Informatik II (D-ITET)
Frühjahrssemester 2017
Prof. Friedemann Mattern
Hossein Shafagh (shafagh@inf.ethz.ch)

ETH Zürich
Institut für Pervasive Computing
Gruppe für Verteilte Systeme
http://vs.inf.ethz.ch

Übungsserie Nr. 9

Ausgabe: 3. Mai 2017
Abgabe: 9. Mai 2017

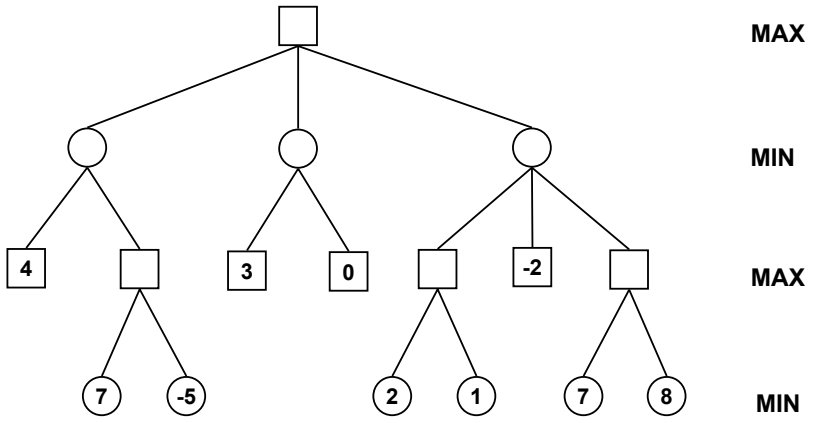
Hinweise

Für diese Serie benötigen Sie die folgenden Archive:
https://www.vs.inf.ethz.ch/edu/I2/downloads/u9.zip
https://www.vs.inf.ethz.ch/edu/I2/downloads/reversi.jar

1. Aufgabe: (8 Punkte) Spieltheorie

ETH Codeboard: https://codeboard.ethz.ch/ifee2u9a1

Die Aufgaben in diesem Teil beziehen sich alle auf den folgenden Spielbaum:



(1a) *(1 Punkt)* Wie gross ist die Suchtiefe dieses Baums? Wie gross ist seine Höhe? (Ein Baum mit nur einem Knoten habe die Höhe 1.)

(1b) *(2 Punkte)* Wenden Sie die Minimax-Regel auf den Spielbaum an und beschriften Sie dabei alle Knoten mit den entsprechenden abgeleiteten Werten. Finden Sie dadurch den besten Zug in der aktuellen Position für den Spieler MAX.

(1c) *(2 Punkte)* Geben Sie für den Spielbaum eine optimale *Strategie* für den Spieler MAX an.

(1d) *(3 Punkte)* Berechnen Sie den Wert der Wurzel des Spielbaums mit Hilfe der α-β-Methode. Versehen Sie dabei jeden Ast mit den aktuellen Werten für α und β. Markieren Sie ausserdem die Stellen, an denen Sie einen entsprechenden Schnitt gemacht haben.

2. Aufgabe: (10 Punkte) Reversi [Teil 3]

ETH Codeboard (Abgabe als Zip Datei): `https://codeboard.ethz.ch/ifee2u9a2`

(2a) *(5 Punkte)* Implementieren Sie einen Reversi-Spieler, der eine Minimax-Analyse der Spielsituation durchführt. Machen Sie dabei die maximale Suchtiefe konfigurierbar und verwenden Sie als Bewertungsfunktion (wie in der letzten Reversi-Aufgabe) die Differenz der Spielsteine.

(2b) *(5 Punkte)* Im Turnier hat jeder Spieler nur eine begrenzte Zeit zur Verfügung. Wie lange diese ist, wird dem Spieler durch die Methode *initialize* mitgeteilt.

Passen Sie Ihren Reversi-Spieler so an, dass er die Minimax-Analyse mit ständig wachsender Suchtiefe so lange immer wieder durchführt, bis ihm die Zeit ausgeht. Am Ende gibt er das Ergebnis der letzten vollständigen Analyse zurück. Eine mögliche Variante der Implementierung ist es, zu Beginn der *min-* bzw. *max*-Methode zu testen, ob noch genügend Zeit vorhanden ist. Falls nicht, kann eine *Exception* geworfen werden, die dann in der Methode *nextMove* abgefangen wird.

Hinweis: Die Methode *System.currentTimeMillis* liefert die aktuelle Systemzeit in Millisekunden seit dem 1. Januar 1970, 00:00:00 (GMT).

(2c) *(fakultativ)* Implementieren Sie eine bessere Bewertungsfunktion. Orientieren Sie sich dabei an der auf der Reversi-Seite verlinkten Literatur.

Hinweis: Der *LogPlayer* kann ein vorheriges Spiel nachahmen, in dem er die Züge aus einem Logfile liest. Das kann hilfreich sein, um Spielsituationen wiederherzustellen, in denen Ihr Spieler sich falsch verhalten hat.