

Was bisher geschah

- ▶ Daten, Information, Wissen
- ▶ Wissensrepräsentation und -verarbeitung
- ▶ Wissensbasierte Systeme

Wissensrepräsentation:

- ▶ Zustandsübergangssystem:
Graph mit markierten Knoten
(Zustände und deren Eigenschaften)
- ▶ Startzustand
- ▶ Eigenschaften der Zielzustände

Lösung: Pfad vom Start- zu einem Zielzustand

Wissensverarbeitung: Suche im Graphen

- uninformiert: Breiten-, Tiefen-, Gleiche-Kosten-Suche
- informiert: Heuristik, Greedy-, A*-Suche

Zwei-Personen-Spiele

Brettspiel

- ▶ aktueller Spielzustand immer für beide Spieler sichtbar (vollständige Information)
- ▶ einer gewinnt, der andere verliert (Nullsummenspiel)

Wissensrepräsentation (Spielbaum):

- ▶ Menge von Zuständen (Min- und Max-Zustände)
- ▶ Startzustand
- ▶ Endzustände (ohne Fortsetzung)
- ▶ Nachfolgermenge $S(v)$ = Menge von Zuständen (nach zulässigen Zügen)
- ▶ Bewertungsfunktion: Menge der Endzustände $\rightarrow \mathbb{Z}$
 - ▶ positiv: Spieler (1, Max, beginnt) gewinnt
 - ▶ negativ: Gegner (0, Min) gewinnt

Beispiel Nim (Variante)

- ▶ n Münzen auf einem Stapel
- ▶ Spielzug: Teilen eines Stapels in zwei nichtleere Stapel ungleicher Größe
- ▶ Sobald ein Spieler keinen Zug mehr ausführen kann, hat er verloren (und der andere gewonnen).

(eine mögliche) Modellierung als Zustandsübergangssystem:

Zustände: $S : \mathbb{N} \rightarrow \mathbb{N}$ (Multimenge)

Münzanzahl \mapsto Anzahl der Stapel mit dieser Zahl an Münzen

Startzustand: $S(n) = 1 \wedge \forall i \neq n : S(i) = 0$

Endzustände: kein Zug möglich

Übergänge: (erlaubte Züge) für $x = x_1 + x_2 \wedge x_1 \neq x_2 \wedge x_1 x_2 \neq 0$:

$S \rightarrow S'$ mit

$$S'(x) = S(x) - 1$$

$$\wedge S'(x_1) = S(x_1) + 1 \wedge S'(x_2) = S(x_2) + 1$$

$$\wedge \forall i \in \mathbb{N} \setminus \{x, x_1, x_2\} : S'(i) = S(i)$$

Minimax-Werte in vollständigen Spielbäume

- ▶ vollständiger Spielbaum $B = (V, E)$
- ▶ Bewertung der Endzustände (Blätter im Spielbaum) bekannt
- ▶ Fortsetzung der Bewertungsfunktion von den Blättern auf alle Knoten im Spielbaum $b : V \rightarrow \mathbb{Z}$

rekursive Berechnung (Minimax-Algorithmus) des Wertes eines Knotens v im Spielbaum:

$$m(v) = \begin{cases} b(v) & \text{falls } v \text{ Endzustand} \\ \max\{m(u) \mid u \in S(v)\} & \text{falls } v \text{ Max-Knoten} \\ \min\{m(u) \mid u \in S(v)\} & \text{falls } v \text{ Min-Knoten} \end{cases}$$

Beispiele (Tafel):

- ▶ Spielbaum,
- ▶ Nim mit $n = 7$

Spielstrategie für Spieler 1 (Max):

Zug wählen, der zum Zustand mit höchstem Minimax-Wert führt

Minimax-Werte mit Heuristik

bei unvollständigem Spielbaum: Kombination von

- ▶ heuristischer Knotenbewertung
- ▶ Berechnung der Minimax-Werte

Beispiele (Tafel): Tic-Tac-Toe

mit Schätzfunktion für den Spieler am Zug:

Differenz der Anzahlen der noch nicht blockierten Gewinntripel

auch dabei Spielstrategie für Spieler 1 (Max):

Zug wählen, der zum Zustand mit höchstem Minimax-Wert führt

α - β -Suche

Idee: Tiefensuche mit Verwaltung zusätzlicher Werte

α : bisher höchster Minimax-Wert an Max-Positionen

β : bisher geringster Minimax-Wert an Min-Positionen

Bei Berechnung des Minimax-Wertes der Wurzel Berechnungen für Teilbäume abbrechen, sobald bekannt ist, dass sie α und β nicht verbessern

Abtrennen jedes Kindes v eines

min-Knotens u , falls $\beta(u) \leq \alpha(v)$

(min-Spieler kann durch Wahl eines zuvor untersuchten Kindes von u den geringeren Minimax-Wert $\beta(u)$ erreichen als durch Wahl von v)

max-Knotens u , falls $\alpha(u) \geq \beta(v)$

(max-Spieler kann durch Wahl eines zuvor untersuchten Kindes von u den höheren Minimax-Wert $\alpha(u)$ erreichen als durch Wahl von v)

Beispiel (Tafel)