Berechnung von Potenzialheuristiken basierend auf Pattern-Datenbanken

Andreas Ferenczi

Universität Basel

andreas.ferenczi@stud.unibas.ch

30. Januar 2017

Überblick

- Einführung
  - Planungsaufgaben
  - Pattern-Datenbanken
  - Potenzialheuristiken
- Greedy Algorithmus
  - Algorithmus
  - Beispiel
- Algebraischer Algorithmus
  - Grundlagen
  - Algorithmus
  - Beispiel
- Experimente
  - Anzahl Iterationen des Greedy Algorithmus
  - Pattern Generatoren
  - Vergleich mit der FF-Heuristik
- Schlussfolgerungen
  - Qualität der Heuristik
  - Weiterführende Arbeit

Einführung

Gebiet

- Künstliche Intelligenz
- Handlungsplanung
- Heuristikbasierte Zustandsraumsuche

Einführung

Gebiet

- Künstliche Intelligenz
- Handlungsplanung
- Heuristikbasierte Zustandsraumsuche

Konkrete Aufgabe

- Potenzialheuristik
- Pattern-Datenbanken
- Implementation in Fast-Downward

Planungsaufgaben

Zustandsvektor

Ein **Zustandsvektor** ist ein Vektor, der aus allen relevanten Variablen eines Systems besteht.

Zustandsraum

Ein **Zustandsraum** kann als der Vektorraum der Zustandsvektoren verstanden werden.

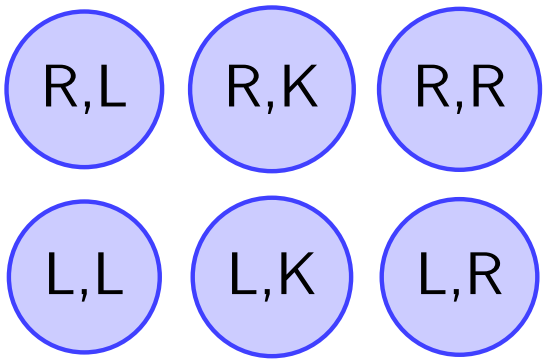
Planungsaufgabe

Eine **Planungsaufgabe** definiert sich über einen Zustandsraum, Aktionen, einen Initialzustand und Zielzuständen. Die Aufgabe besteht darin eine Folge von Aktionen zu finden, die den Initialzustand in einen Zielzustand überführt.

Beispiel

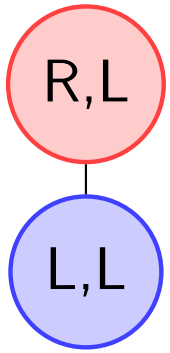
- Karren: links (L) oder rechts (R)
- Kiste: links (L), rechts (R) oder auf dem Karren (K)

Zustandsraum:

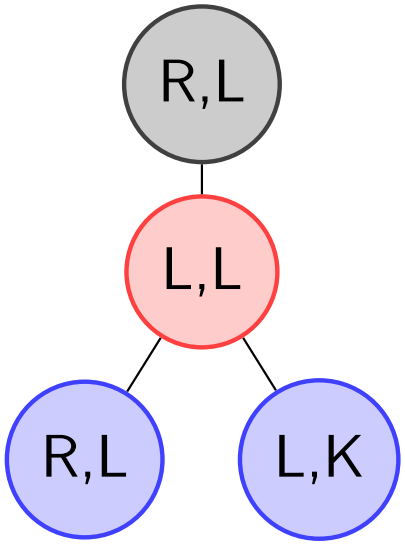


Beispiel Suche

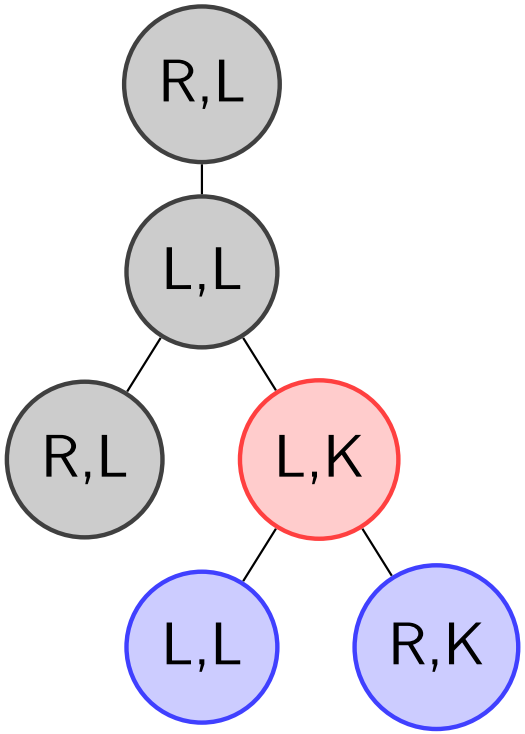
Beispiel Suche



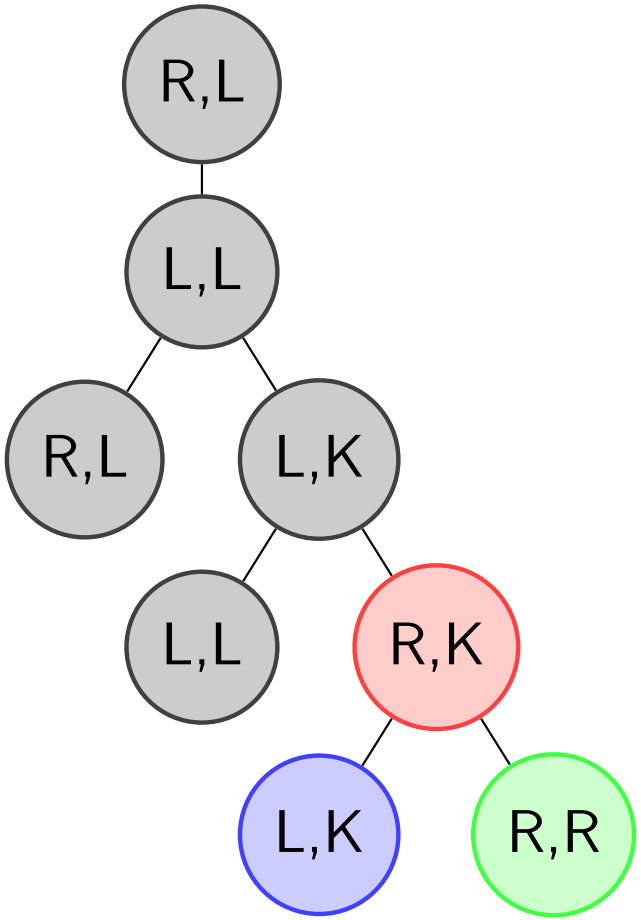
Beispiel Suche



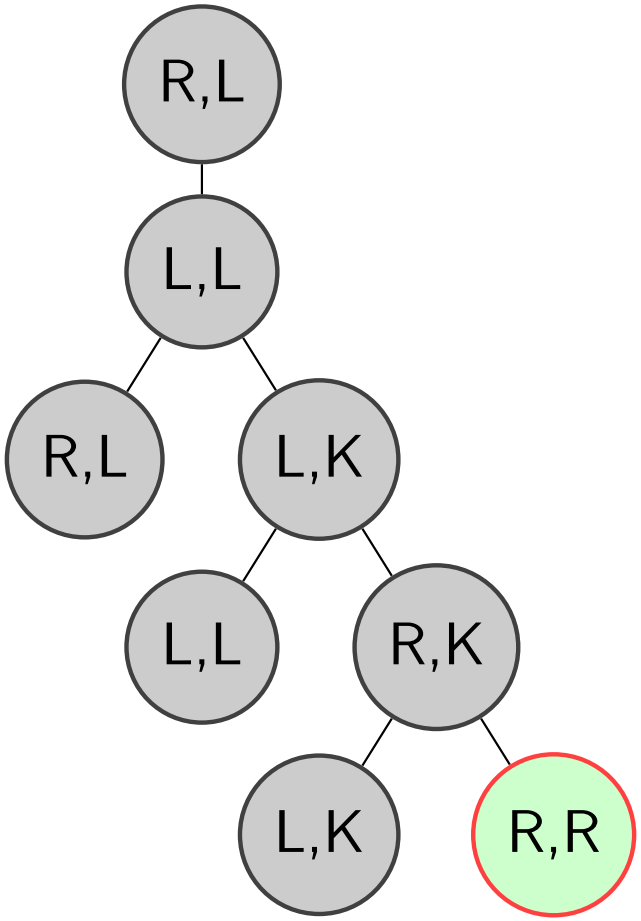
Beispiel Suche



Beispiel Suche



Beispiel Suche



Pattern-Datenbanken

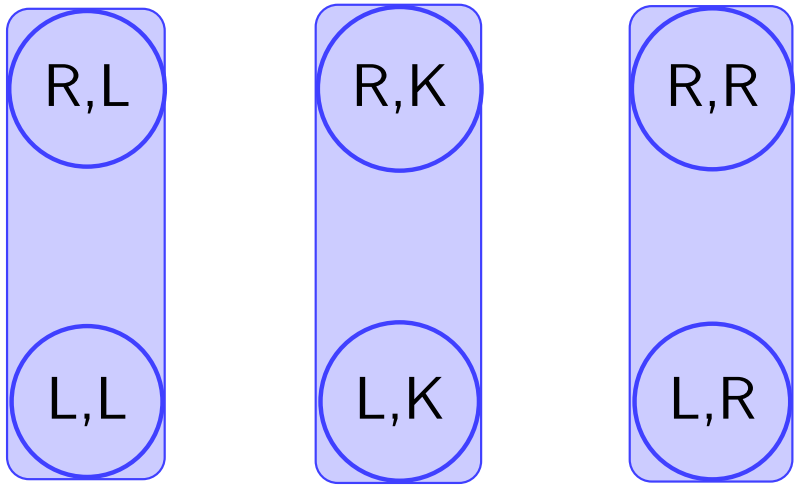
Pattern-Datenbank

Eine **Pattern-Datenbank** ist eine Vereinfachung eines Zustandsraumes, die man mittels einer Projektion auf einen Unterraum erhält.

Pattern

Man bezeichnet als **Pattern** der Pattern-Datenbank die Menge der Variablen des Unterraumes, auf den projiziert wird.

Beispiel



Potenzialheuristiken

Fakten

Fakten sind Variablen-Werte-Paare. Ein Fakt wird als wahr betrachtet, genau dann wenn die Variable den gegebenen Wert hat.

Potenzialheuristik

Eine **Potenzialheuristik** ist eine Funktion, die aus dem Zustandsraum auf reelle, positive Zahlen, sowie Unendlich abbildet ($S \rightarrow \mathbb{R}_0^+ \cup \infty$). Hierbei wird eine Menge Fakten definiert und zu jedem ein sog. **Potenzial** bestimmt. Die Funktion berechnet sich schliesslich als Summe der Potenziale, deren Fakten wahr sind.

Konkrete Aufgabe

- Die Fehlerquadrate zwischen h-Werten einer Sammlung von Pattern-Datenbanken und einer Potenzialheuristik minimieren
- Verschiedene Pattern-Generatoren und Konfigurationen ausprobieren und allenfalls modifizieren

Greedy Algorithmus

Die Zahl x, die die Summe der Fehlerquadrate zu einer Menge von Zahlen $\{x_i | i \in \mathbb{N}; i \leq n\}$ minimiert, ist deren Durchschnitt $\frac{1}{n} \sum_{i=1}^{n} x_i$.

Greedy Algorithmus

Die Zahl x, die die Summe der Fehlerquadrate zu einer Menge von Zahlen $\{x_i | i \in \mathbb{N}; i \leq n\}$ minimiert, ist deren Durchschnitt $\frac{1}{n} \sum_{i=1}^{n} x_i$.

- **Grundidee**: Kleinste Fehlerquadrate iterativ annähern
- **Vorgehen**: Potenziale einzeln korrigieren während alle anderen Potenziale als gegeben betrachtet werden
- **Iteration**: Immer das Potenzial korrigieren, welches die grösste Korrektur benötigt

Die Korrektur eines einzelnen Potenzials wird hierbei als ein Iterationsschritt betrachtet.

Beispiel

Gegeben seien folgende Pattern-Datenbanken:

$v_0 \setminus v_1$	0	1	2
0	0	1	2
1	1	3	2

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	1	3	2
2	3	3	5

Beispiel

$v_0 \setminus v_1$	0	1	2
0	0	1	2
1	1	3	2

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	1	3	2
2	3	3	5

Durchschnitte der Differenzen:

$$
\begin{aligned}
v_0 = 0: &\quad 1 \\
v_0 = 1: &\quad 2 \\
v_1 = 0: &\quad \tfrac{1}{5} \\
v_1 = 1: &\quad 2 \\
v_1 = 2: &\quad 3 \\
v_2 = 0: &\quad 1 \\
v_2 = 1: &\quad 2 \\
v_2 = 2: &\quad 2
\end{aligned}
$$

Beispiel

$v_0 \setminus v_1$	0	1	2
0	0	1	2
1	1	3	2

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	1	3	2
2	3	3	5

$$
\begin{aligned}
v_0 = 0: &\quad 1 \\
v_0 = 1: &\quad 2 \\
v_1 = 0: &\quad \tfrac{1}{5} \\
v_1 = 1: &\quad 2 \\
v_1 = 2: &\quad 3 \\
v_2 = 0: &\quad 1 \\
v_2 = 1: &\quad 2 \\
v_2 = 2: &\quad 2
\end{aligned}
$$

Potenzialvektor:

$$
\vec{p} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 3 \\ 0 \\ 0 \\ 0 \end{pmatrix}
$$

Beispiel

2. Iterationsschritt:

$v_0 \setminus v_1$	0	1	2
0	0	1	-1
1	1	3	-1

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	1	3	2
2	0	0	2

Beispiel

2. Iterationsschritt:

$v_0 \setminus v_1$	0	1	2
0	0	1	-1
1	1	3	-1

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	1	3	2
2	0	0	2

$$
\begin{aligned}
v_0 = 0: &\quad 0 \\
v_0 = 1: &\quad 1 \\
v_1 = 0: &\quad \tfrac{1}{5} \\
v_1 = 1: &\quad 2 \\
v_1 = 2: &\quad 0 \\
v_2 = 0: &\quad 0 \\
v_2 = 1: &\quad 1 \\
v_2 = 2: &\quad 1
\end{aligned}
$$

Beispiel

2. Iterationsschritt:

$v_0 \setminus v_1$	0	1	2
0	0	1	-1
1	1	3	-1

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	1	3	2
2	0	0	2

$$
\begin{aligned}
v_0 = 0: &\quad 0 \\
v_0 = 1: &\quad 1 \\
v_1 = 0: &\quad \tfrac{1}{5} \\
v_1 = 1: &\quad 2 \\
v_1 = 2: &\quad 0 \\
v_2 = 0: &\quad 0 \\
v_2 = 1: &\quad 1 \\
v_2 = 2: &\quad 1
\end{aligned}
\qquad
\vec{p} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 3 \\ 0 \\ 0 \\ 0 \end{pmatrix}
$$

Beispiel

3. Iterationsschritt:

$v_0 \setminus v_1$	0	1	2
0	0	-1	-1
1	1	1	-1

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	-1	1	0
2	0	0	2

Beispiel

3. Iterationsschritt:

$v_0 \backslash v_1$	0	1	2
0	0	-1	-1
1	1	1	-1

$v_1 \backslash v_2$	0	1	2
0	0	0	0
1	-1	1	0
2	0	0	2

$$
\begin{aligned}
v_0 = 0: &\quad -\tfrac{2}{3} \\
v_0 = 1: &\quad \tfrac{1}{3} \\
v_1 = 0: &\quad \tfrac{1}{5} \\
v_1 = 1: &\quad 0 \\
v_1 = 2: &\quad 0 \\
v_2 = 0: &\quad -\tfrac{2}{3} \\
v_2 = 1: &\quad \tfrac{1}{3} \\
v_2 = 2: &\quad \tfrac{1}{3}
\end{aligned}
$$

Beispiel

3. Iterationsschritt:

$v_0 \backslash v_1$	0	1	2
0	0	-1	-1
1	1	1	-1

$v_1 \backslash v_2$	0	1	2
0	0	0	0
1	-1	1	0
2	0	0	2

$$
\begin{aligned}
v_0 = 0: &\quad -\tfrac{2}{3} \\
v_0 = 1: &\quad \tfrac{1}{3} \\
v_1 = 0: &\quad \tfrac{1}{5} \\
v_1 = 1: &\quad 0 \\
v_1 = 2: &\quad 0 \\
v_2 = 0: &\quad -\tfrac{2}{3} \\
v_2 = 1: &\quad \tfrac{1}{3} \\
v_2 = 2: &\quad \tfrac{1}{3}
\end{aligned}
$$

$$
\vec{p} = \begin{pmatrix} -\frac{2}{3} \\ 0 \\ 0 \\ 2 \\ 3 \\ 0 \\ 0 \\ 0 \end{pmatrix}
$$

Beispiel

4. Iterationsschritt:

$v_0 \setminus v_1$	0	1	2
0	$\frac{2}{3}$	$-\frac{1}{3}$	$-\frac{1}{3}$
1	1	1	-1

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	-1	1	0
2	0	0	2

Beispiel

4. Iterationsschritt:

$v_0 \backslash v_1$	0	1	2
0	$\frac{2}{3}$	$-\frac{1}{3}$	$-\frac{1}{3}$
1	1	1	-1

$v_1 \backslash v_2$	0	1	2
0	0	0	0
1	-1	1	0
2	0	0	2

$$
\begin{aligned}
v_0 = 0: &\quad 0 \\
v_0 = 1: &\quad \frac{1}{3} \\
v_1 = 0: &\quad \frac{1}{3} \\
v_1 = 1: &\quad \frac{2}{15} \\
v_1 = 2: &\quad \frac{2}{15} \\
v_2 = 0: &\quad -\frac{2}{3} \\
v_2 = 1: &\quad \frac{1}{3} \\
v_2 = 2: &\quad \frac{1}{3}
\end{aligned}
$$

Beispiel

4. Iterationsschritt:

$v_0 \setminus v_1$	0	1	2
0	$\frac{2}{3}$	$-\frac{1}{3}$	$-\frac{1}{3}$
1	1	1	-1

$v_1 \setminus v_2$	0	1	2
0	0	0	0
1	-1	1	0
2	0	0	2

$$
\begin{aligned}
v_0 = 0: &\quad 0 \\
v_0 = 1: &\quad \tfrac{1}{3} \\
v_1 = 0: &\quad \tfrac{1}{3} \\
v_1 = 1: &\quad \tfrac{2}{15} \\
v_1 = 2: &\quad \tfrac{2}{15} \\
v_2 = 0: &\quad -\tfrac{2}{3} \\
v_2 = 1: &\quad \tfrac{1}{3} \\
v_2 = 2: &\quad \tfrac{1}{3}
\end{aligned}
$$

$$
\vec{p} = \begin{pmatrix} -\frac{2}{3} \\ 0 \\ 0 \\ 2 \\ 3 \\ -\frac{2}{3} \\ 0 \\ 0 \end{pmatrix}
$$

Algebraischer Algorithmus - Grundlagen

Matrix der Potenzialfunktion

Sei $\vec{p}$ der Vektor der Potenziale und $\vec{h}$ der Vektor mit den h-Werten, so gibt es eine Matrix A mit: $A\vec{p} = \vec{h}$.

Einträge der Matrix

Die Zeilen dieser Matrix entsprechen hierbei Zuständen und sie enthalten dort eine 1, wo das Zugehörige Fakt wahr ist und eine 0 sonst.

Reduktion auf eine Pattern-Datenbank

Es macht Sinn $\vec{p}$ und A auf Spalten und Zeilen zu reduzieren, die für die PDb relevant sind. Dieser Vektor bzw. diese Matrix wird hier mit $\vec{p}_{pdb}$ bzw. A_{pdb} bezeichnet.

Kleinste Fehlerquadrate für eine Pattern-Datenbank

Kleinste Fehlerquadrate

Die Lösung für kleinste Fehlerqudrate einer Matrix-Gleichung $A\vec{x} = \vec{b}$ berechnet sich als Lösung der Gleichung $A^\top A\vec{x} = A^\top \vec{b}$.

Gleichung für eine Pattern-Datenbank

Für eine einzelne Pattern-Datenbank muss demnach die Lösung der Gleichung $A_{pdb}^\top A_{pdb}\vec{p}_{pdb} = A_{pdb}^\top \vec{h}_{pdb}$ berechnet werden. Hierbei bezeichnet $\vec{h}_{pdb}$ den Vektor der h-Werte aus der Pattern-Datenbank.

Algebraischer Algorithmus

- **Grundidee**: Die Lösung gemäss kleinster Fehlerquadrate für jede Pattern-Datenbank algebraisch berechnen
- **Vorgehen**: $A_{pdb}^\top A_{pdb}$ sowie $A_{pdb}^\top \vec{h}_{pdb}$ (direkt) berechnen und dann den Durchschnitt der Lösungen aller Gleichungen $A_{pdb}^\top A_{pdb} \vec{p}_{pdb} = A^\top \vec{h}_{pdb}$ als Potenziale festlegen

Die Matrix $A_{pdb}^{\top} A_{pdb}$

- Die Einträge von $A_{pdb}^{\top} A_{pdb}$ sind die Skalarprodukte der Spaltenvektoren von A_{pdb}.
- Somit sind die Einträge die Anzahl abstrakter Zustände, in denen die zugehörigen Fakten wahr sind.
- Die Anzahl abstrakter Zuständen einer Pattern-Datenbank n berechnet sich über $n = \prod_{v_i \in P} |dom(v_i)|$.
- Für jede Variable v_i kann immer nur genau ein Fakt wahr sein. Es gibt immer $n_i = \frac{n}{|dom(v_i)|}$ abstrakte Zustände in denen ein Fakt von v_i gilt.
- Für zwei Variablen v_i, v_j wird dies: $n_{i,j} = \frac{n_i}{|dom(v_j)|}$.

Die Matrix $A_{pdb}^{\top} A_{pdb}$

Somit hat die Matrix $A_{pdb}^{\top} A_{pdb}$ Blockgestalt. Die Diagonalblöcke $B_{i,i}$ gehören zu je einer Variablen v_i und sehen wie folgt aus:

$$B_{i,i} = E \cdot n_i$$

Nicht-Diagonalblöcke $B_{i,j}$ gehören zu den Variablen v_i sowie v_j und alle ihre Einträge sind gleich $n_{i,j}$.

$$B_{i,j} = [n_{i,j}]$$

Beispiel

Sei $P = \{v_0, v_1\}$ das Pattern einer Pattern-Datenbank. Seien $|dom(v_0)| = 2$ und $|dom(v_1)| = 3$. So gilt:

$$A_{pdb} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}; A_{pdb}^{\top} A_{pdb} = \begin{pmatrix} 3 & 0 & 1 & 1 & 1 \\ 0 & 3 & 1 & 1 & 1 \\ 1 & 1 & 2 & 0 & 0 \\ 1 & 1 & 0 & 2 & 0 \\ 1 & 1 & 0 & 0 & 2 \end{pmatrix}$$

Die Lösung von $A_{pdb}^{\top} A_{pdb} \vec{p}_{pdb} = A^{\top} \vec{h}_{pdb}$

Die Lösung einer Matrixgleichung kann mittels Gauss-Algorithmus bestimmt werden.

Gauss-Algorithmus:

- Rechenaufwand der Ordnung $\mathcal{O}(N^3)$, wobei N die Anzahl Zeilen bzw. Spalten der Matrix ist
- Separation kann Abhilfe schaffen
- Im Fall der Matrix $A_{pdb}^{\top} A_{pdb}$ ist eine solche Separation möglich

Separation der Diagonalblöcke

- Alle Zeilen eines beliebigen Zeilenblocks addieren
- Es entsteht stets dieselbe Zeile $\vec{z}$, mit n_i im i-ten Spaltenblock

Separation der Diagonalblöcke

- Alle Zeilen eines beliebigen Zeilenblocks addieren
- Es entsteht stets dieselbe Zeile $\vec{z}$, mit n_i im i-ten Spaltenblock
- Alle anderen Zeilen eines i-ten Zeilenblocks mit $|dom(v_i)|$ multiplizieren
- Es entsteht eine Zeile, die bis auf die Einträge des Diagonalblocks mit $\vec{z}$ identisch ist.

Separation der Diagonalblöcke

- Alle Zeilen eines beliebigen Zeilenblocks addieren
- Es entsteht stets dieselbe Zeile $\vec{z}$, mit n_i im i-ten Spaltenblock
- Alle anderen Zeilen eines i-ten Zeilenblocks mit $|dom(v_i)|$ multiplizieren
- Es entsteht eine Zeile, die bis auf die Einträge des Diagonalblocks mit $\vec{z}$ identisch ist.
- Damit lassen sich alle Diagonalblöcke, bis auf einen, aus dem Gleichungssystem separieren.

Beispiel

Sei $P = \{v_0, v_1, v_2\}$ das Pattern einer Pattern-Datenbank. Seien $|dom(v_0)| = 2$, $|dom(v_1)| = 3$ und $|dom(v_2)| = 3$. So ist:

$$A_{pdb}^{\top} A_{pdb} = \begin{pmatrix} 9 & 0 & 3 & 3 & 3 & 3 & 3 & 3 \\ 0 & 9 & 3 & 3 & 3 & 3 & 3 & 3 \\ 3 & 3 & 6 & 0 & 0 & 2 & 2 & 2 \\ 3 & 3 & 0 & 6 & 0 & 2 & 2 & 2 \\ 3 & 3 & 0 & 0 & 6 & 2 & 2 & 2 \\ 3 & 3 & 2 & 2 & 2 & 6 & 0 & 0 \\ 3 & 3 & 2 & 2 & 2 & 0 & 6 & 0 \\ 3 & 3 & 2 & 2 & 2 & 0 & 0 & 6 \end{pmatrix} \rightarrow \begin{pmatrix} 9 & 0 & 3 & 3 & 3 & 3 & 3 & 3 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 3 & 3 & 6 & 0 & 0 & 2 & 2 & 2 \\ 3 & 3 & 0 & 6 & 0 & 2 & 2 & 2 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 3 & 3 & 2 & 2 & 2 & 6 & 0 & 0 \\ 3 & 3 & 2 & 2 & 2 & 0 & 6 & 0 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \end{pmatrix} \rightarrow$$

Beispiel

$$\begin{pmatrix} 9 & 0 & 3 & 3 & 3 & 3 & 3 & 3 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 3 & 3 & 6 & 0 & 0 & 2 & 2 & 2 \\ 3 & 3 & 0 & 6 & 0 & 2 & 2 & 2 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 3 & 3 & 2 & 2 & 2 & 6 & 0 & 0 \\ 3 & 3 & 2 & 2 & 2 & 0 & 6 & 0 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \end{pmatrix} \rightarrow \begin{pmatrix} 18 & 0 & 6 & 6 & 6 & 6 & 6 & 6 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 9 & 9 & 18 & 0 & 0 & 6 & 6 & 6 \\ 9 & 9 & 0 & 18 & 0 & 6 & 6 & 6 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 9 & 9 & 6 & 6 & 6 & 18 & 0 & 0 \\ 9 & 9 & 6 & 6 & 6 & 0 & 18 & 0 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \end{pmatrix} \rightarrow$$

Beispiel

$$\begin{pmatrix} 18 & 0 & 6 & 6 & 6 & 6 & 6 & 6 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 9 & 9 & 18 & 0 & 0 & 6 & 6 & 6 \\ 9 & 9 & 0 & 18 & 0 & 6 & 6 & 6 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \\ 9 & 9 & 6 & 6 & 6 & 18 & 0 & 0 \\ 9 & 9 & 6 & 6 & 6 & 0 & 18 & 0 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \end{pmatrix} \rightarrow$$

$$\begin{pmatrix} 9 & -9 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 12 & -6 & -6 & 0 & 0 & 0 \\ 0 & 0 & -6 & 12 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 12 & -6 & -6 \\ 0 & 0 & 0 & 0 & 0 & -6 & 12 & -6 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \end{pmatrix} \rightarrow$$

Beispiel

$$\begin{pmatrix} 9 & -9 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 12 & -6 & -6 & 0 & 0 & 0 \\ 0 & 0 & -6 & 12 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 12 & -6 & -6 \\ 0 & 0 & 0 & 0 & 0 & -6 & 12 & -6 \\ 9 & 9 & 6 & 6 & 6 & 6 & 6 & 6 \end{pmatrix} \rightarrow \begin{pmatrix} 9 & 0 & 0 & 0 & 0 & 0 \\ 0 & 12 & -6 & 0 & 0 & 0 \\ 0 & -6 & 12 & 0 & 0 & 0 \\ 0 & 0 & 0 & 12 & -6 & -6 \\ 0 & 0 & 0 & -6 & 12 & -6 \\ 9 & 6 & 6 & 6 & 6 & 6 \end{pmatrix}$$

Überblick über die Experimente

Experimente wurden auf einer Satisficing-Suite der Fast-Downward-Benchmarks durchgeführt.

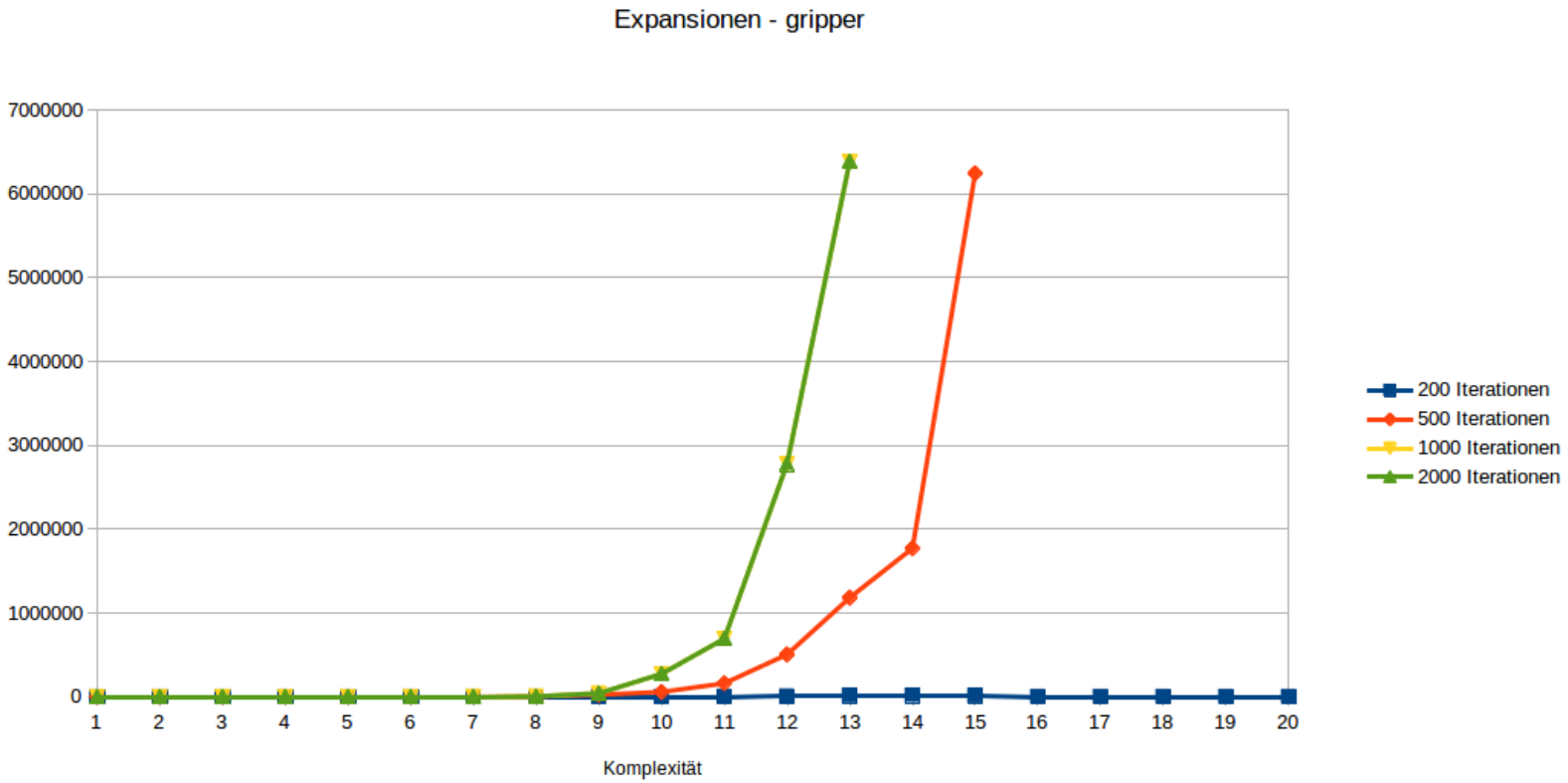
Fragestellungen

- Qualität der Heuristik des Greedy-Algorithmus in Abhängigkeit der Anzahl Iterationen
- Eignung verschiedener Pattern-Generatoren
- Vergleich mit der FF-Heuristik

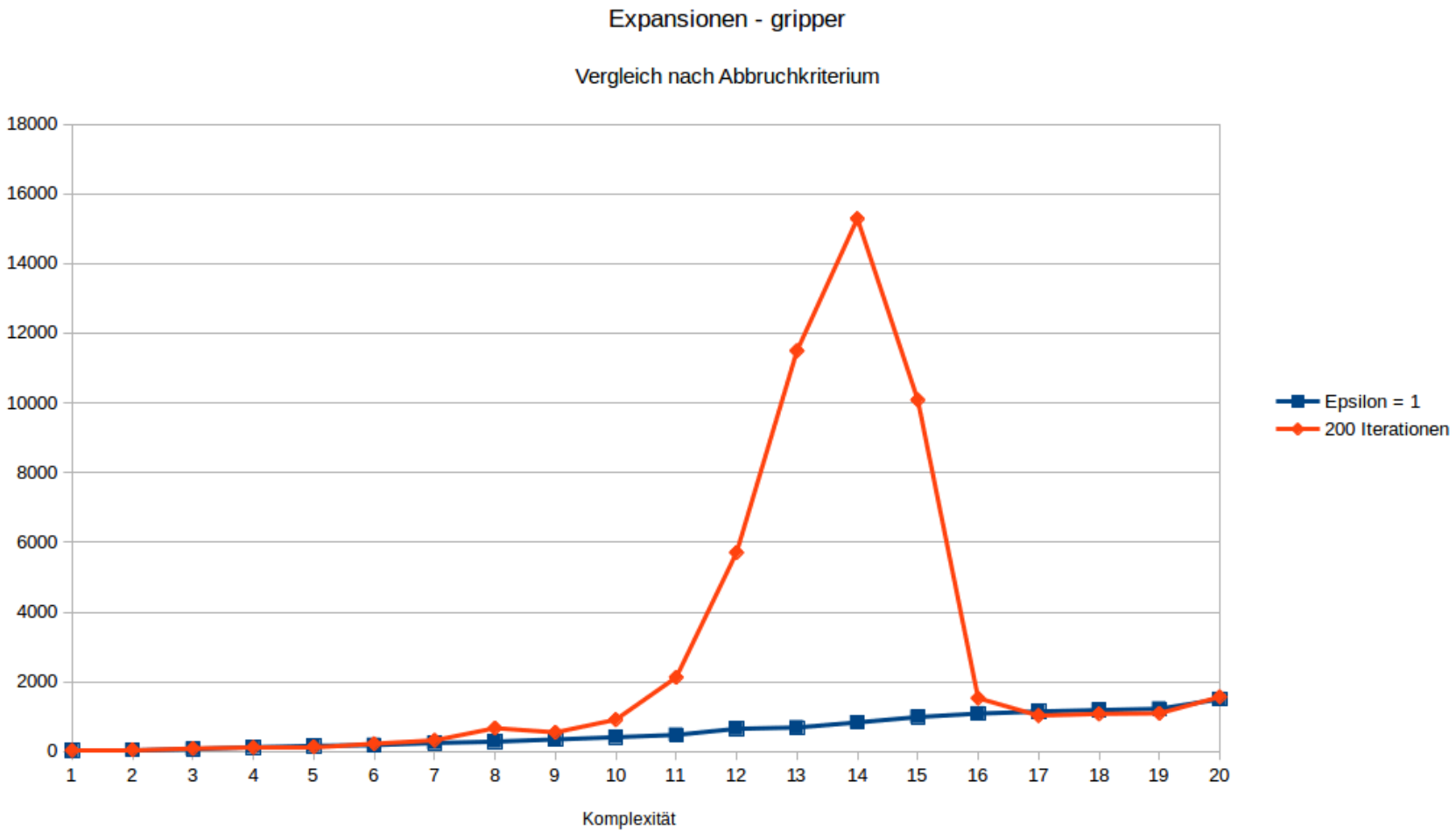
Anzahl Iterationen des Greedy Algorithmus

- Meist eine Zunahme der Qualität mit mehr Iterationen
- In einzelnen Problemklassen jedoch sinkt die Qualität signifikant ab einer gewissen Anzahl Iterationen
- Verschiebung dieses Punktes zu mehr Iterationen bei komplexeren Problemen
- Inkohärenter Verlauf der Qualität über die Komplexität der Probleme

Anzahl Iterationen des Greedy Algorithmus



Anzahl Iterationen des Greedy Algorithmus



Vergleich von Pattern-Generatoren

Coverage				
Pattern-Generator	Systematic		Hillclimbing	
Algorithmus	Greedy	Algebraic	Greedy	Algebraic
Summe	908	721	476	579

Vergleich mit der FF-Heuristik

Coverage			
Konfiguration	Greedy-best	Algebraic-best	FF
Coverage - Total	908	721	**1115**

- FF-Heuristik stärker als die hier entwickelten Heuristiken
- Greedy-Algorithmus stärker als der algebraische

Schlussfolgerungen

- Greedy-Algorithmus mit Fehlertoleranz steuern
- Teils Unterschiede in der Qualität der Information
- Hillclimbing mit algebraischem Algorithmus
- Systematic mit Greedy-Algorithmus
- Noch einiges Verbesserungspotenzial vorhanden

Weiterführende Arbeit

- Gewichtung der abstrakten Zustände bzw. der Patterns
- Statistische Analyse der h-Werte der Pattern-Datenbanken
- Handhabe von Dead-Ends
- Ergründen weshalb mit weniger Iterationen teils bessere Resultate erzielt werden können
- Andere Heuristiken als Grundlage für eine Potenzialheuristik

Literaturverzeichnis

S. Edelkamp.

Planning with pattern databases.
In A. Cesta and D. Borrajo, editors, *Proceedings of the Sixth European Conference on Planning (ECP 2001)*, pages 84–90. AAAI Press, 2001.

P. Haslum, A. Botea, M. Helmert, B. Bonet, and S. Koenig.

Domain-independent construction of pattern database heuristics for cost-optimal planning.
In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI 2007)*, pages 1007–1012. AAAI Press, 2007.

M. Helmert.

The fast downward planning system.
Journal of Artifcial Intelligence Research, 26:191–246, 2009.

F. Pommerening, M. Helmert, G. Röger, and J. Seipp.

From non-negative to general operator cost partitioning.
In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI 2015)*, pages 3335–3341. AAAI Press, 2015.

F. Pommerening, G. Röger, and M. Helmert.

Getting the most out of pattern databases for classical planning.
In F. Rossi, editor, *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, pages 2357–2364, 2013.

Fragen