Creating an Upper-Confidence-Tree program for Havannah

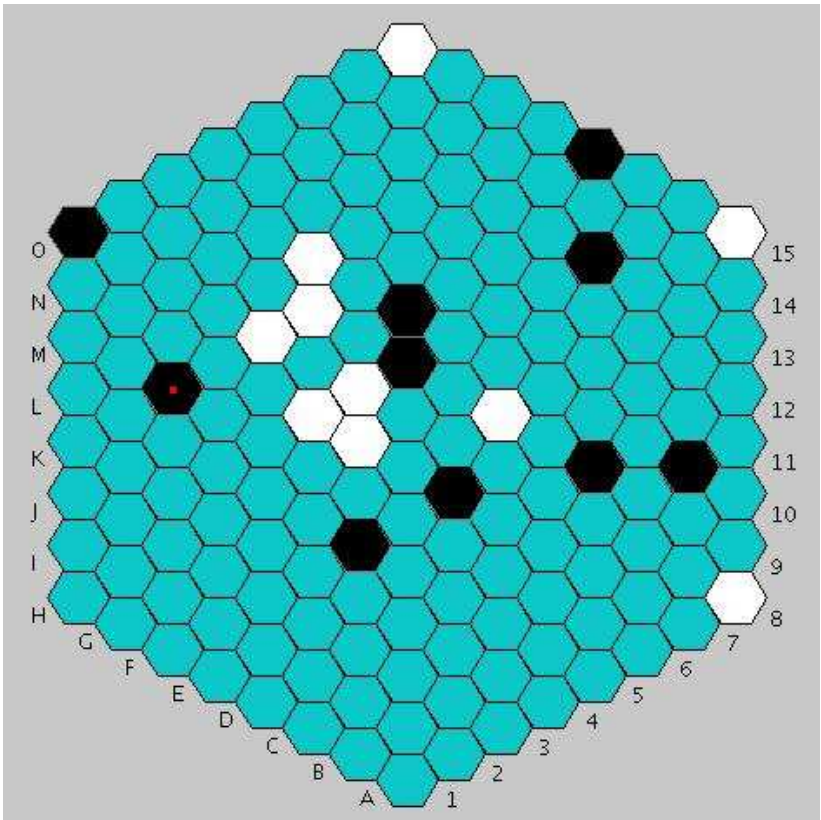
Fabien Teytaud, Olivier Teytaud

Tao, Inria Saclay Ile-De-France, LRI (Université Paris Sud, France),
UMR CNRS 8623, I&A team, Digiteo, Pascal Network of Excellence

May 6, 2009

The game of Havannah

- Christian Freeling
- 2-players board game
- hexagonal board of hexagonal locations

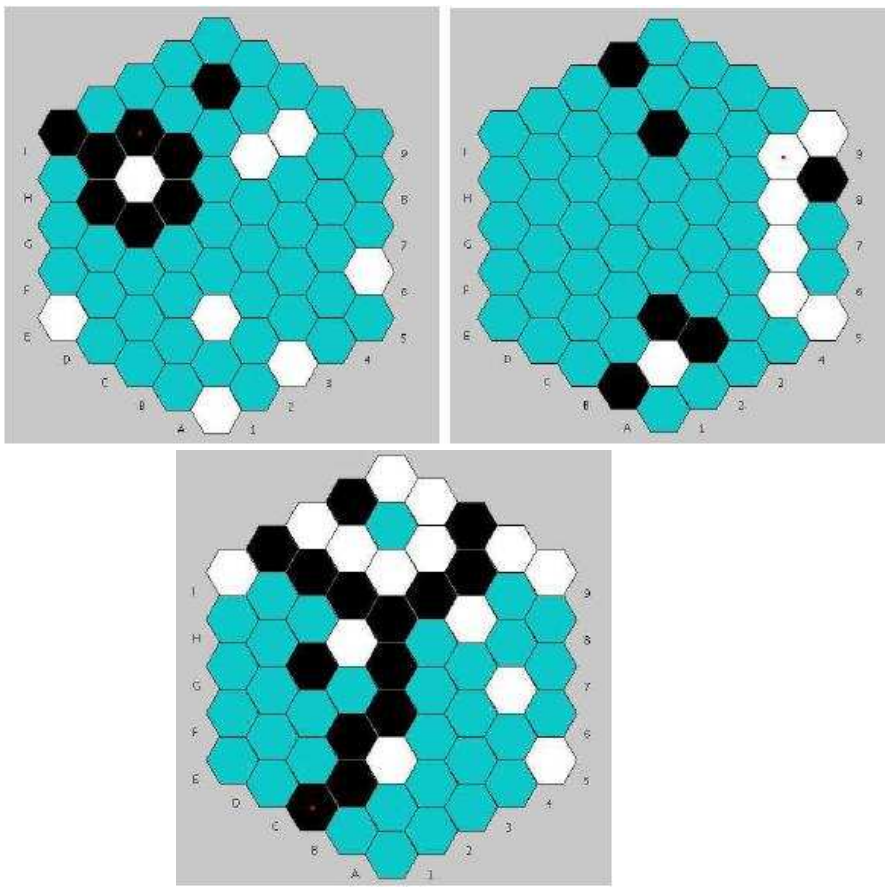


The game of Havannah

To win, a player has to realise :

- A ring
- A bridge
- A fork

$\Longrightarrow$ Connection game
(related to Hex)



Goal of the paper

Havannah is difficult for computers :

- few local patterns are known
- no natural evaluation function
- no pruning rule for reducing the number of reasonable moves
- large action space (271 for the first move on a board of size 10)

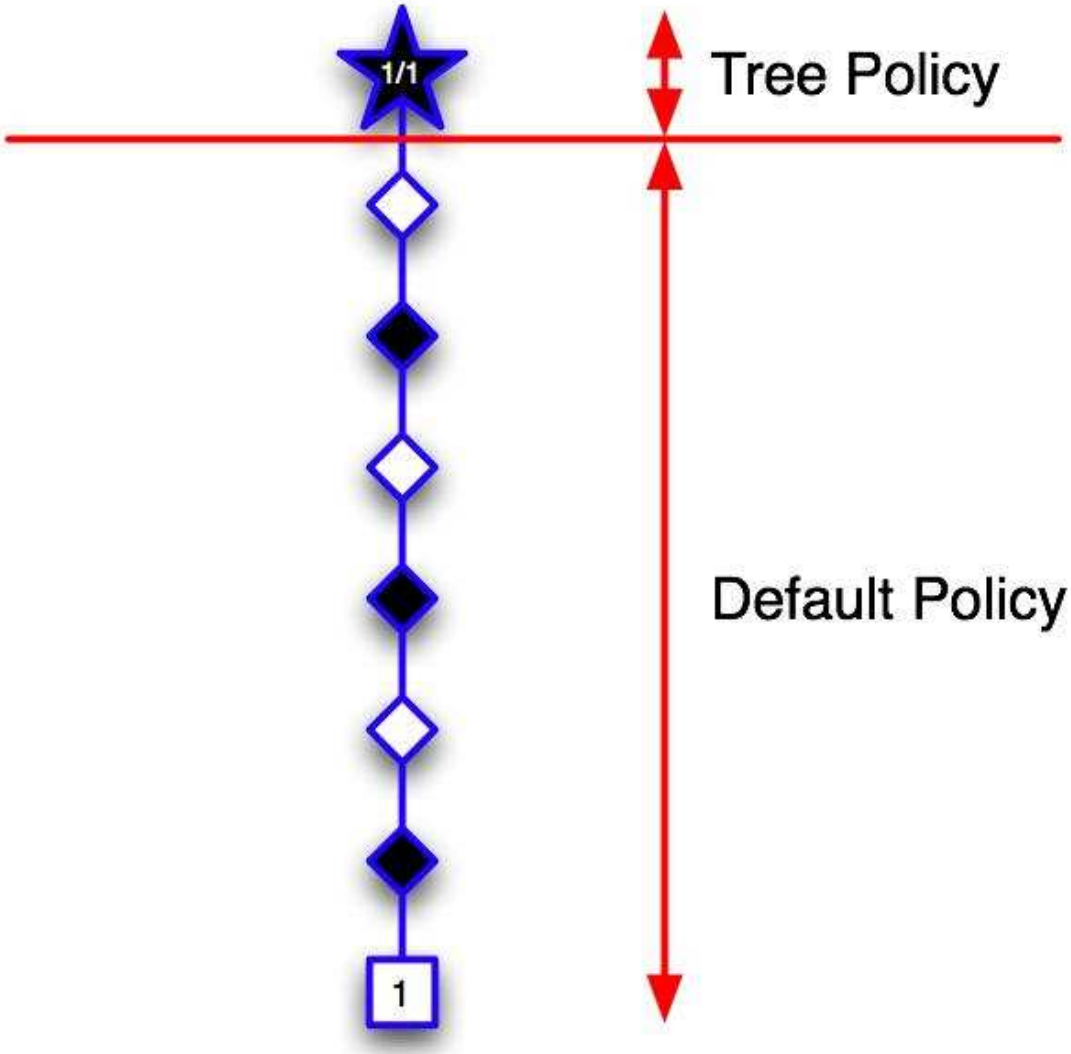
$\Rightarrow$ Investigate MCTS/UCT approaches by testing it in Havannah.

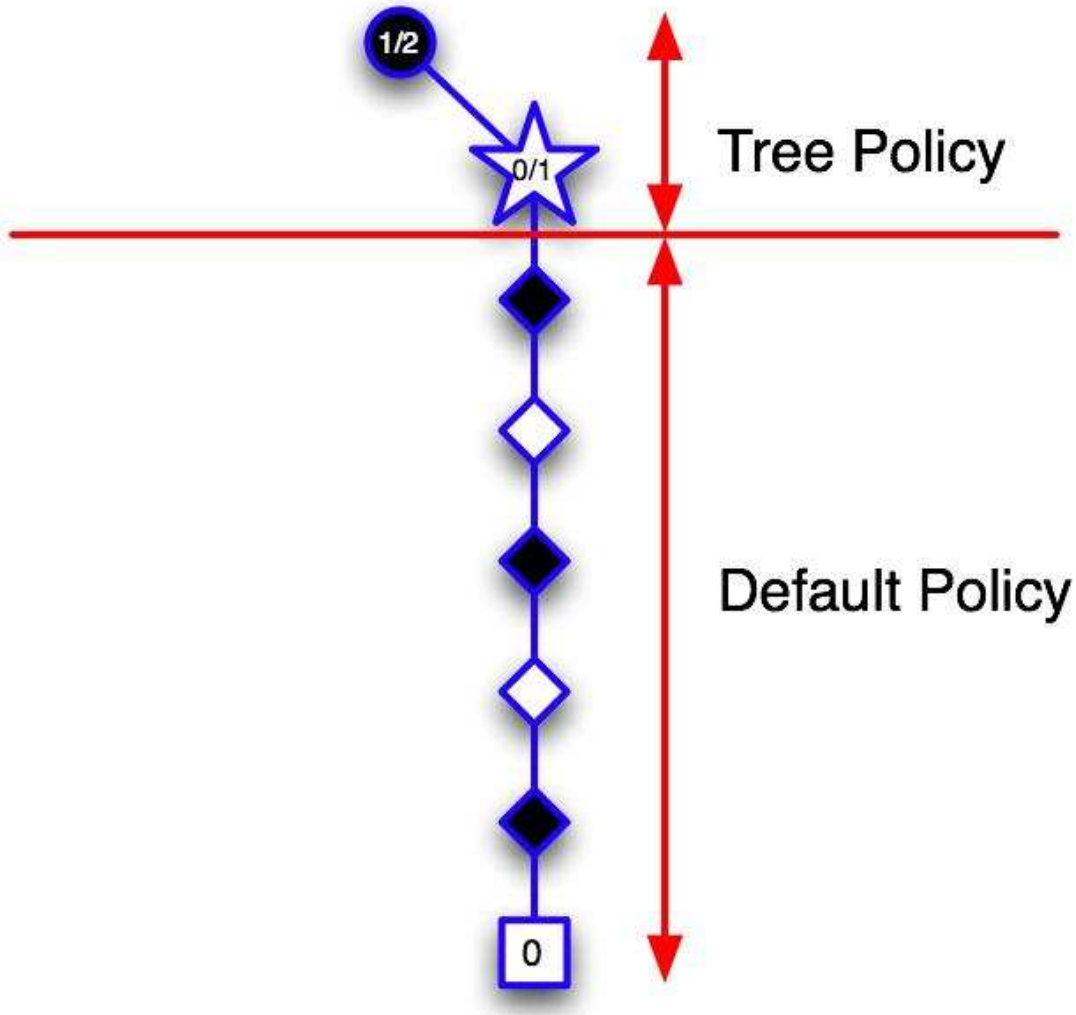
- Coulom, Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search, 2006.
- Chaslot et al, Monte-carlo strategies for computer go, 2006.
- Kocsis et al, Bandit-based Monte-Carlo planning, 2006.

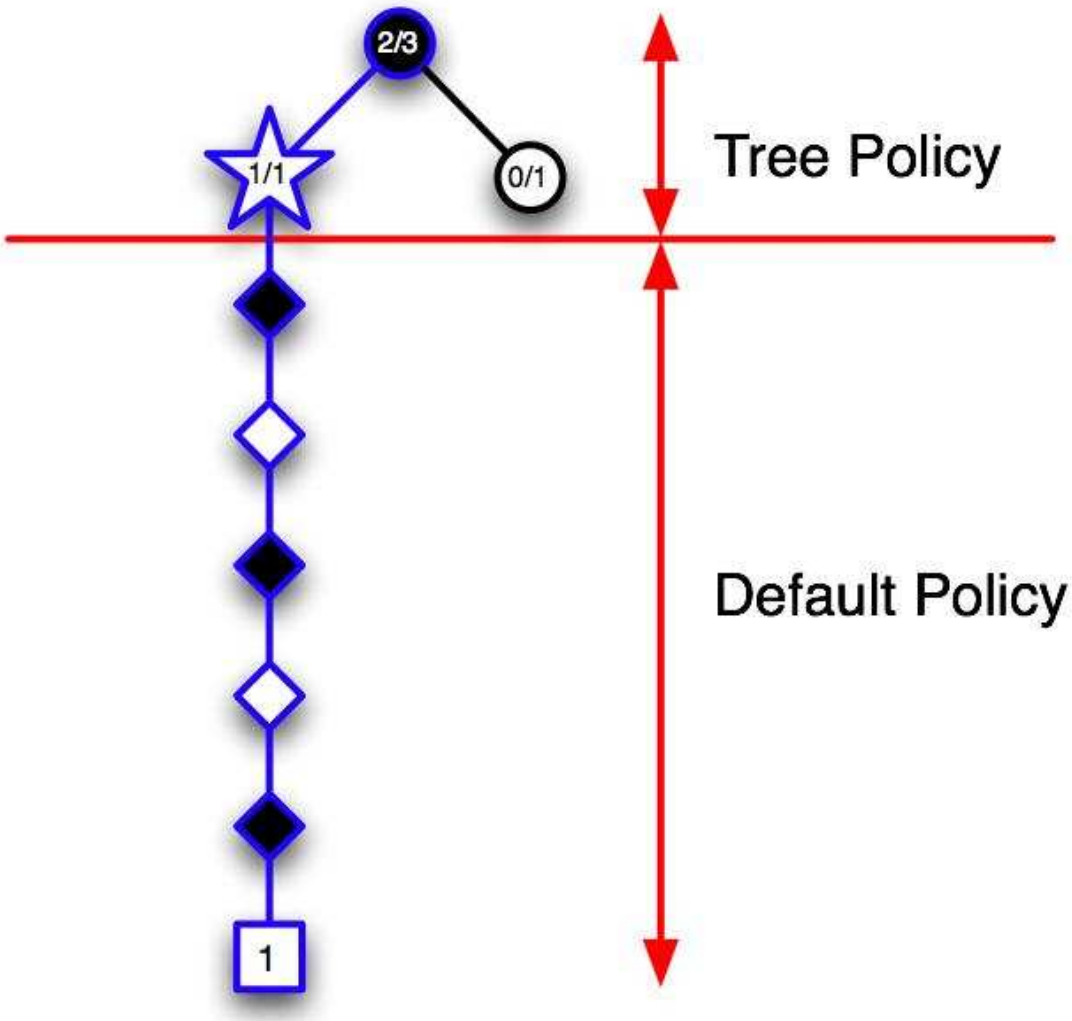
Monte-Carlo Tree Search

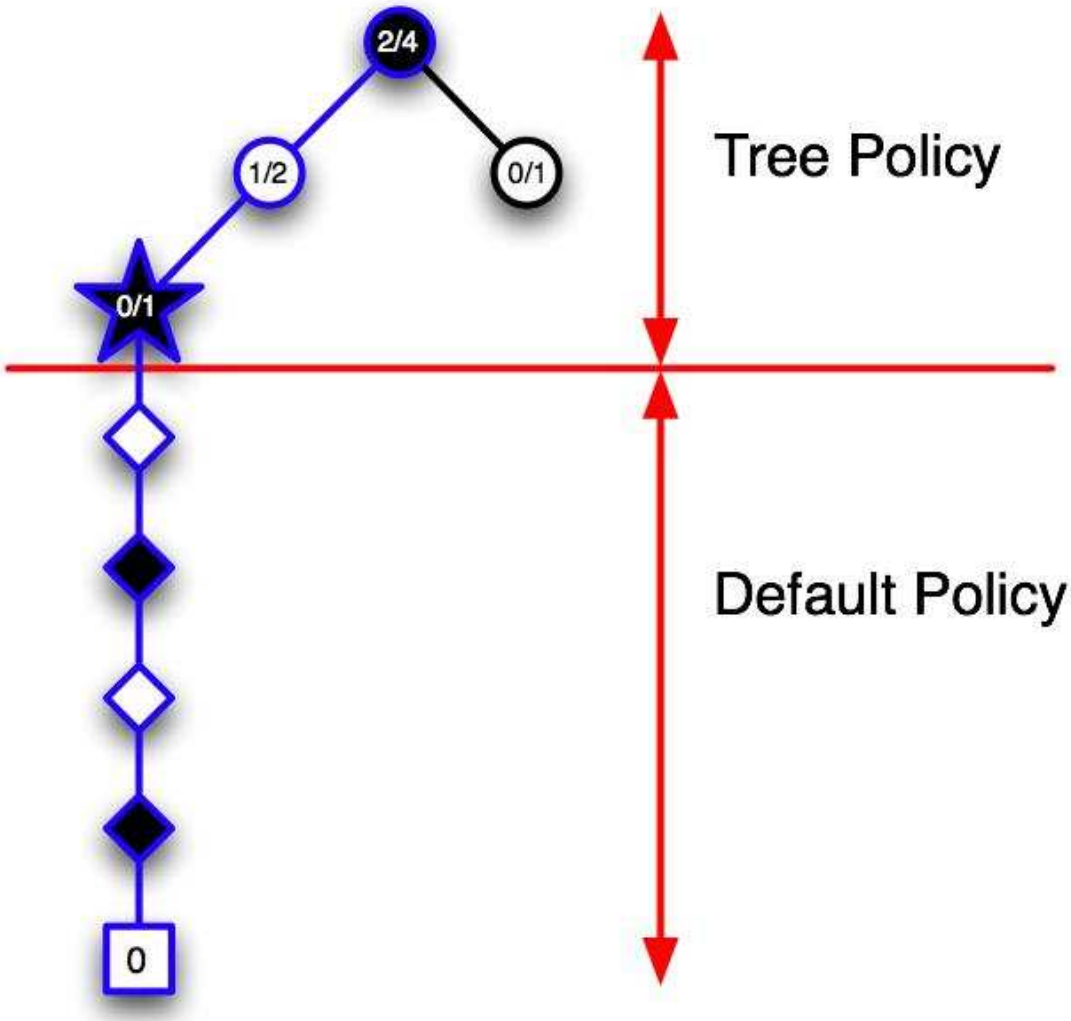
- Construction of an unbalanced Tree of possible futures.
- Evaluation of each node with random simulations.
- Trade off between exploration and exploitation.

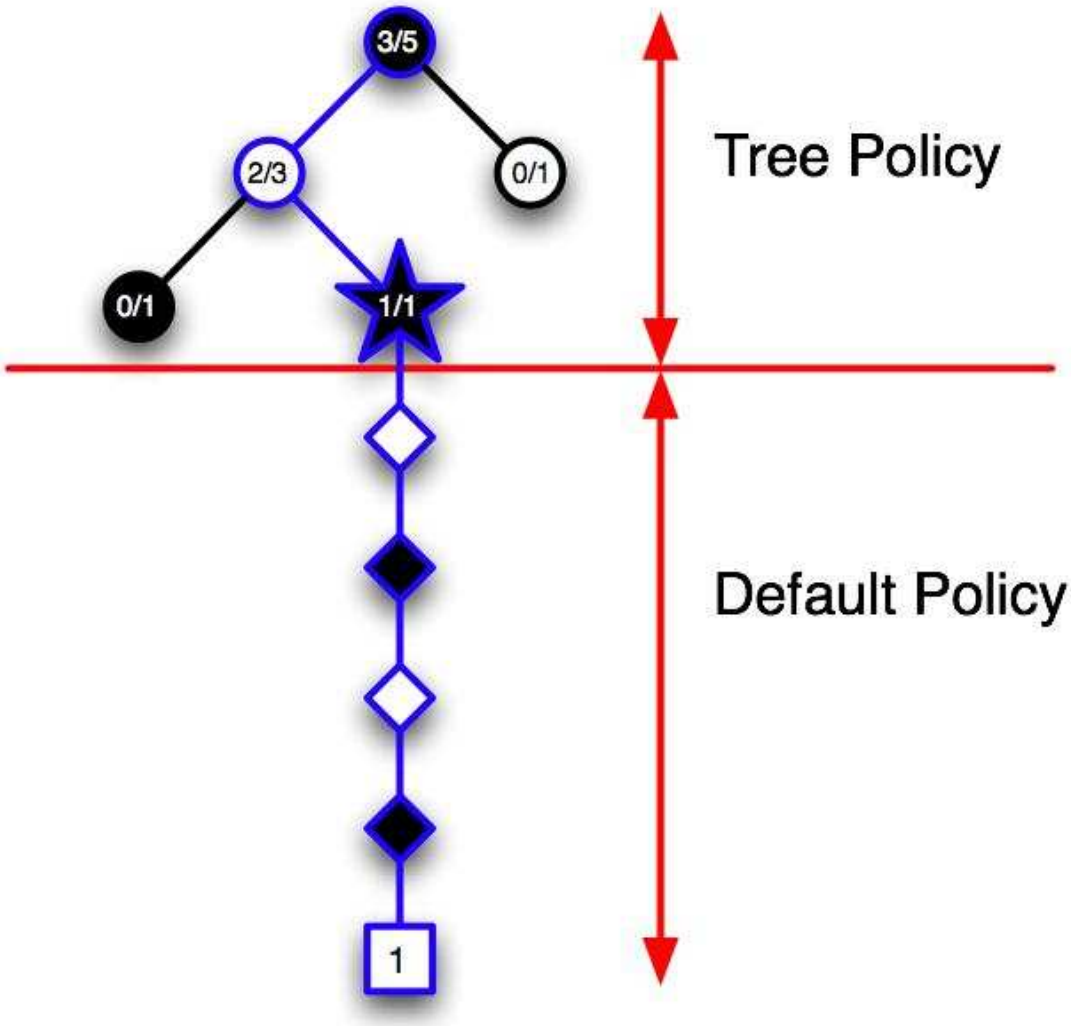
$\Longrightarrow$ very different from alpha-beta
$\Longrightarrow$ much better at least for the game of Go


1/1
Tree Policy
Default Policy
1


1/2
0/1
Tree Policy
Default Policy
0


2/3
1/1
0/1
1
Tree Policy
Default Policy

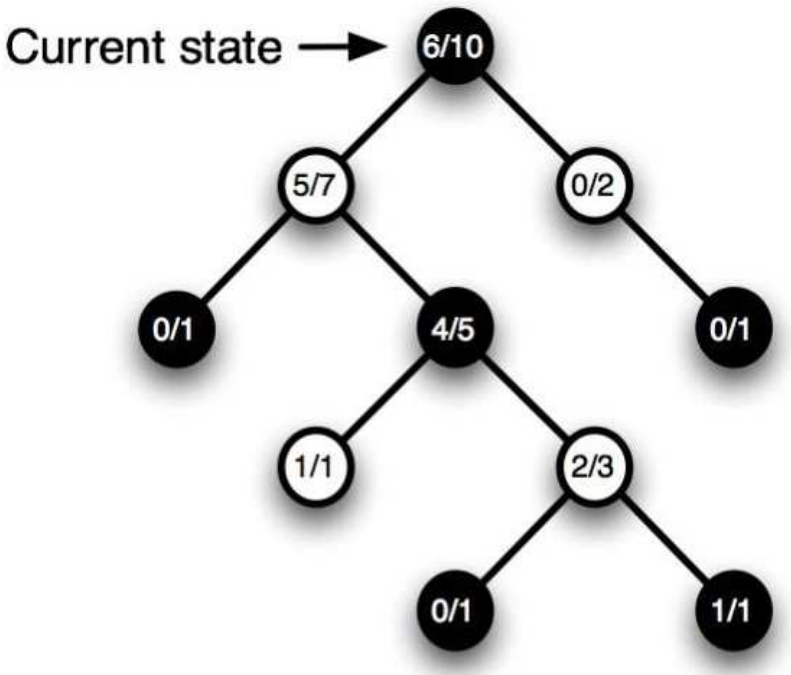

2/4
1/2
0/1
0/1
0
Tree Policy
Default Policy


3/5
2/3
0/1
0/1
1/1
1
Tree Policy
Default Policy

Policies

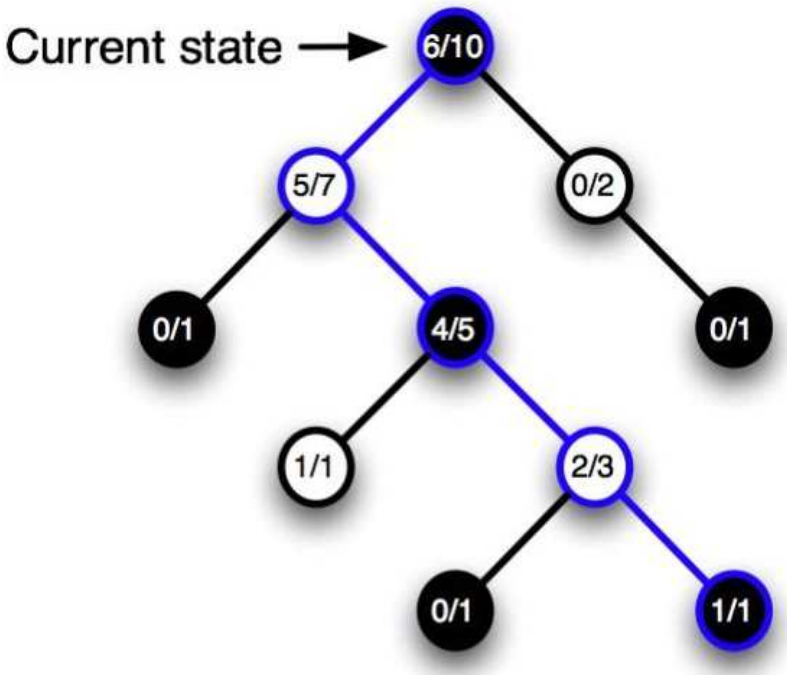
- Tree Policy:
  We choose the move i with the highest
  $$\hat{X}_i + p\sqrt{\frac{logT}{T_i}}$$
  $\hat{X}_i$: Empirical average reward for move i
  T_i: Number of trials for move i
  T: Total number of trials
- Default Policy:
  Random
  Some handcrafted simple rules

Trade-off between...



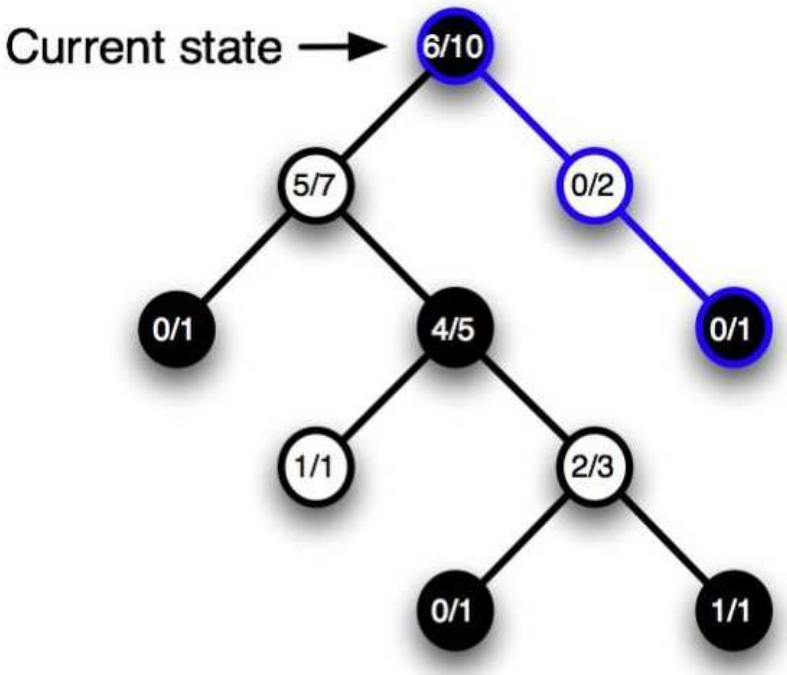
$$\hat{X}_i + p\sqrt{\frac{logT}{T_i}}$$

Exploitation



$$\hat{X}_i + p\sqrt{\frac{logT}{T_i}}$$

Exploration



$$\hat{X}_i + p\sqrt{\frac{logT}{T_i}}$$

Scalling of UCT

How does the program scale with computational power ?

Number of simulations of both player	Success rate
250 vs 125	0.75 ± 0.02
500 vs 250	0.84 ± 0.02
1000 vs 500	0.72 ± 0.03
2000 vs 1000	0.75 ± 0.02
4000 vs 2000	0.74 ± 0.05

$\Rightarrow$ doubling the computational power provides $\simeq$ 74% success rate

($\simeq$ 63% in the case of Go)

Variance-based confidence bounds

For a move d with empirical variance σ^2

$$\sqrt{\sigma^2 2\log(3/\delta)/n} + 3\log(3/\delta)/n \qquad (\textit{Bernstein})$$

instead of

$$\sqrt{\log(2/\delta)/n}. \qquad (\textit{Hoeffding})$$

$\Rightarrow$ better when small variance

Audibert et al. Use of variance estimation in the multi-armed bandit problem.

Results

K	Score against standard Formula
0.250	0.503 ± 0.011
0.100	0.578 ± 0.010
0.030	0.646 ± 0.005
0.010	0.652 ± 0.006
0.001	0.582 ± 0.012
0.000	0.439 ± 0.015

$\Rightarrow$ moderate but significant improvement

RAVE

For a move d in situation s simulated n times,

$$newScore = (1-\alpha)score + \alpha\, rave + exploration$$

where

- *score* is the ratio of won sims with move d *in* s.
- *rave* is the ratio of won sims with move d *after* s,
- $\alpha = R/(R+n), \quad exploration = \sqrt{K \log(2+m)/n}$.

(m = total number of sims in situation s)
$\Rightarrow$ exploration constant K, weight of Rave values R.

S. Gelly et al. Combining online and offline knowledge in UCT. 2007.

Rave

size 5	R=50, K=0.25, size 5, 1000 sims/move	47.26% $\pm$ 4.0 %
	R=50, K=0.05, size 5, 1000 sims/move	60.46% $\pm$ 2.9 %
	R=50, K=0, size 5, 1000 sims/move	95.33% $\pm$ 0.01 %
size 8	R=50, K=0, size 8, 1000 sims/move	100% on 1347 runs

$\Rightarrow$ Good results, in particular with $K = 0$!

Conclusions: the generality of MCTS

- Good overall performance for UCT, without expertise
- Rapid action value estimates:
  - Especially efficient with big size
  - But less efficient for big nb of simulations
- Bernstein is validated (stable improvement)
- No good results with heuristic-free[1] progressive widening[2]

[1] Progressive widening is proved in the heuristic-free case in Wang and al, Algorithms for infinitely many-armed bandits, 2008.

[2] R. Coulom, Computing Elo Ratings of Move Patterns in the Game of Go.