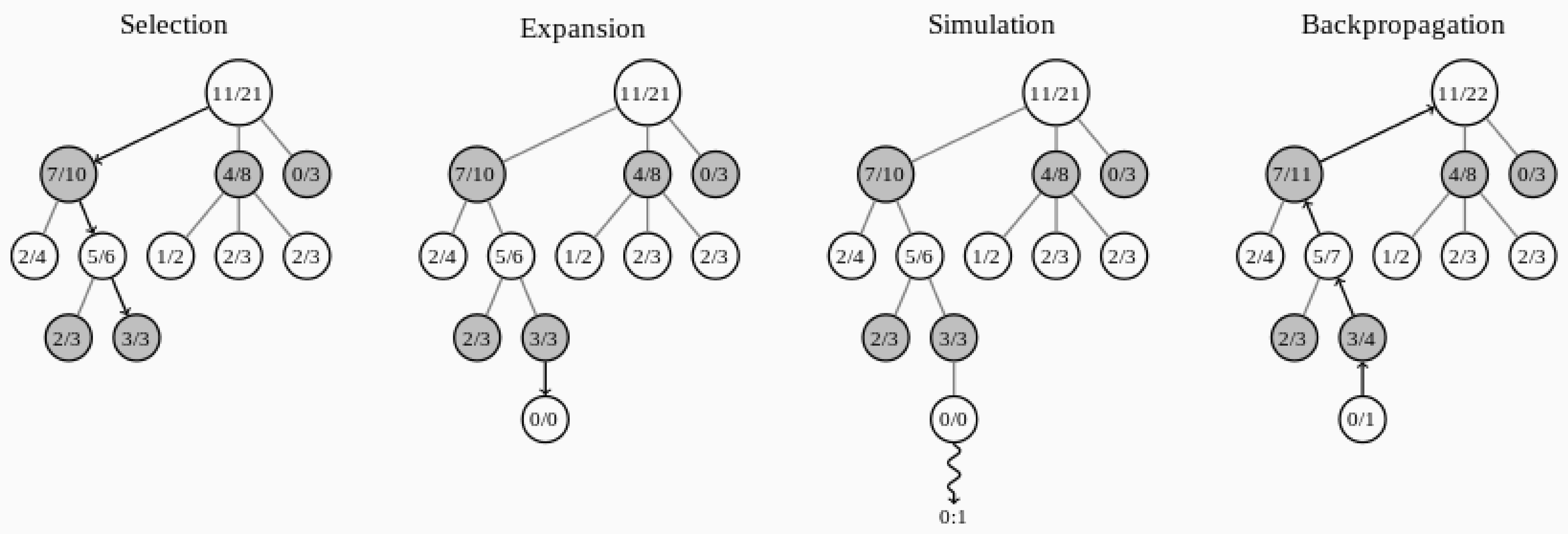
Monte-Carlo Tree Search by Best Arm Identification

Emilie Kaufmann Wouter M. Koolen

Centrum Wiskunde & Informatica

Monte Carlo Tree Search Successful . . .



source: Wikipedia

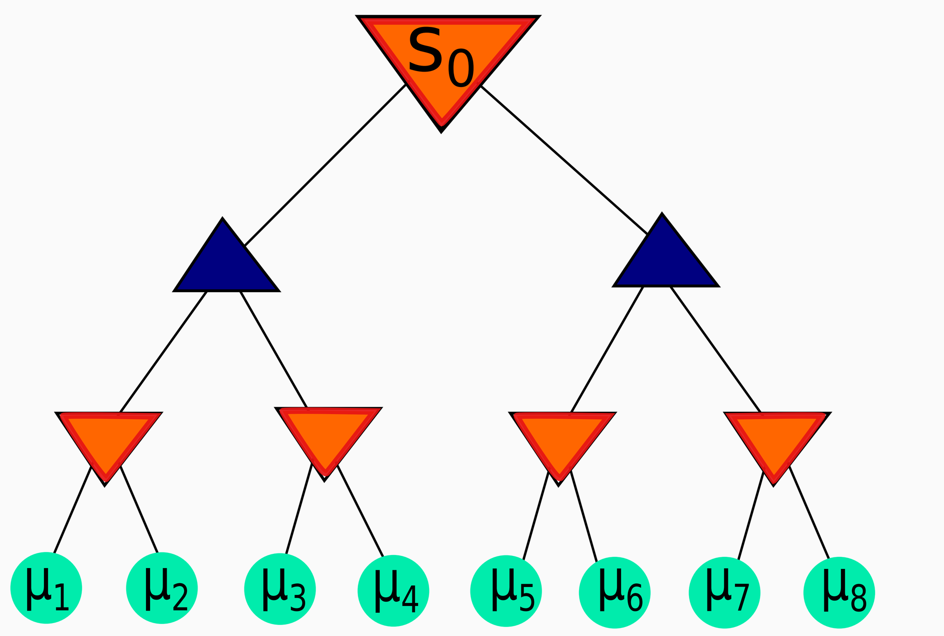
. . . But Theory Missing!

We introduce an idealized model:

- **fixed** maximin tree
- **i.i.d.** playouts starting from each leaf

and propose **new algorithms** with **sample complexity guarantees** based on **Best Arm Identification** methods.

Stylised Model



A fixed MAXMIN game tree $\mathcal{T}$, with leaves $\mathcal{L}$.

- MAX node (= your move)
- MIN node (= adversary move)
- Leaf ℓ: stochastic oracle $\mathcal{O}_\ell$ that evaluates the position

Learning Protocol

At round t a **MCTS algorithm**:

- picks a path down to a leaf L_t
- get an evaluation of this leaf $X_t \sim \mathcal{O}_{L_t}$

Assumption: i.i.d. sucessive evaluations, $\mathbb{E}_{X\sim\mathcal{O}_\ell}[X] = \mu_\ell$

Goal

A MCTS algorithm should find the best move at the root:

$$V_s = \begin{cases} \mu_s & \text{if } s \in \mathcal{L}, \\ \max_{c\in\mathcal{C}(s)} V_c & \text{if } s \text{ is a MAX node,} \\ \min_{c\in\mathcal{C}(s)} V_c & \text{if } s \text{ is a MIN node.} \end{cases} \qquad s^* = \underset{s\in\mathcal{C}(s_0)}{\operatorname{argmax}}\, V_s$$

A PAC learning framework

MCTS algorithm: $(L_t, \tau, \hat{s}_\tau)$, where

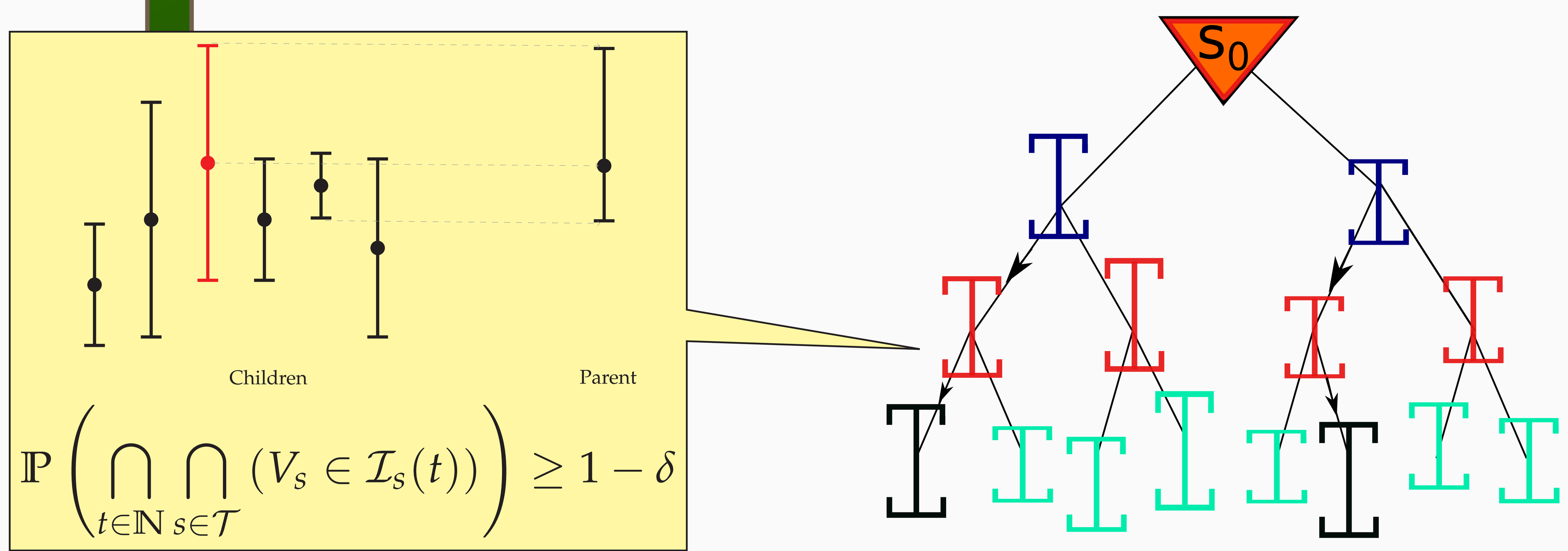
- L_t is the sampling rule
- τ is the stopping rule
- $\hat{s}_\tau \in \mathcal{C}(s_0)$ is the recommendation rule

is (ϵ, δ) – PAC if $\mathbb{P}\left(V_{\hat{s}_\tau} \geq V_{s^*} - \epsilon\right) \geq 1 - \delta.$

Goal: (ϵ, δ)-PAC algorithm with a small sample complexity τ.

Confidence Intervals at Depth One

$\ell_s(t)$: representative leaf of internal node $s \in \mathcal{T}$.



$$\mathbb{P}\left(\bigcap_{t\in\mathbb{N}} \bigcap_{s\in\mathcal{T}} \left(V_s \in \mathcal{I}_s(t)\right)\right) \geq 1 - \delta$$

Idea: alternate optimistic/pessimistic moves starting from s

BAI-MCTS Architecture

Input: a BAI algorithm
Initialization: $t = 1$.
while not BAIStop $(\{s \in \mathcal{C}(s_0)\})$ **do**
  $R_t = \text{BAIStep}\,(\{s \in \mathcal{C}(s_0)\})$
  Sample the representative leaf $L_t = \ell_{R_t}(t)$
  Update conf. intervals and representative leaves; $t = t + 1$.
end
Output: BAIReco $(\{s \in \mathcal{C}(s_0)\})$

UGapE-MCTS

Based on the UGapE algorithm [Gabillon et al., 2012]

- Sample: the least sampled among two promising nodes:

$$\underline{a_t} = \underset{a\in\mathcal{C}(s_0)}{\operatorname{argmin}}\, B_a(t) \quad \text{and} \quad \underline{b_t} = \underset{b\in\mathcal{C}(s_0)\setminus\{\underline{a_t}\}}{\operatorname{argmax}}\, \text{UCB}_b(t),$$

where $B_s(t) = \max_{s'\in\mathcal{C}(s_0)\setminus\{s\}} \text{UCB}_{s'}(t) - \text{LCB}_s(t)$.

- Stop : at time $\tau = \inf\left\{t \in \mathbb{N} : \text{UCB}_{\underline{b_t}}(t) - \text{LCB}_{\underline{a_t}}(t) < \epsilon\right\}$
- Recommend: $\hat{s}_\tau = \underline{a_\tau}$

Alternative: LUCB-MCTS, see [Kalyanakrishnan et al., 2012]

Theoretical Results

UGapE-MCTS is (ϵ, δ)-PAC for confidence intervals of the form

$$\hat{\mu}_\ell(t) \pm \sqrt{\frac{\beta(N_\ell(t), \delta)}{2N_\ell(t)}}$$

where $\beta(s, \delta) = \log(|\mathcal{L}|/\delta) + 3\log\log(|\mathcal{L}|/\delta) + (3/2)\log(\log s + 1)$.

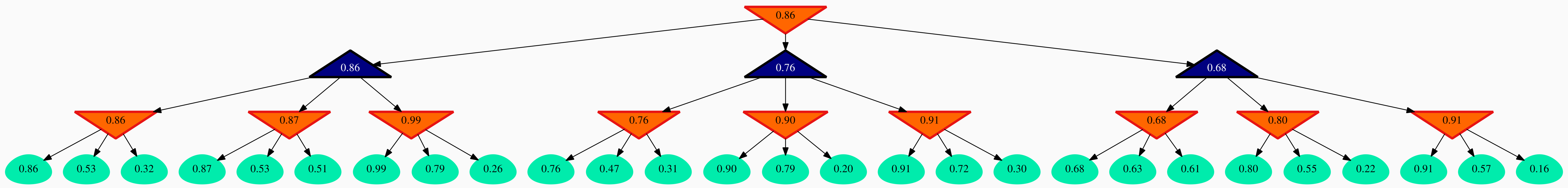
Sample complexity: $\tau = O\left(\sum_{\ell\in\mathcal{L}} \frac{1}{\Delta_\ell^2 \vee \Delta_*^2 \vee \epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$ w.p. $\geq 1 - \delta$,

where

$$\Delta_* := V(s^*) - V(s_2^*),$$

$$\Delta_\ell := \max_{s\in\texttt{Ancestors}(\ell)\setminus\{s_0\}} \left|V_{\texttt{Parent}(s)} - V_s\right|.$$

Numerical Results



	0.86	0.53	0.32	0.87	0.53	0.51	0.99	0.79	0.26	0.76	0.47	0.31	0.90	0.79	0.20	0.91	0.72	0.30	0.68	0.63	0.61	0.80	0.55	0.22	0.91	0.57	0.16
LUCB-MCTS	409	7	3	281	6	6	21	2	0	491	12	6	30	11	1	25	6	1	94	40	33	47	6	2	12	1	1
UGapE-MCTS	336	7	3	237	6	5	20	2	0	606	12	6	32	13	1	26	7	1	113	45	37	49	6	2	12	1	1
FindTopWinner	3092	187	89	3092	176	161	1267	707	36	3091	286	130	1246	1199	38	827	680	52	771	768	763	771	449	65	584	169	36

Our benchmark 3-way tree of depth 3. Shown below the leaves are the numbers of pulls of 3 algorithms: LUCB-MCTS (0.72% errors, 1551 samples), UGapE-MCTS (0.75%, 1584), and FindTopWinner (0%, 20730). Numbers are averages over 10K repetitions with $\epsilon = 0$ and $\delta = 0.1 \cdot 27$.