Universitatea
POLITEHNICA
București

Facultatea de
Automatică și
Calculatoare

Departamentul
de Calculatoare

Metoda „Monte Carlo Tree Search” în Chinese Checkers

Proiect de diplomă - 2015

Autor

Macri Matei

Conducător științific

Prof. Dr. Ing. Adina Magda Florea

Introducere

- Metoda Monte Carlo Tree Search (MCTS) a avut performanțe deosebite în jocul de GO
- Este o metodă versatilă și potrivită pentru un număr mai mare de jucători
- Jocul Chinese Checkers este o platformă corespunzătoare pentru testarea metodei

Jocul Chinese Checkers

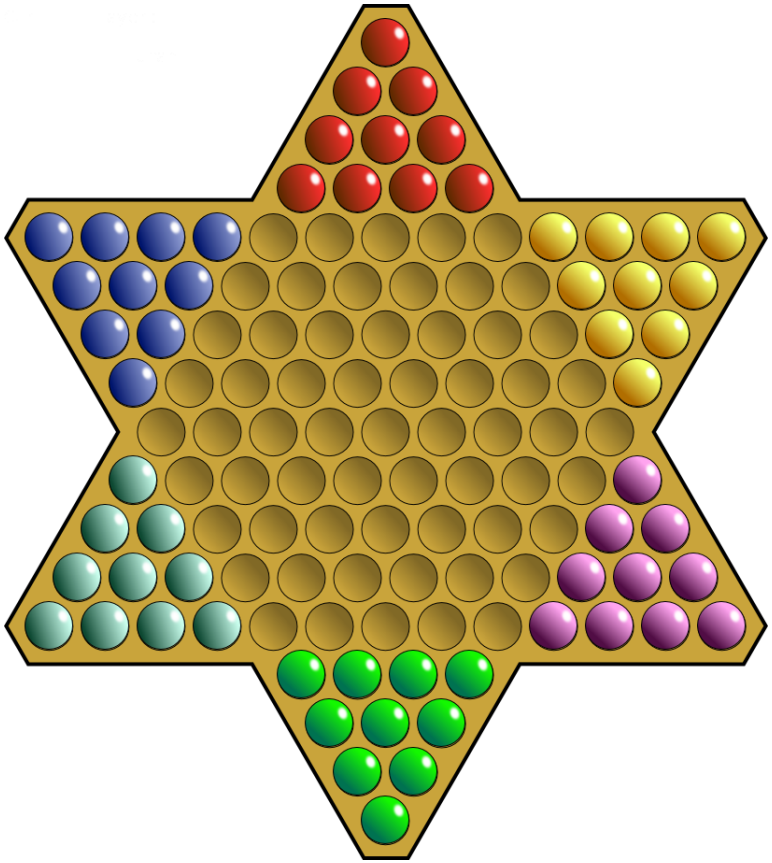
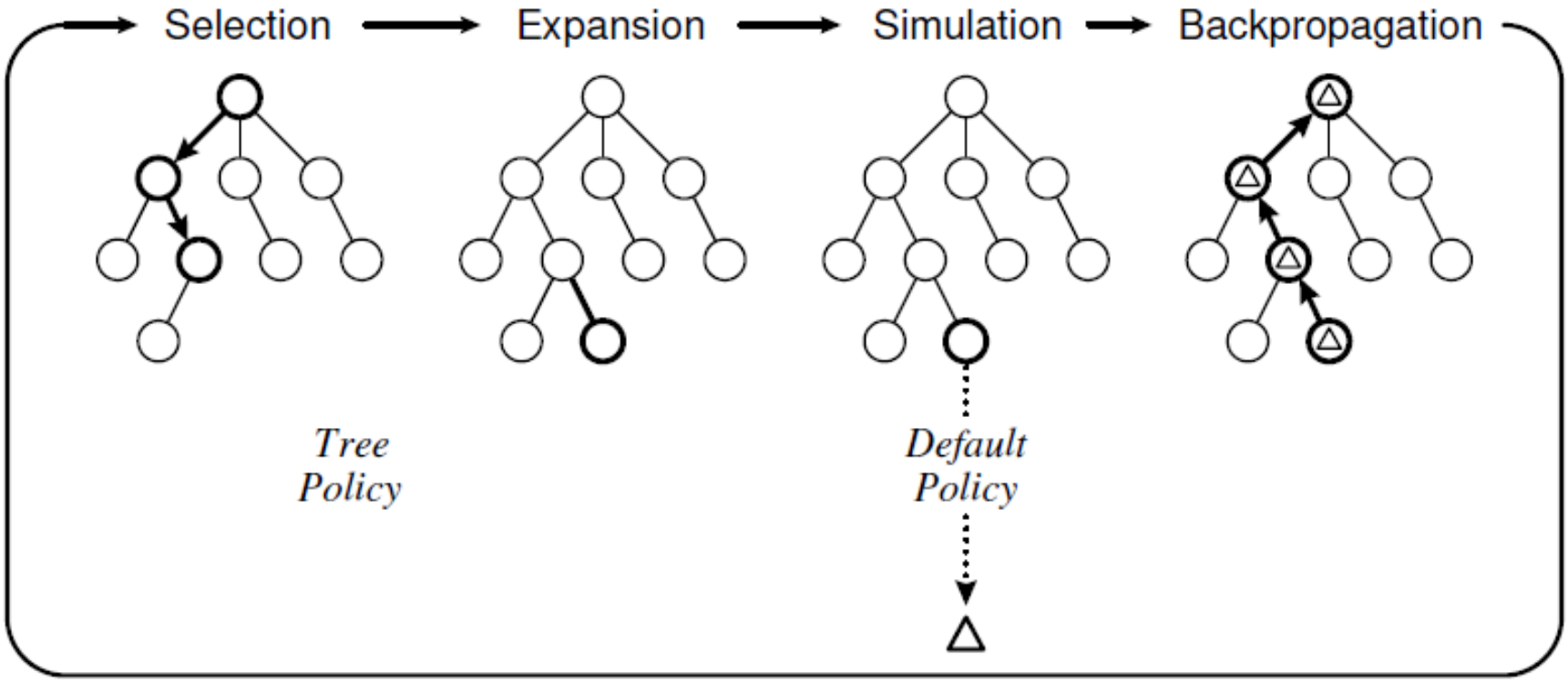


Tabla de joc Chinese Checkers

- 2-6 jucători
- Scopul jocului este să-ți duci piesele în colțul opus
- Se poate sări peste piese
- Se pot înlănțui oricâte salturi

Algoritmul MCTS



Un ciclu al algoritmului Monte Carlo Tree Search

Algoritmul MCTS (2)

- Selecția nodului celui mai promițător se realizează conform unei politici, cum ar fi:

$$v_i = \frac{s_i}{n_i} + C \cdot \sqrt{\frac{\ln(n_p)}{n_i}}$$

- Simularea (playout) folosește de asemenea o politică, care stabilește mutările de executat în simulare
- Modificarea etapelor de selecție și simulare (eventual și backpropagation) diferențiază implementările de MCTS

Implementări existente

- Simulări mai complexe, 2-ply (max^n, paranoid, best reply search)
- Simulări mai rapide, evaluare după număr fix de ture (euristică, lookup table)
- Alte optimizări: progressive history, ε-greedy playouts, k-best pruning

Soluții implementate

- Politici default, dar cu ε-greedy playouts și k-best pruning
- Evaluare după număr fix de ture
- Simulare 2-ply
- MCTS cu scor, în loc de victorii/înfrângeri
- Euristica de continuitate
- Istorie pe termen lung
- Istorie progresivă

Rezultat experiment 1 vs 1

În jocurile 1 vs 1 pentru determinarea implementării celei mai bune, am constatat:

- MCTS clasic, cu euristica de continuitate se dovedește mai bun decât variantele cu evaluare și scor
- Istoria progresivă aduce un mic avantaj în selectarea celei mai bune mutări
- Istoria pe termen lung ajută doar după un număr considerabil de jocuri

Rezultat experiment 1 vs 1 (2)

1ul jucător – 2lea jucător	5 secunde	15 secunde
Clasic - Evaluare	10-0	9-1
Evaluare - Clasic	0-10	3-7
Clasic - Scor	9-1	9-1
Scor - Clasic	0-10	2-8
Clasic - 2ply	9-1	7-3
2ply - Clasic	0-10	1-9
Clasic – Istorie Progresivă	7-3	7-3
Istorie Progresivă - Clasic	9-1	9-1
Clasic – Istoric acțiuni	6-4	7-3
Istoric acțiuni - Clasic	10-0	9-1

Rezultat experiment 3-6 jucători

Jucători	Scor în 10 jocuri
Clasic-Clasic-Clasic	5-3-2
Clasic-Clasic-Evaluare	0-7-3
Clasic-Clasic-Istorie Progresivă	2-3-5
Clasic-Clasic-Clasic-Clasic	3-4-0-3
IP – Clasic – IP – Clasic	1-3-2-4
Clasic-Clasic-Clasic-Clasic-Clasic	1-1-1-6-1
Clasic-Clasic-Clasic-Clasic-IP	3-3-1-2-1
Clasic-Clasic-Clasic-Clasic-Clasic-Clasic	0-2-2-3-1-2
IP – Clasic – IP – Clasic – IP – Clasic	3-2-2-2-1-0

Concluzii

- Politica default, plus câteva optimizări, este varianta cea mai bună pentru Chinese Checkers
- Folosind această implementare de MCTS, AI-ul poate fi învins doar dacă este al doilea jucător, acesta neavând avantajul primei mutări
- În forma implementată, istoria progresivă cât și cea pe termen lung nu aduc îmbunătățiri semnificative