

Grundlagen der Künstlichen Intelligenz

39. Brettspiele: Alpha-Beta-Suche und Ausblick

Malte Helmert

Universität Basel

23. Mai 2014

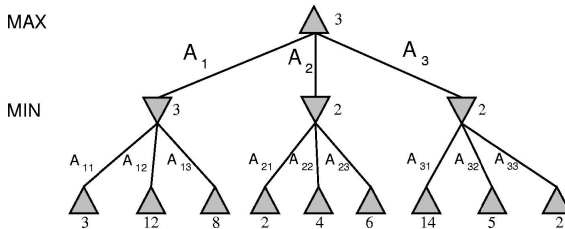
Brettspiele: Überblick

Kapitelüberblick:

- 38. Einführung und Minimax-Suche
- 39. Alpha-Beta-Suche und Ausblick

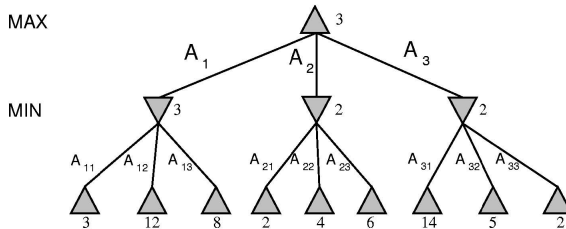
Alpha-Beta-Suche

Alpha-Beta-Suche



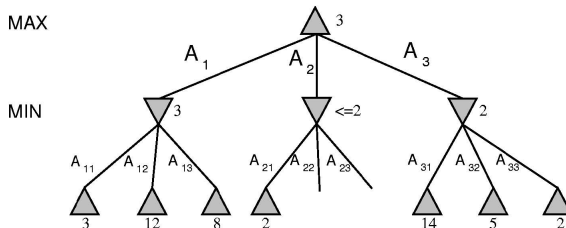
Kann man Sucharbeit sparen?

Alpha-Beta-Suche

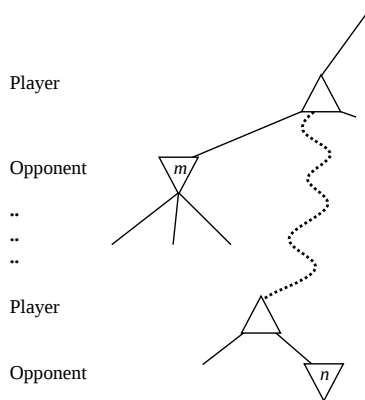


Kann man Sucharbeit sparen?

Nicht alle Knoten müssen betrachtet werden!



Alpha-Beta-Suche: allgemein



Falls $m > n$ gilt, wird Knoten mit Nutzen in n bei perfektem Spiel nie erreicht!

Alpha-Beta-Suche: Idee

Idee: Führe in Minimax-Tiefensuche zwei Zahlenwerte α und β mit, so dass für jeden rekursiven Aufruf gilt:

Alpha-Beta-Suche: Idee

Idee: Führe in Minimax-Tiefensuche zwei Zahlenwerte α und β mit, so dass für jeden rekursiven Aufruf gilt:

- Wenn der Nutzen im aktuellen Teilbaum $\leq \alpha$ ist,
interessiert er uns nicht, weil MAX dann bei perfektem Spiel diesen Teilbaum nicht betreten muss
- Wenn der Nutzen im aktuellen Teilbaum $\geq \beta$ ist,
interessiert er uns nicht, weil MIN dann bei perfektem Spiel diesen Teilbaum nicht betreten muss

Alpha-Beta-Suche: Idee

Idee: Führe in Minimax-Tiefensuche zwei Zahlenwerte α und β mit, so dass für jeden rekursiven Aufruf gilt:

- Wenn der Nutzen im aktuellen Teilbaum $\leq \alpha$ ist, **interessiert er uns nicht**, weil MAX dann bei perfektem Spiel diesen Teilbaum nicht betreten muss
- Wenn der Nutzen im aktuellen Teilbaum $\geq \beta$ ist, **interessiert er uns nicht**, weil MIN dann bei perfektem Spiel diesen Teilbaum nicht betreten muss

Wenn $\alpha \geq \beta$ im Teilbaum gilt, ist der Teilbaum uninteressant und muss nicht weiter durchsucht werden (**α - β -Pruning**).

Alpha-Beta-Suche: Idee

Idee: Führe in Minimax-Tiefensuche zwei Zahlenwerte α und β mit, so dass für jeden rekursiven Aufruf gilt:

- Wenn der Nutzen im aktuellen Teilbaum $\leq \alpha$ ist, **interessiert er uns nicht**, weil MAX dann bei perfektem Spiel diesen Teilbaum nicht betreten muss
- Wenn der Nutzen im aktuellen Teilbaum $\geq \beta$ ist, **interessiert er uns nicht**, weil MIN dann bei perfektem Spiel diesen Teilbaum nicht betreten muss

Wenn $\alpha \geq \beta$ im Teilbaum gilt, ist der Teilbaum uninteressant und muss nicht weiter durchsucht werden (**α - β -Pruning**).

Wird Alpha-Beta-Suche mit $\alpha = -\infty$ und $\beta = +\infty$ gestartet, ist Ergebnis **identisch** zu Minimax bei weniger Suchaufwand.

Alpha-Beta-Suche: Pseudo-Code

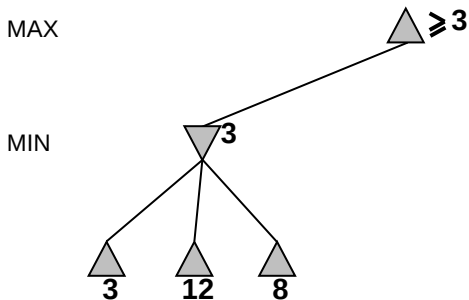
```
function ALPHA-BETA-SEARCH(state) returns an action  
   $v \leftarrow \text{MAX-VALUE}(\text{state}, -\infty, +\infty)$   
  return the action in  $\text{ACTIONS}(\text{state})$  with value  $v$ 
```

```
function MAX-VALUE(state,  $\alpha$ ,  $\beta$ ) returns a utility value  
  if  $\text{TERMINAL-TEST}(\text{state})$  then return  $\text{UTILITY}(\text{state})$   
   $v \leftarrow -\infty$   
  for each  $a$  in  $\text{ACTIONS}(\text{state})$  do  
     $v \leftarrow \text{MAX}(v, \text{MIN-VALUE}(\text{RESULT}(s, a), \alpha, \beta))$   
    if  $v \geq \beta$  then return  $v$   
     $\alpha \leftarrow \text{MAX}(\alpha, v)$   
  return  $v$ 
```

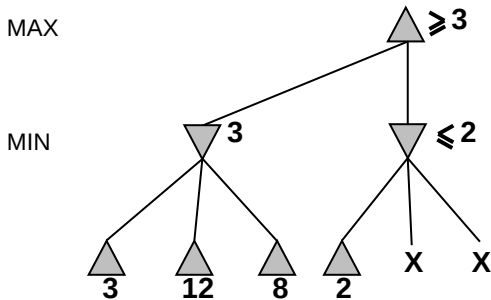
```
function MIN-VALUE(state,  $\alpha$ ,  $\beta$ ) returns a utility value  
  if  $\text{TERMINAL-TEST}(\text{state})$  then return  $\text{UTILITY}(\text{state})$   
   $v \leftarrow +\infty$   
  for each  $a$  in  $\text{ACTIONS}(\text{state})$  do  
     $v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(\text{RESULT}(s, a), \alpha, \beta))$   
    if  $v \leq \alpha$  then return  $v$   
     $\beta \leftarrow \text{MIN}(\beta, v)$   
  return  $v$ 
```

Anmerkung: bei mehreren identisch bewerteten Zügen
muss ALPHA-BETA-SEARCH den **ersten** Maximierer wählen.

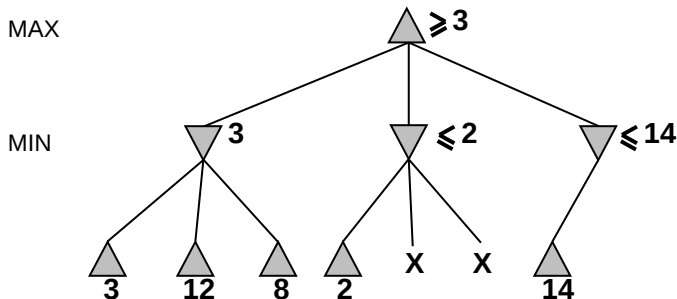
Alpha-Beta-Suche: Beispiel



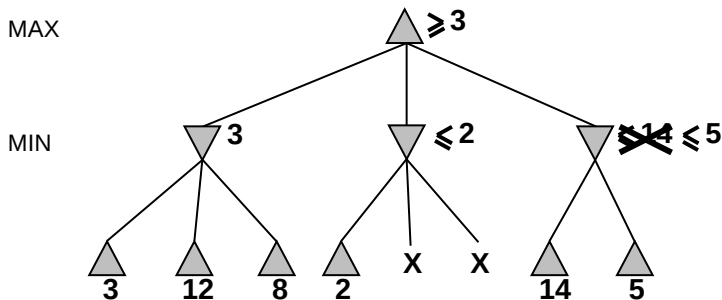
Alpha-Beta-Suche: Beispiel



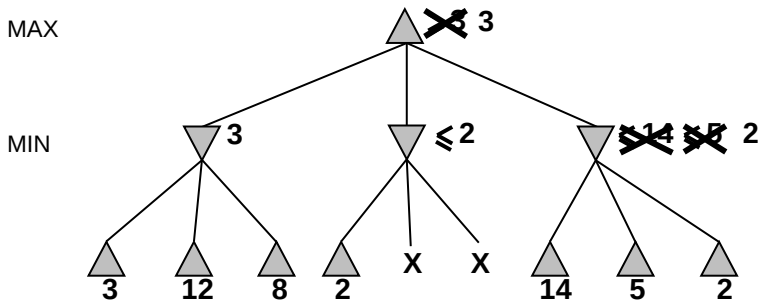
Alpha-Beta-Suche: Beispiel



Alpha-Beta-Suche: Beispiel



Alpha-Beta-Suche: Beispiel



Wie viel bringt Alpha-Beta-Suche?

Annahme: uniformer Spielbaum, Tiefe d , Verzweigungsgrad $b \geq 2$;
MAX- und MIN-Positionen immer abwechselnd

- Alpha-Beta-Suche ist am erfolgreichsten, wenn **bester Zug** für den Spieler am Zug immer **als erstes** betrachtet wird
 - also ein maximierender Zug bei MAX,
ein minimierender Zug bei MIN

~> Aufwand reduziert sich in diesem besten Fall
von $O(b^d)$ (Minimax) auf $O(b^{d/2})$

~> doppelte Suchtiefe in derselben Zeit möglich

- in der Praxis kommt man oft sehr nah an das Optimum heran

Stand der Wissenschaft

Stand der Wissenschaft

einige bekannte Brettspiele:

- **Schach**: ~→ nächste Folien
- **Othello**: **Logistello** besiegt 1997 menschlichen Weltmeister; beste Computer-Spieler deutlich stärker als beste Menschen
- **Dame (Checkers)**: **Chinook** offizieller Weltmeister (seit 1994); 2007 Unbesiegbarkeit gezeigt (Spiel „gelöst“)
- **Go**: Die besten Programme (**Zen**, **Mogo**, **Crazystone**) verwenden Monte-Carlo-Techniken (UCT) und sind in etwa auf dem Niveau starker Amateure (1 kyu/1 dan)

Computerschach

Weltmeister Garri Kasparov 1997 durch **Deep Blue**
in 6 Partien 3,5 : 2,5 besiegt

- spezialisierte Schach-Hardware, 30 Knoten mit je 16 Chips
- Alpha-Beta-Suche (mit Erweiterungen)
- Eröffnungsdatenbank aus Millionen von Schachpartien

Heutzutage spielt normale PC-Hardware auf Weltmeisterniveau.

Computerschach: Zitate

Claude Shannon (1949)

The chess machine is an ideal one to start with, since

- 1 the problem is sharply defined both in allowed operations (the moves) and in the ultimate goal (checkmate),
- 2 it is neither so simple as to be trivial nor too difficult for satisfactory solution,
- 3 chess is generally considered to require “thinking” for skilful play, [...]
- 4 the discrete structure of chess fits well into the digital nature of modern computers.

Computerschach: Zitate

Claude Shannon (1949)

The chess machine is an ideal one to start with, since

- 1 the problem is sharply defined both in allowed operations (the moves) and in the ultimate goal (checkmate),
- 2 it is neither so simple as to be trivial nor too difficult for satisfactory solution,
- 3 chess is generally considered to require “thinking” for skilful play, [...]
- 4 the discrete structure of chess fits well into the digital nature of modern computers.

Alexander Kronrod (1965)

Chess is the drosophila of Artificial Intelligence.

Computerschach: noch ein Zitat

John McCarthy (1997)

In 1965, the Russian mathematician Alexander Kronrod said,
“Chess is the Drosophila of artificial intelligence.”

Computerschach: noch ein Zitat

John McCarthy (1997)

In 1965, the Russian mathematician Alexander Kronrod said,
“Chess is the *Drosophila* of artificial intelligence.”

However, computer chess has developed much as genetics might have if the geneticists had concentrated their efforts starting in 1910 on breeding racing *Drosophilae*. We would have some science, but mainly we would have very fast fruit flies.

Zusammenfassung

Zusammenfassung

- **Alpha-Beta-Suche** merkt sich, welchen Nutzen beide Spieler anderswo im Baum erzwingen können und vermeidet so viele unnötige Berechnungen
 - im besten Fall Aufwand $O(b^{d/2})$ bei uniformen Bäumen
- ~> doppelt so tiefe Suche wie bei Minimax möglich