Multiple Tree for Partially Observable Monte-Carlo Tree Search

D. Auger

Tao, LRI, Université Paris-Sud, Inria Saclay-IDF

We adapt the EXP3 Algorithm, an efficient algorithm solving the adversarial bandit problem, to the case of tree-structured partially observable games. Every player will select his/her strategy along repeated games with a Monte-Carlo Tree Search algorithm, receiving observations from other players via a referee. We give experimental results for the game of Phantom Tic-Tac-Toe.

Multi-Armed Bandit Problem

K one-armed bandit slot machines.

- At each new timestep the player :
- pulls an arm i_t according to his strategy, which can depend on past observation and be randomized
- observes the reward $r_{i_t}(t)$ of the chosen arm i_t
- **Stochastic Setting**: rewards are given by stationnary unknown probability distributions r_i
- **Adversarial Setting**: an opponent, aware of the player's past decisions and rewards, chooses simultaneously with the player a (possibly randomized) reward $r_i(t)$ for each arm.

What can you do in the adversarial case ?

- impossible to "maximize" reward
- You can try to minimize the **external regret**: difference at time T between one's gain and the gain which could have been obtained by allways pulling *the same* arm

$$R_T = \max_{i=1\dots k}\left(\sum_{t=1}^{T} r_i(t)\right) - \sum_{t=1}^{T} r_{i_t}(t)$$

External regret can be minimized (in expectation or with high probability) by the EXP3 Algorithm [ACBFS03]

EXP3 Algorithm

Parameter : real $\gamma \in]0;1]$

Initialization : define the weight $w_i(t) = 1$ for $t = 1$ and all $i = 1, \dots k$

For each $t = 1, 2, \dots$

1. Set
$$p_i(t) = (1-\gamma)\frac{w_i(t)}{\sum_{j=1}^{k} w_j(t)} + \frac{\gamma}{K}$$
for $i = 1, \dots K$.
2. Select randomly an arm i_t according the the probabilities $p_1(t), \dots, p_K(t)$
3. Observe the reward $r_{i_t}(t)$
4. Update the weight of i_t by
$$w_{i_t}(t+1) = w_{i_t}(t)\exp\left(\frac{\gamma}{K}\frac{r_{i_t}(t)}{p_{i_t}(t)}\right)$$
and set $w_j(t+1) = w_j(t)$ for other arms.

- In order to find good actions EXP3 maintains a balance between
- **exploration**: weighting actions according to the reward: $1-\gamma$ term in the probability, and
- **exploration**: the uniform term $\frac{\gamma}{K}$ ensures that unsufficiently tested actions will be regularly selected.

Theorem [Auer *et al* [ACBFS03]] *When run with parameter*

$$\gamma = \min 0.8\sqrt{\frac{\ln K}{TK}}, \frac{1}{K}$$

the expected regret satisfies

$$\frac{R_T}{T} \leq 2.7\sqrt{\frac{K\ln K}{T}}$$

External regret and Zero-Sum Matrix Games

- Two players simultaneously choose a column i and a line j of a given matrix M
- The line player receives $M_{i,j}$ and the column player receives $-M_{i,j}$.

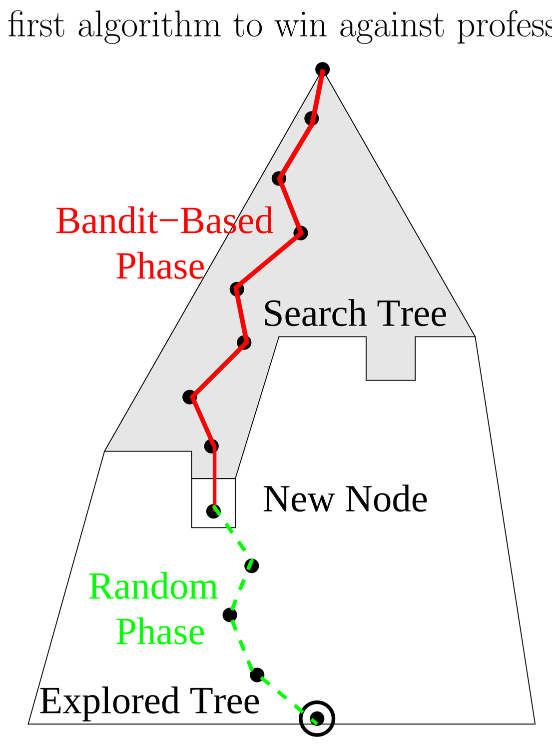
Exemple : the Rock-Paper-Scissors game

$$\begin{pmatrix} & \text{Rock} & \text{Cissors} & \text{Paper} \\ \text{Rock} & 0 & +1 & -1 \\ \text{Cissors} & -1 & 0 & +1 \\ \text{Paper} & +1 & 0 & -1 \end{pmatrix}$$

If both players repeatedly play, selecting their strategies with an algorithm minimizing external regret, then the empirical frequencies of play converge to optimal strategies ("Nash Equilibrium").

Monte-Carlo Tree Search Algorithms

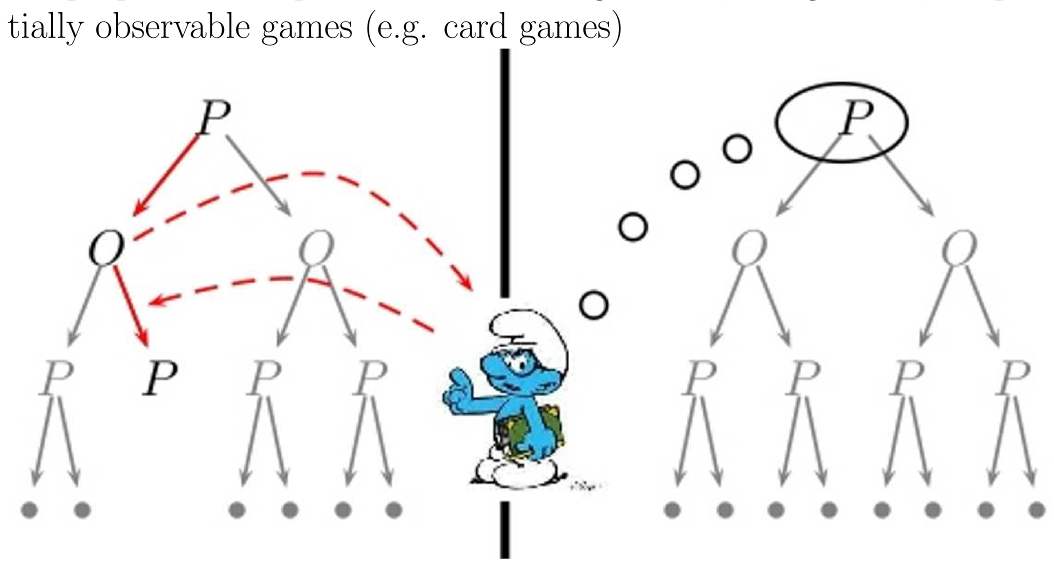
- MCTS Algorithms are efficient algorithms designed for tree-searching with huge inputs.
- They have been used to design computer players in games with full observation, e.g. Go
- Mogo was the first algorithm to win against professional Go players



- As in the multi-armed bandit problem we must balance *exploitation* and *exploration*
- classical implementations [KS06] use the UCB bandit algorithm [LR85], designed for the stochastic setting.

Multiple Tree Monte-Carlo Tree Search

We propose an adaptation of MCTS algorithms, using EXP3, for partially observable games (e.g. card games)



Each player runs a separate MCTS algorithm and sees other players' moves via observations sent by the referee

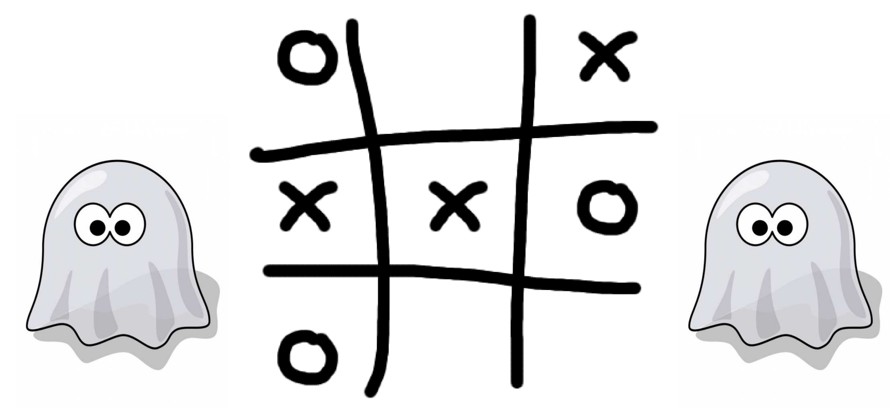
- in a Player Node "P", the player chooses his next action with the EXP3 algorithm
- in an Observation Node "O" the player waits for the referee to send an observation

Properties: consistant, efficient, online

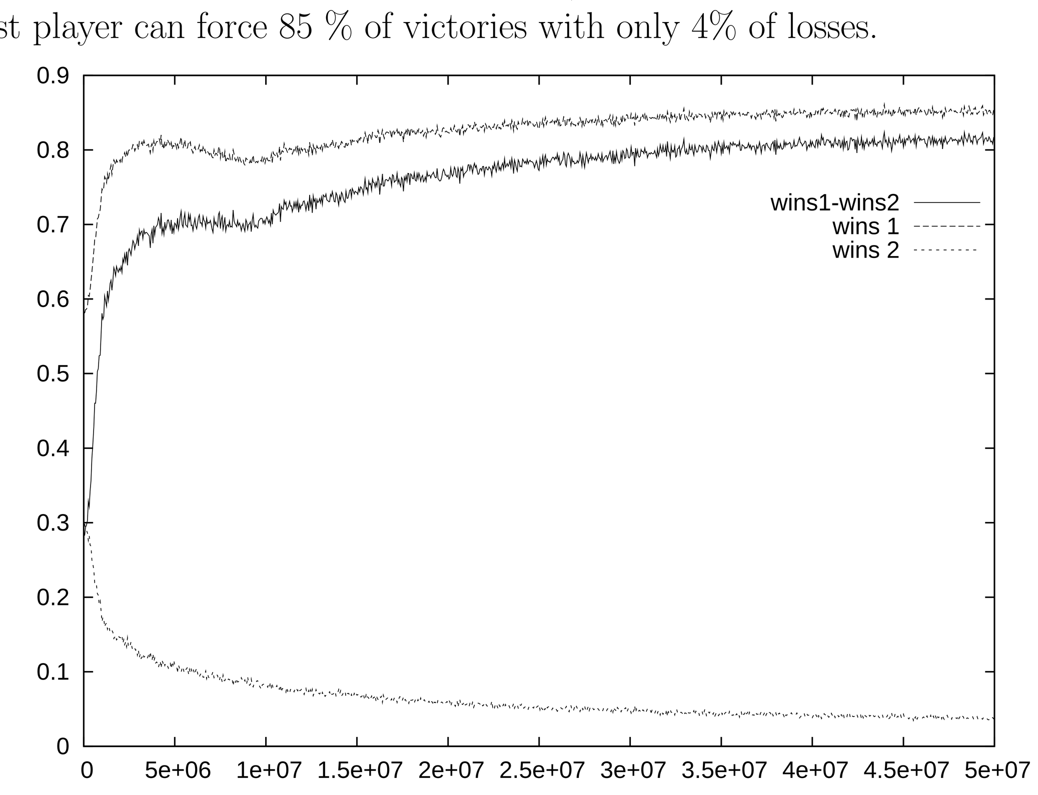
Main advantage: the tree is only partially explored, as opposed to [ZJBP08]

Application to the game of Phantom Tic-Tac-Toe

- Played like standard Tic-Tac-Toe but one does not see where the opponent plays.
- In case of an illegal move the player must play somewhere else



Whereas standard Tic-Tac-Toe is a draw, in Phantom Tic-Tac-Toe the first player can force 85 % of victories with only 4% of losses.



simulations

A Phantom Tic-Tac-Toe personal Olympiad

Opponents are :

- MMCTS 500K, 5M and 50M, mixed strategies obtained by our algorithm after 500K, 5M or 50M iterations
- Belief Sampler, who plays standard tic-tac-toe perfectly and randomizes his moves according to optimal strategies compatible with observations (standard approach for P.O. games, see e.g [Caz06])
- Random Player, a dummy uniform random player.

P1 \ P2	MMCTS 500K	MMCTS 5M	MMCTS 50M	Random	B.Sampler
M.500K	65% \ 25%	51% \ 37%	44% \ 47%	67% \ 22%	40% \ 43%
M. 5M	88% \ 06%	82% \ 10%	78% \ 17%	88% \ 05%	78% \ 10%
M. 50M	93% \ 02%	89% \ 03%	85% \ 04%	93% \ 02%	82% \ 03%
Random	55% \ 33%	48% \ 39%	41% \ 47%	59% \ 28%	30% \ 53%
B.Sampler	77% \ 14%	73% \ 18%	68% \ 22%	79% \ 12%	56% \ 28%

References

[ACBFS03] P. Auer, N. Cesa-Bianchi, Y. Freund, and R.E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2003.

[Caz06] T. Cazenave. A Phantom-Go program. *Advances in Computer Games*, pages 120–125, 2006.

[KS06] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. *Machine Learning: ECML 2006*, pages 282–293, 2006.

[LR85] T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

[ZJBP08] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione. Regret minimization in games with incomplete information. *Advances in Neural Information Processing Systems*, 20:1729–1736, 2008.