

## Softwarepraktikum SS 2010

### Implementierung heuristischer Algorithmen für Brettspiele

### – Übungsblatt 3 –

Abgabe/Besprechung der Lösungen am 2.6.2010 im Seminarraum des Lehrstuhls

#### Aufgabe 1

Implementieren Sie das in der Besprechung vorgestellte *Paranoidsuchverfahren* mit variabler Tiefe (die Tiefe kann vom Benutzer oder Server vorgegeben werden), so dass der Client den optimalen Zug gemäß dieser Strategie ausführt. Verwenden Sie zur Bewertung der Blätter die von Ihnen in Aufgabe 2 der letzten Übung implementierte Bewertungsfunktion. Einstiegsliteratur finden Sie u.a. in [1].

#### Aufgabe 2

Implementieren Sie das in der Besprechung vorgestellte *Alpha-Beta-Pruning* Verfahren zum Abschneiden von „uninteressanten“ Teilästen im Suchbaum (für  $n$  Spieler). Sehen sie außerdem einen Kommandozeilenparameter vor, mit dem man das Alpha-Beta-Pruning an- und abstellen kann.

Zur Erinnerung (bei 2 Spielern):

**Definiere:**

- $\alpha$ : Die bisher beste Option für MAX-Knoten (Initialisierung mit  $-\infty$ )
- $\beta$ : Die bisher beste Option für MIN-Knoten (Initialisierung mit  $+\infty$ )

Immer wenn in einem neu untersuchten Teilbaum eine Option gefunden wird, die schlechter bzw. besser ist als die jeweiligen Alpha- bzw. Beta-Werte, wird die Suche dort abgebrochen.

**Verfahren:**

- Sei  $\alpha$  die Bewertung des Knoten  $X$  und  $\beta$  die Bewertung des Knotens  $X_k$ .
- Seien  $X_1, \dots, X_n$  die Nachfolger von  $X$ , aus denen MAX wählt.  
Seien  $X_{k1}, \dots, X_{km}$  die Nachfolger von  $X_k$ , aus denen MIN wählt.  
Seien  $X_{kj1}, \dots, X_{kjo}$  die Nachfolger von  $X_{kj}$ , aus denen MAX wählt.
- Wenn  $eval(X_{kj}) < \alpha$ , dann breche Suchzweig für  $X_k$  ab.  
Wenn  $eval(X_k) > \alpha$ , dann setze  $\alpha$  in  $X$  auf  $eval(X_k)$ .
- Wenn  $eval(X_{kji}) > \beta$ , dann breche Suchzweig für  $X_{kj}$  ab.  
Wenn  $eval(X_{kj}) < \beta$ , dann setze  $\beta$  in  $X_k$  auf  $eval(X_{kj})$ .

Weitere Literatur zu diesem Thema finden Sie in den Literaturreferenzen dieses Aufgabenblatts sowie im Internet.

### Aufgabe 3

Erweitern Sie Ihr Programm dahingehend, dass Eigenschaften wie Effizienz (Rechenzeit pro Zustand, Anzahl Zustände, Gesamtzeit pro Zug etc.) berechnet werden können. Erstellen Sie außerdem Diagramme für diese Eigenschaften, die die in Aufgabe 1 implementierte Version der *Paranoid*-Strategie mit Ihrer Implementation aus Aufgabe 2 vergleichen.

### Literatur

- [1] Russell, Norvig: Artificial intelligence. A modern approach. 1. Auflage, Kapitel 5
- [2] Lorenz, Tscheuschner: Player Modeling, Search Algorithms and Strategies in Multi Player Games. (Paper)