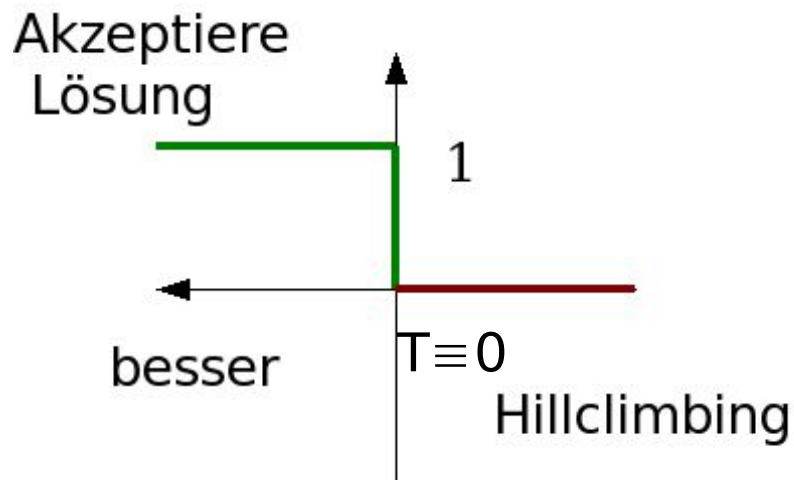


- B: ebene Fitnesslandschaft \Rightarrow zufällige Suche bis E gefunden
- C läuft nicht zu F sondern auf lokales Optimum nach rechts
- D läuft zwar Richtung A, bleibt jedoch ebenfalls hängen
- Ist A wirklich das Optimum? Andere Berggipfel rechts und links des Bilds?

Simulated Annealing Algorithmus (1)

Unterschiede zu Hillclimbing

- Akzeptiere mit Wahrscheinlichkeit T auch schlechtere Lösungen um lokale Optima zu überwinden
- Verringere die T stufenweise oder rellwertig langsam über die Zeit
- Option: Verwerfe u.U. Minimal bessere Lösungen



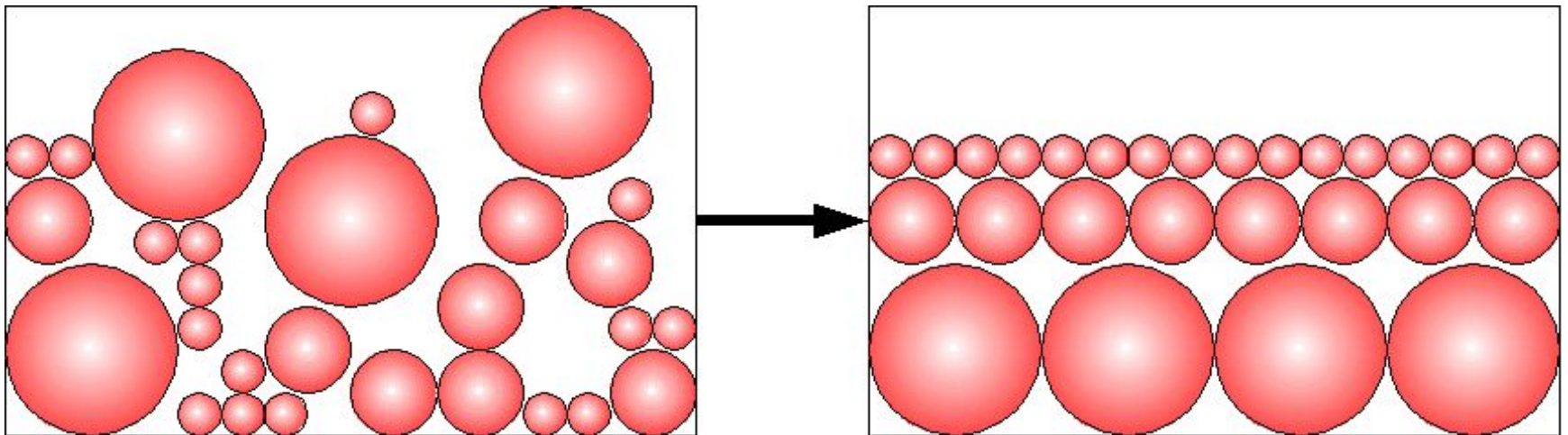
Simulated Annealing (2)

Beispiel

Ziel: Sand oben, Kies in der Mitte und Steine unten

Idee: Zustand zufällig verändern \Rightarrow Schütteln

\Rightarrow Steine wandern nach unten und Sand nach oben



Simulated Annealing (3)

Beispiel

3 Temperaturstufen:

- $T = 3$: Steine bewegen sich

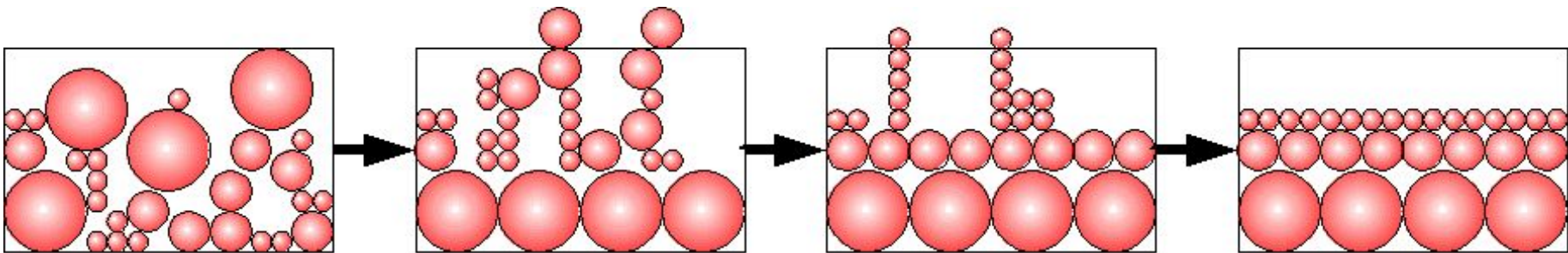
Rest wird durchgeschüttelt, mitunter in schlechtere Positionen

- $T = 2$: Kies bewegt sich

Sand wird durchgeschüttelt, mitunter in schlechtere Positionen

- $T = 1$: Sand bewegt sich bis optimaler Zustand bei

- $T = 0$ erreicht ist



Ausblick: Genetische Algorithmen

- Breitensuche mit mehreren “Agenten”
 - Gedächtnis mit Hilfe von Allelen und Intronen
 - Abkopplung von mutierendem Genmaterial und tatsächlicher Ausprägung
 - Rekombination durch Genaustausch
- ⇒ Abflachung steiler Bereiche in der Fitnesslandschaft
- Aber kein “heiliger Gral”: Hauptarbeit liegt in Formulierung von Problem und Bewertungsfunktion
- ⇒ Abschätzungen, Kreativität, gute Kenntnisse in Abstraktion von Problemen und Grenzen der genetischen Algorithmen sind wichtig

Zusammenfassung (1)

- **Allgemeine Suchalgorithmen:**
 - blinde Suche
 - direkt für unterschiedliche Probleme verwendbar
- **Breitensuche**
- Gesamte Kantenlänge \Rightarrow **Uniform Cost search**
- Pfade bis Sackgasse verfolgen \Rightarrow (Begrenzte) **Tiefensuche**
- Schrittweise Erhöhung der Suchtiefe \Rightarrow **Iterative Tiefensuche**
- Parallele Suche von Start und Ziel \Rightarrow **Bidirektionale Suche**
- CSP (**Backtracking, Forward checking**)

Zusammenfassung (2)

- **Heuristik**
 - Reduktion der Suchzeit durch zusätzliche Information
 - Finden über vereinfachtes Problem
- uniform cost search + greedy search \Rightarrow **A* Search** Algorithmus
- Optimieren fertiger Lösung \Rightarrow **Hillclimbing** Algorithmus
- Verfeinerung um lokalen Optima zu entkommen \Rightarrow **Simulated Annealing**
- Ausblick auf **genetische Algorithmen**