

Monte Carlo Tree Search for MDPs: Formal Guarantees and Symbolic Advice

GT ALGA – Journées annuelles 2021

Online

Damien Busatto-Gaston¹, Debraj Chakraborty¹,
Shibashis Guha², Guillermo A. Perez³ and Jean-Francois Raskin¹

1-Université Libre de Bruxelles

2-Tata Institute of Fundamental Research, Mumbai

3-University of Antwerp

June 16, 2021

Introduction

- ▶ Controller | Stochastic model of environment \models System
- ▶ Maximize **reward** \rightsquigarrow exploring the consequences of our decisions
- ▶ Very large systems \rightsquigarrow sparse exploration, anytime algorithms

Introduction

- ▶ Controller | Stochastic model of environment \models System
- ▶ Maximize **reward** \rightsquigarrow exploring the consequences of our decisions
- ▶ Very large systems \rightsquigarrow sparse exploration, anytime algorithms

- ▶ Monte Carlo tree search algorithm

Introduction

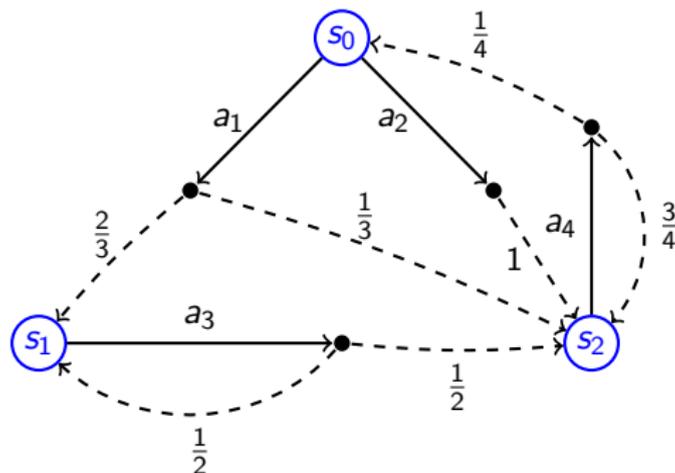
- ▶ Controller | Stochastic model of environment \models System
- ▶ Maximize **reward** \rightsquigarrow exploring the consequences of our decisions
- ▶ Very large systems \rightsquigarrow sparse exploration, anytime algorithms

- ▶ Monte Carlo tree search algorithm

- ▶ Formal guarantees
- ▶ Symbolic **advice** to guide the exploration
- ▶ Learn the model?

Playing on an MDP

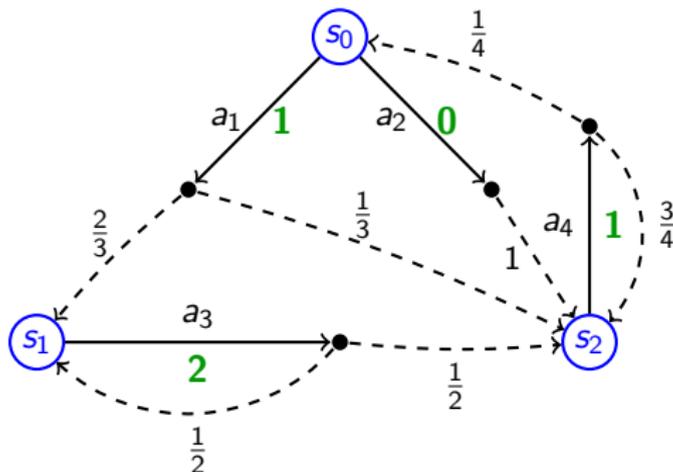
Markov Decision Process



- ▶ Path of length 2: $s_0 \xrightarrow{a_1} \text{dot} \xrightarrow{\frac{2}{3}} s_1 \xrightarrow{a_3} \text{dot} \xrightarrow{\frac{1}{2}} s_2$
- ▶ Strategy: $\sigma : S \rightarrow A$

Playing on an MDP

Markov Decision Process



▶ Path of length 2: $s_0 \xrightarrow{a_1} \xrightarrow{\frac{2}{3}} s_1 \xrightarrow{a_3} \xrightarrow{\frac{1}{2}} s_2$

▶ Strategy: $\sigma : S \rightarrow A$

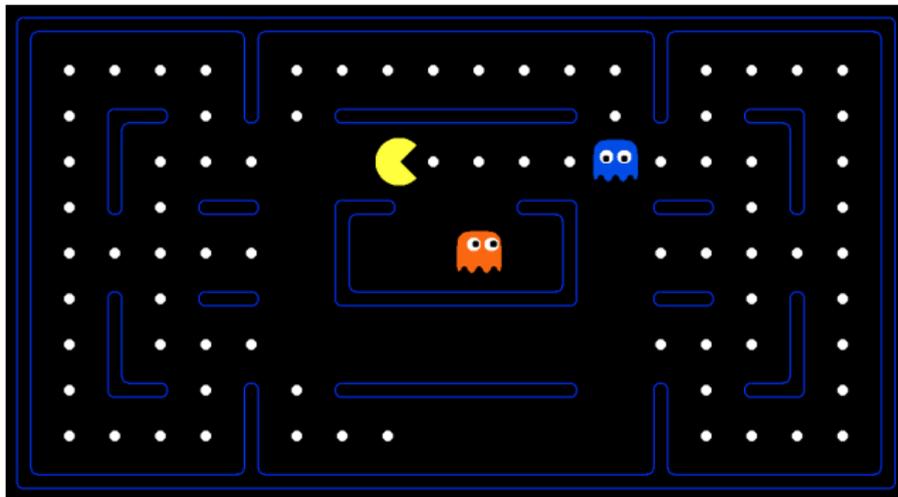
▶ Infinite-horizon average reward:

$$\text{Val}(s_0, \sigma) = \lim_{H \rightarrow \infty} \frac{1}{H} \mathbb{E} [\text{Reward}(p)]$$

where p is a random variable over $\text{Paths}^H(s_0, \sigma)$

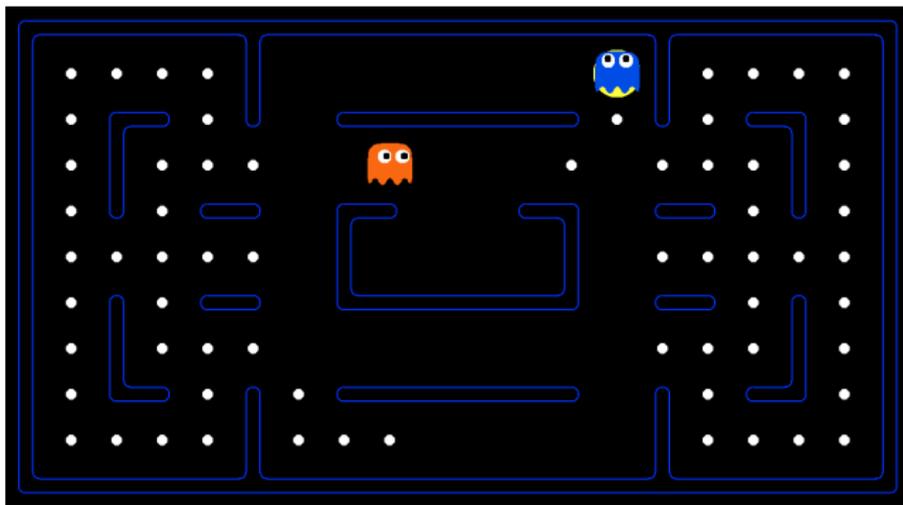
▶ $\text{Val}(s_0) = \max_{\sigma: S \rightarrow A} \text{Val}(s_0, \sigma)$

Example: Pac-Man as an MDP



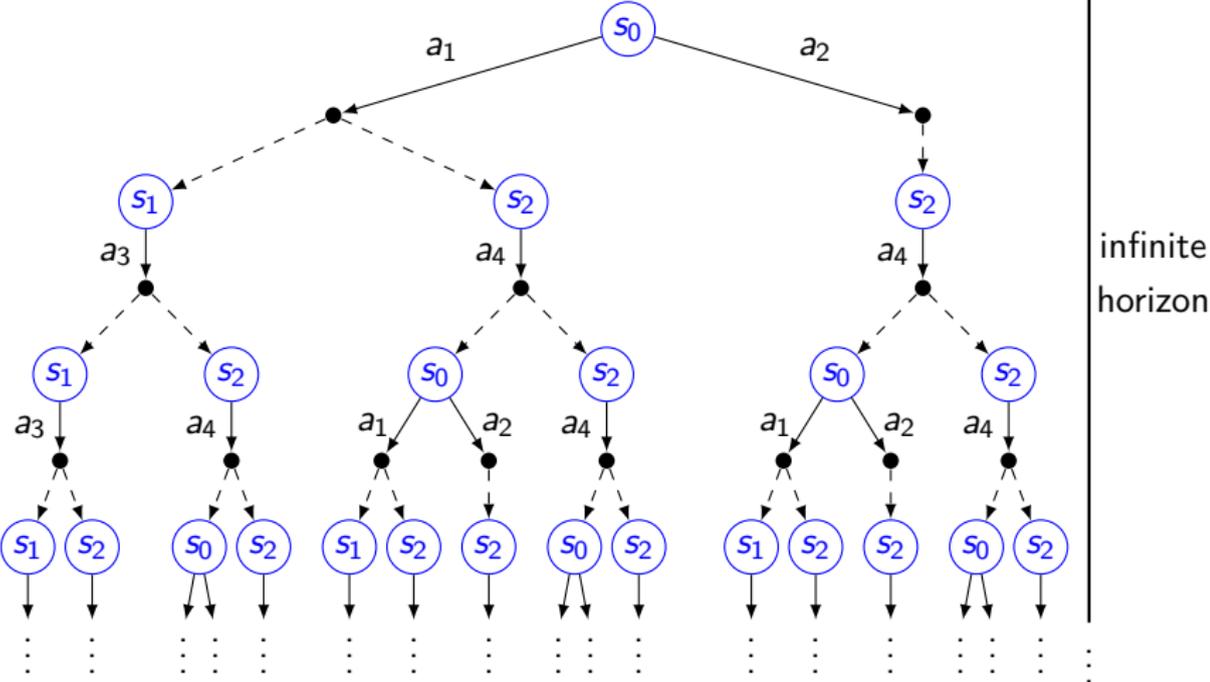
- ▶ Controller: Pac-Man
- ▶ Probabilistic model of ghosts
- ▶ **States:** position of every agent, what food is left
- ▶ Actions: Pac-Man moves
- ▶ Stochastic transitions: ghost moves

Example: Pac-Man as an MDP



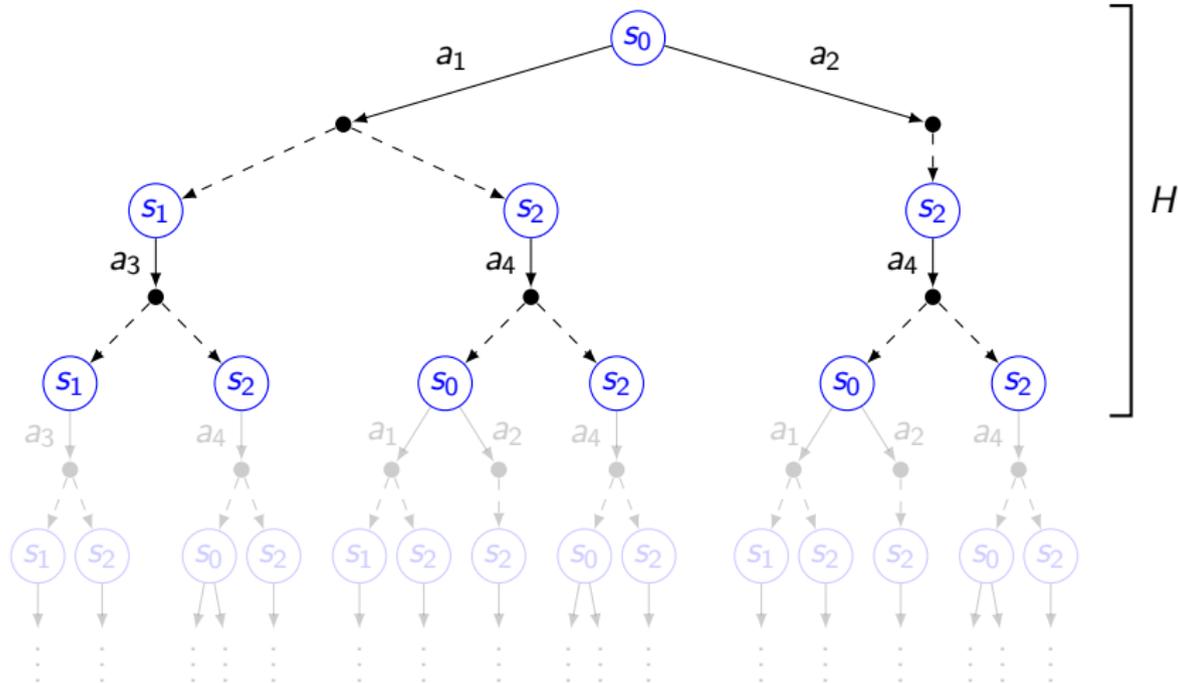
- ▶ Controller: Pac-Man
- ▶ Probabilistic model of ghosts
- ▶ Reward for eating food
- ▶ Large penalty for losing
- ▶ States: position of every agent, what food is left
- ▶ Actions: Pac-Man moves
- ▶ Stochastic transitions: ghost moves
- ▶ Large MDP: $\sim 10^{16}$ states

Receding horizon



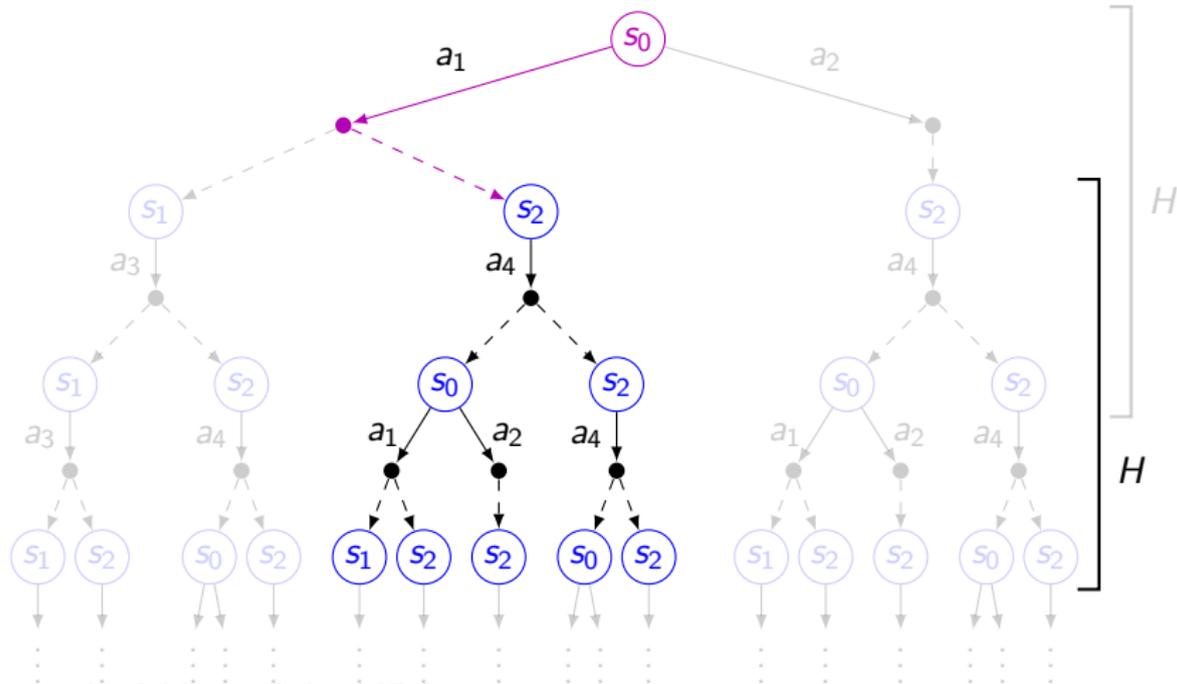
► Unfolding of the MDP

Receding horizon



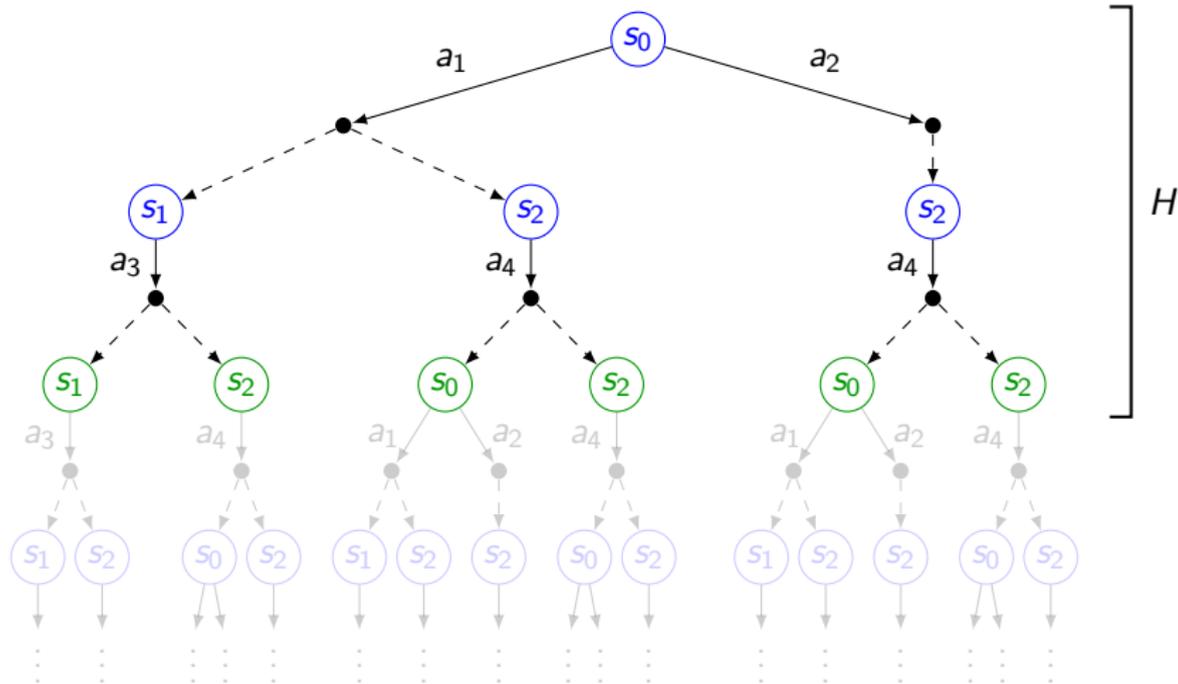
- ▶ Unfolding of the MDP
- ▶ Finite horizon computation of the best action: **total reward**
- ▶ Sliding window of depth H

Receding horizon



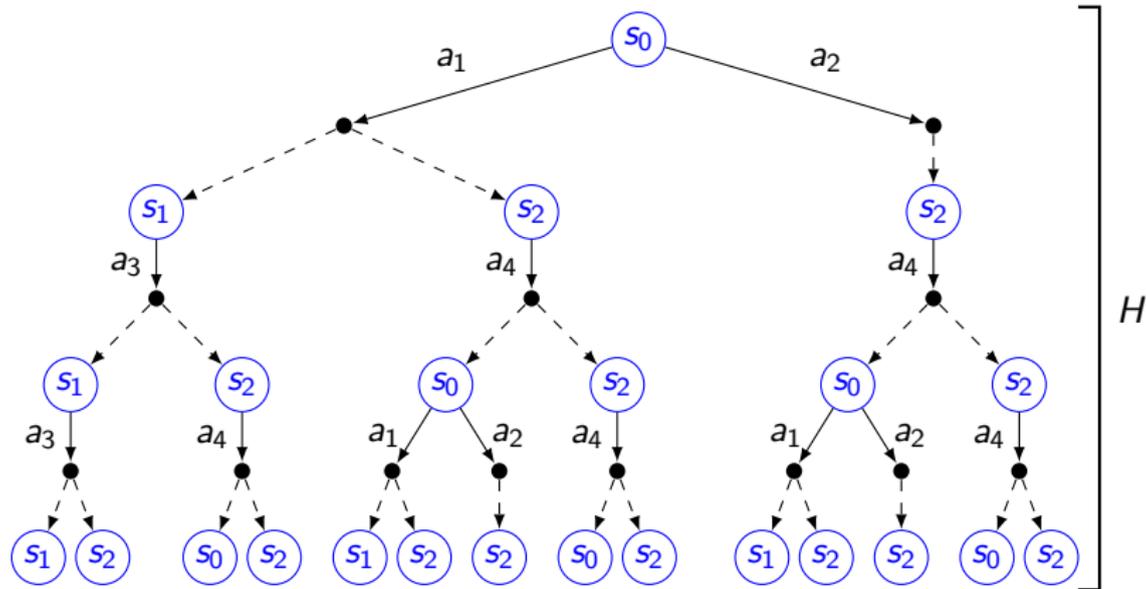
- ▶ Unfolding of the MDP
- ▶ Finite horizon computation of the best action: **total reward**
- ▶ Sliding window of depth H
- ▶ H big enough \leadsto optimal strategy

Final rewards



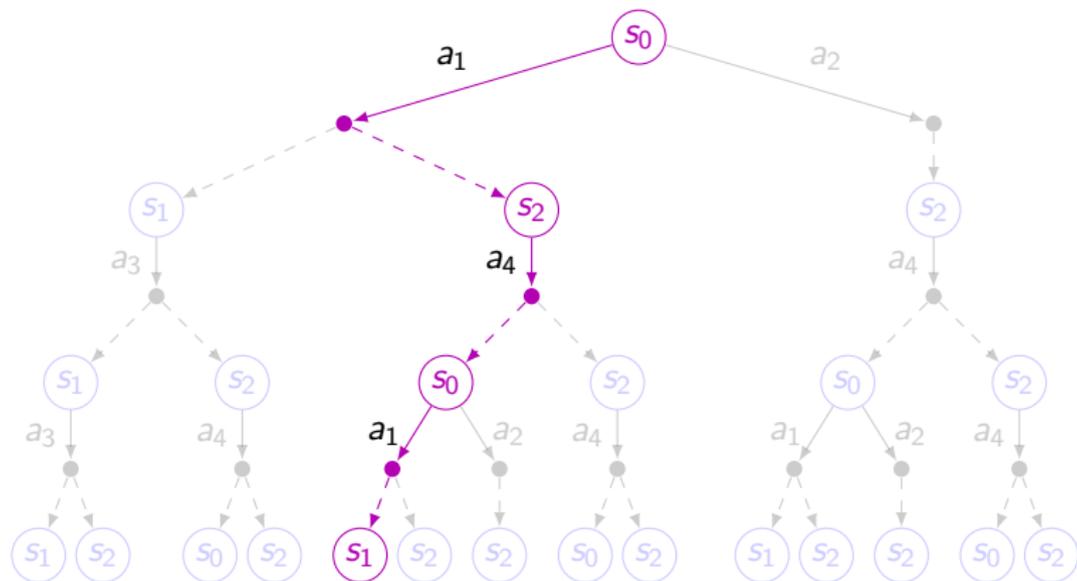
- ▶ H not big enough \leadsto rewards on leaves
- ▶ Estimations for long-term behaviours

Sparse exploration



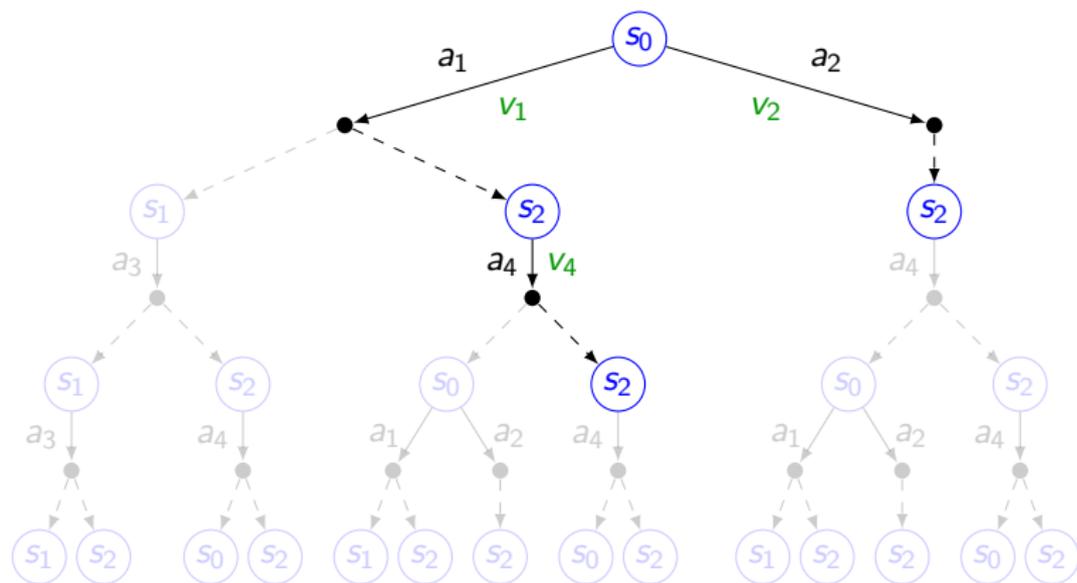
► Large unfolding \leadsto heuristics

Sparse exploration



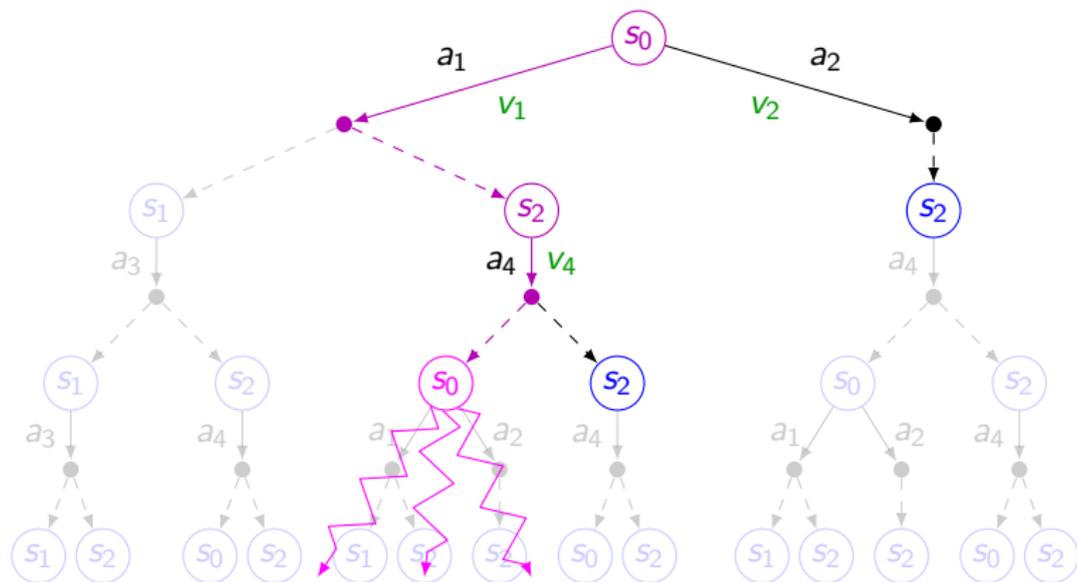
- ▶ Large unfolding \leadsto heuristics
- ▶ Uniform **simulation**: select actions at random to obtain a path
- ▶ Average **reward** over a few simulations \leadsto estimate of $\text{Val}^H(s_0)$
- ▶ No formal guarantees of convergence

Monte Carlo tree search (MCTS)



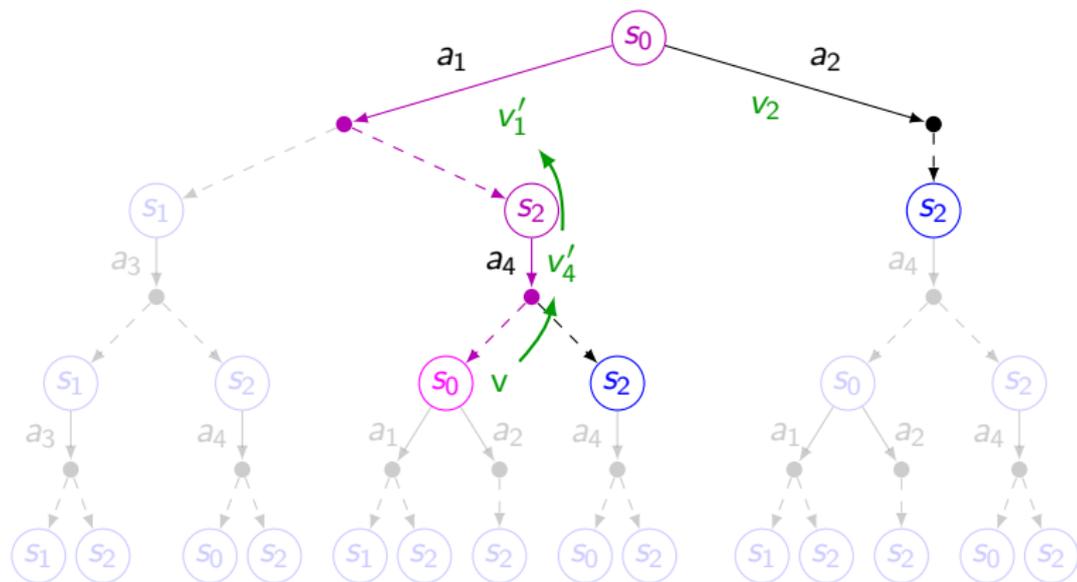
- Iterative construction of a sparse tree with **value estimates**

Monte Carlo tree search (MCTS)



- ▶ Iterative construction of a sparse tree with **value estimates**
- ▶ **Selection** of a new node \rightsquigarrow **simulation**

Monte Carlo tree search (MCTS)



- ▶ Iterative construction of a sparse tree with **value estimates**
- ▶ **Selection** of a new node \rightsquigarrow **simulation** \rightsquigarrow update of the **estimates**
- ▶ MCTS converges to the optimal choice (Kocsis & Szepesvári, 2006)

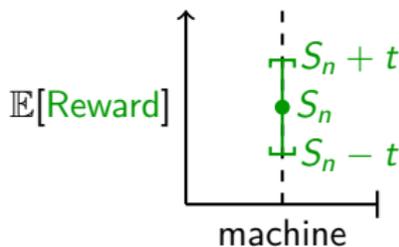
Theoretical guarantees

Sampling an unknown distribution



- ▶ Consider a slot machine (one-armed bandit)
- ▶ hidden **reward distribution**
- ▶ Estimate the expected reward?

Sampling an unknown distribution



- ▶ Consider a slot machine (one-armed bandit)
- ▶ hidden reward distribution
- ▶ Estimate the expected reward?

Chernoff-Hoeffding inequalities

Let X_1, X_2, \dots, X_n be indep. random variables in $[0, 1]$, $S_n = \frac{1}{n} \sum_n X_i$.

- ▶ $\mathbb{P}[\mathbb{E}[S_n] \geq S_n + t] \leq \exp(-2nt^2)$
- ▶ $\mathbb{P}[\mathbb{E}[S_n] \leq S_n - t] \leq \exp(-2nt^2)$.

Multi-armed bandit and UCB algorithm



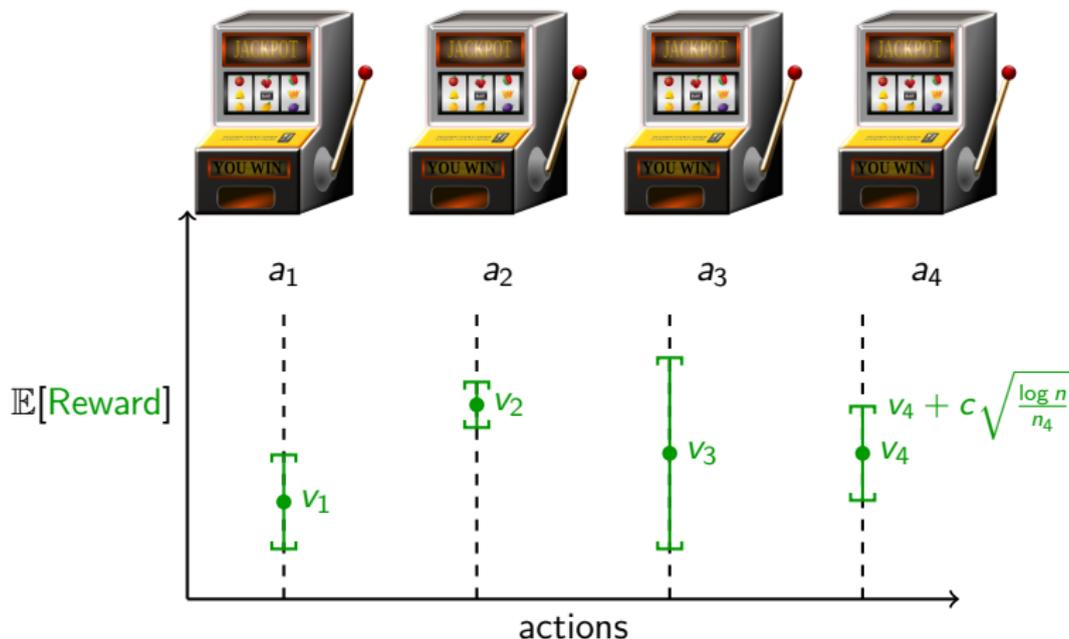
- ▶ Finite set of machines (actions), that give rewards when played
- ▶ Every machine has a hidden **reward distribution**
- ▶ How to find the best machine (expected reward)?
- ▶ Take samples according to a strategy, try to minimize regret

Multi-armed bandit and UCB algorithm



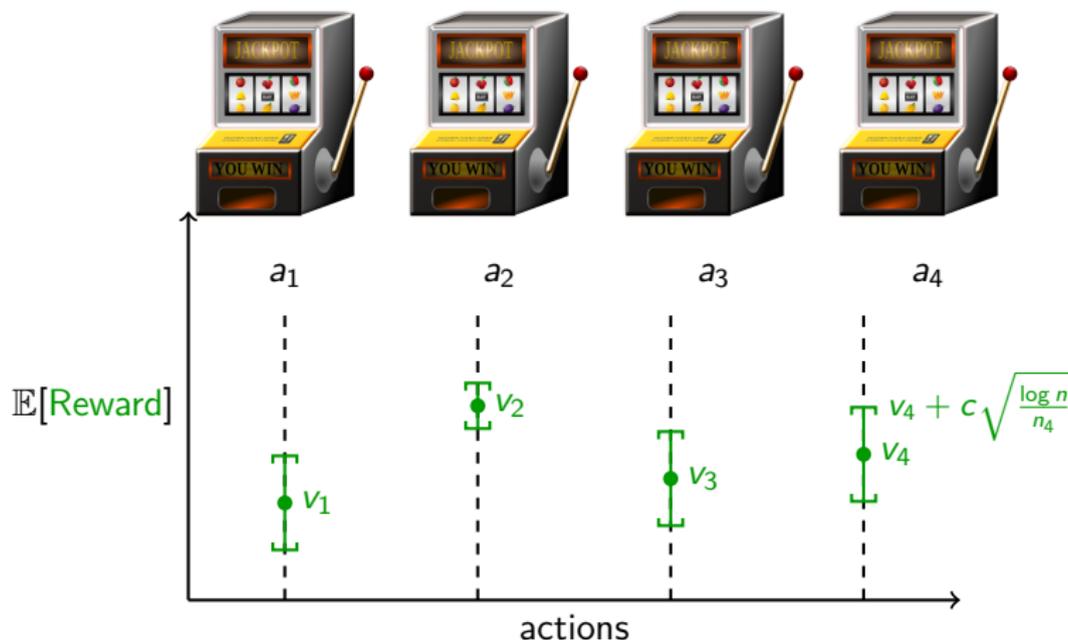
- ▶ Finite set of machines (actions), that give rewards when played
- ▶ Every machine has a hidden **reward distribution**
- ▶ How to find the best machine (expected reward)?
- ▶ Take samples according to a strategy, try to minimize regret
- ▶ UCB (Auer, Cesa-Bianchi, & Fischer, 2002) is a popular strategy
- ▶ It offers a solution to the exploitation/exploration trade-off
- ▶ Optimal: regret is bounded logarithmically

Upper-Confidence Bounds



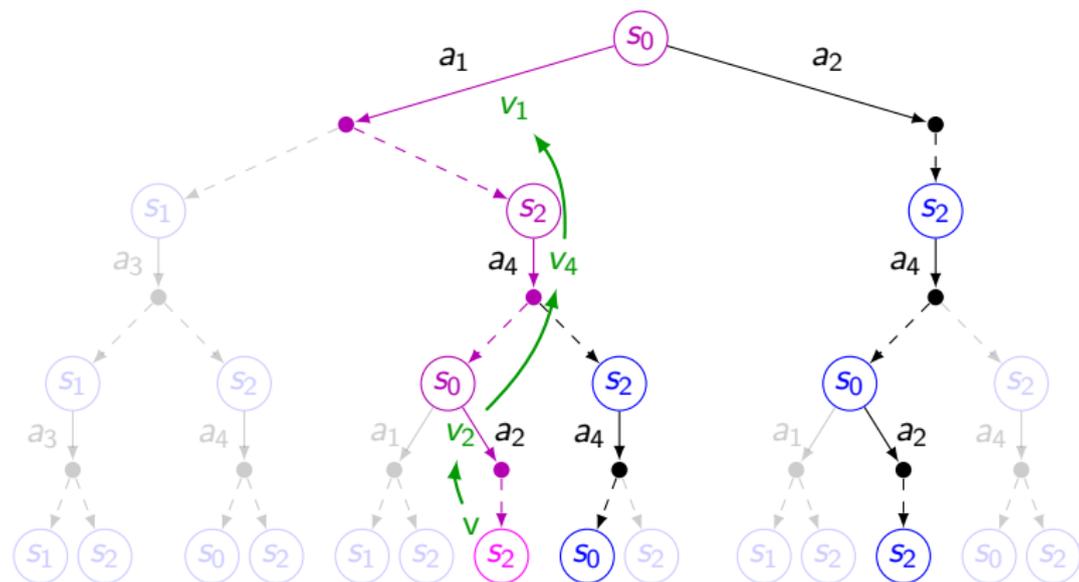
- ▶ confidence intervals around our observations
- ▶ UCB chooses the action with highest upper bound
- ▶ Optimism in the Face of Uncertainty

Upper-Confidence Bounds



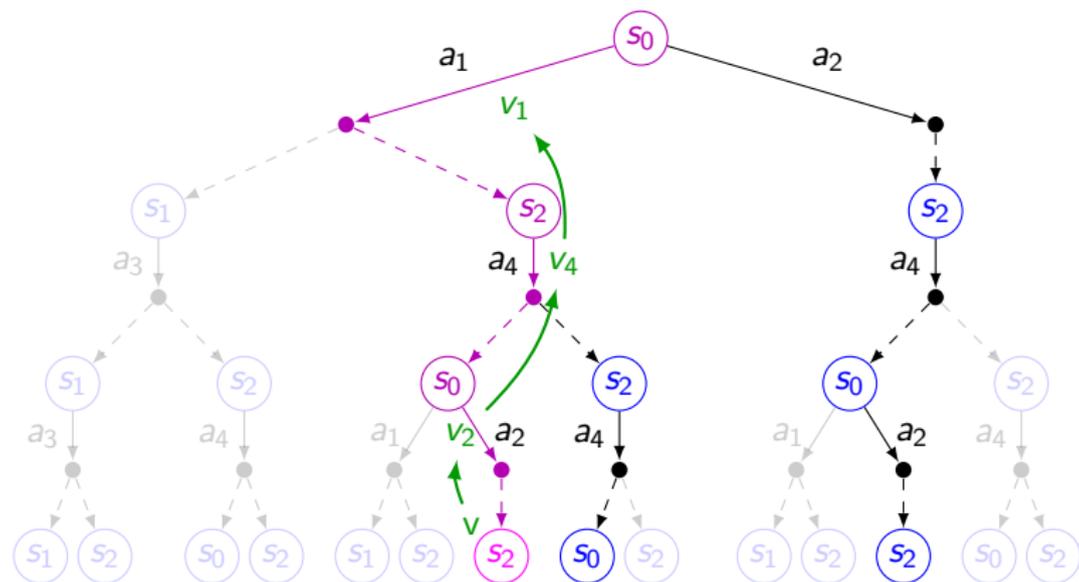
- ▶ confidence intervals around our observations
- ▶ UCB chooses the action with highest upper bound
- ▶ Optimism in the Face of Uncertainty

The MCTS algorithm using UCB (Kocsis & Szepesvári, 2006)



- ▶ Every state is seen as an instance of a bandit problem
- ▶ Selecting an action \leadsto reward in the backwards propagation phase

The MCTS algorithm using UCB (Kocsis & Szepesvári, 2006)



- ▶ Every **state** is seen as an instance of a bandit problem
- ▶ **Selecting** an action \leadsto **reward** in the backwards propagation phase
- ▶ Using UCB for selection \leadsto the rewards **change over time**
- ▶ Non-stationary bandits with **Drift conditions**

Non-stationary bandits and drift conditions



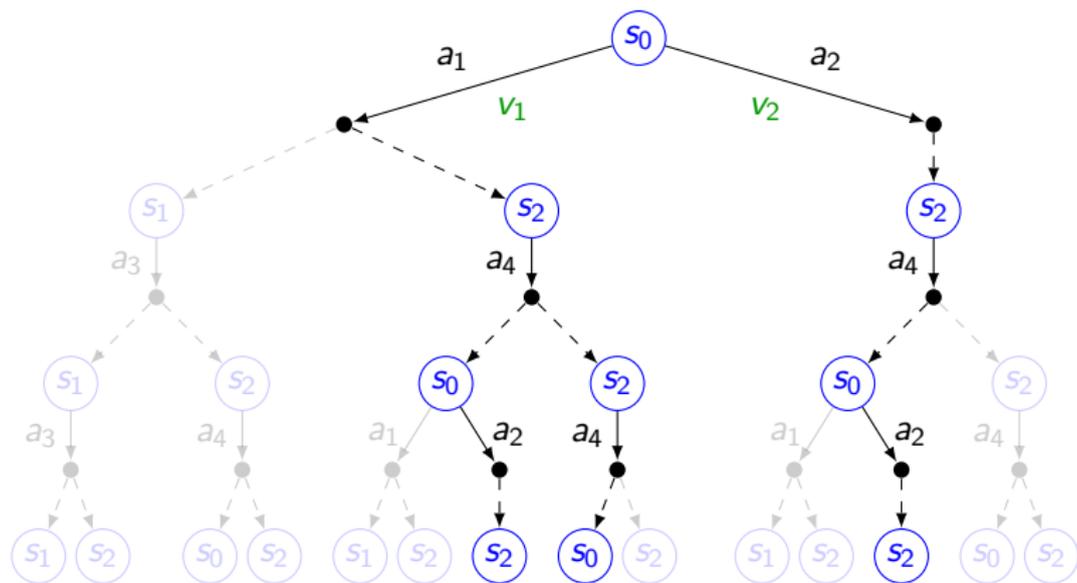
- ▶ The reward distributions change after each play
- ▶ They must follow some assumptions (**Drift conditions**):
 - ▶ The expected average reward of the first n plays of a converges
 - ▶ Tail inequalities: same shape as Chernoff-Hoeffding

Non-stationary bandits and drift conditions



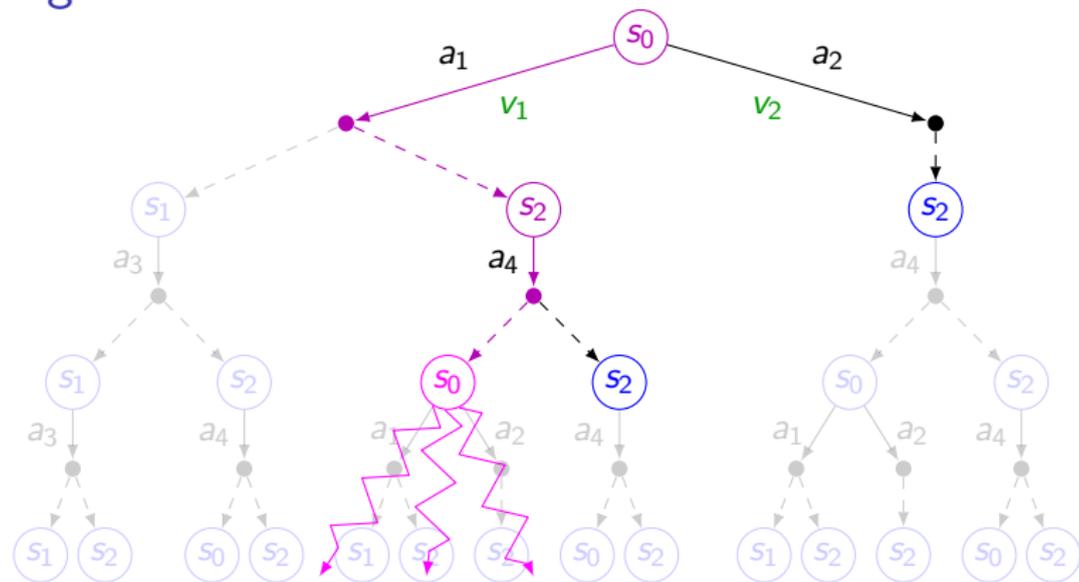
- ▶ The reward distributions change after each play
- ▶ They must follow some assumptions (**Drift conditions**):
 - ▶ The expected average reward of the first n plays of a converges
 - ▶ Tail inequalities: same shape as Chernoff-Hoeffding
- ▶ UCB can be extended under these assumptions
- ▶ When using UCB for selecting actions in MCTS, the reward distributions satisfy the drift conditions (**Kocsis & Szepesvári, 2006**)

Convergence of MCTS (Kocsis & Szepesvári, 2006)



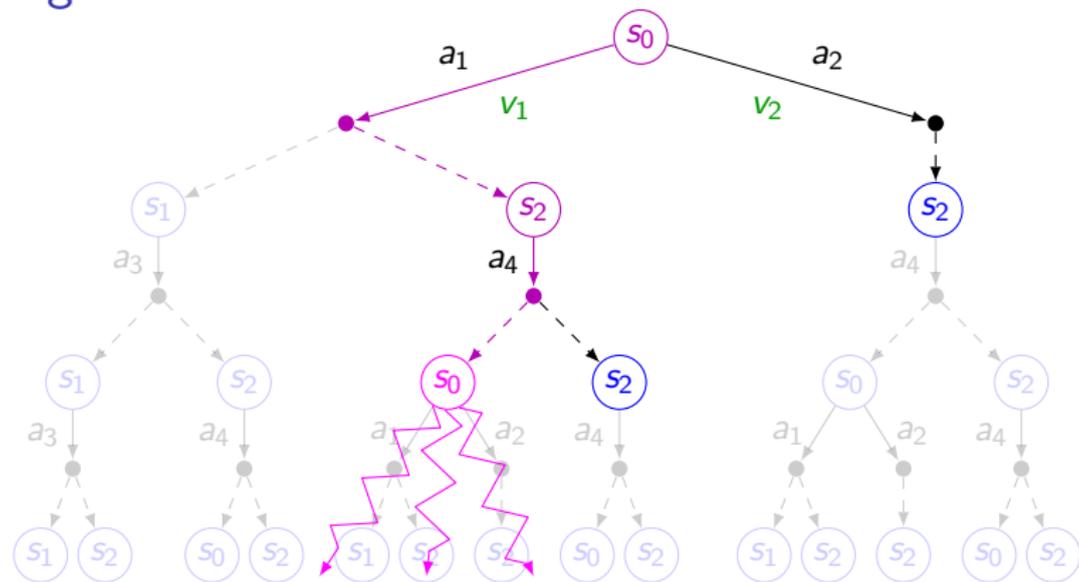
- ▶ After a given number of iterations n , MCTS outputs the best action
- ▶ The probability of choosing a suboptimal action converges to zero
- ▶ v_i converges to the real value of a_i at a speed of $(\log n)/n$

Convergence of MCTS with simulation



- ▶ Unlike (Kocsis & Szepesvári, 2006), MCTS is often implemented with a **simulation phase** used to initialise **value estimates**
- ▶ This changes the **reward distributions** of all UCB instances

Convergence of MCTS with simulation



- ▶ Unlike (Kocsis & Szepesvári, 2006), MCTS is often implemented with a **simulation phase** used to initialise **value estimates**
- ▶ This changes the **reward distributions** of all UCB instances
- ▶ We show that the convergence properties of MCTS are maintained **for all** simulations: any strategy can be used to draw samples

Recent patch to MCTS



- ▶ The proof of (Kocsis & Szepesvári, 2006) is incomplete
- ▶ random variables assumed to be independent are not

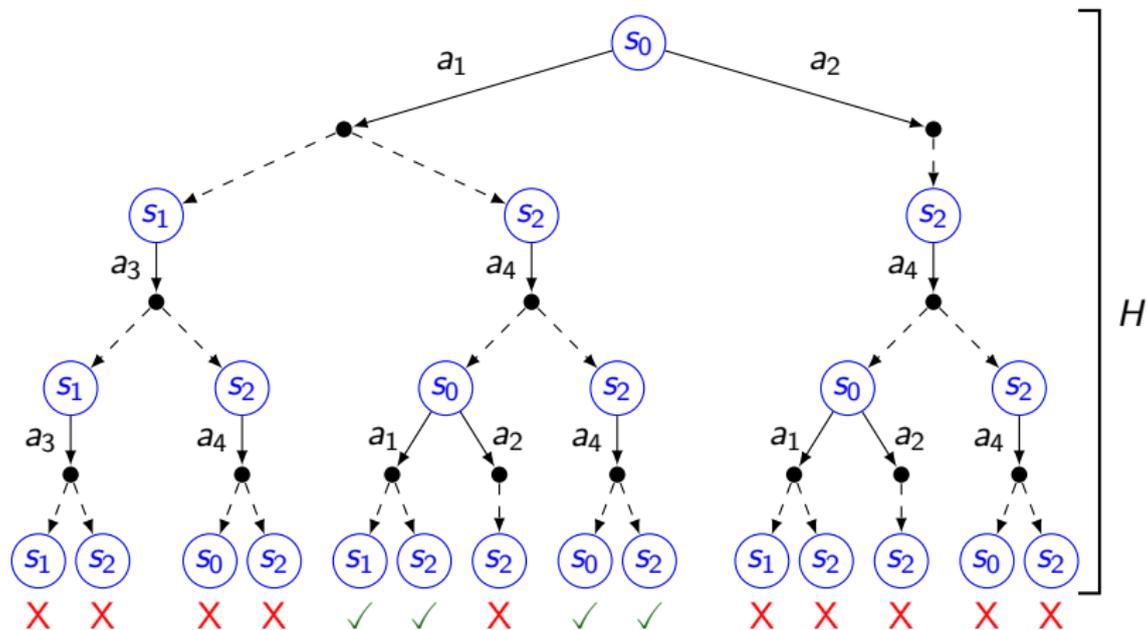
Recent patch to MCTS



- ▶ The proof of (Kocsis & Szepesvári, 2006) is incomplete
- ▶ random variables assumed to be independent are not
- ▶ *Non-Asymptotic Analysis of Monte Carlo Tree Search - SIGMETRICS '20*, by (Shah, Xie, & Xu, 2020) fixed it!
- ▶ polynomial bias: \sqrt{n} instead of $\log(n)$

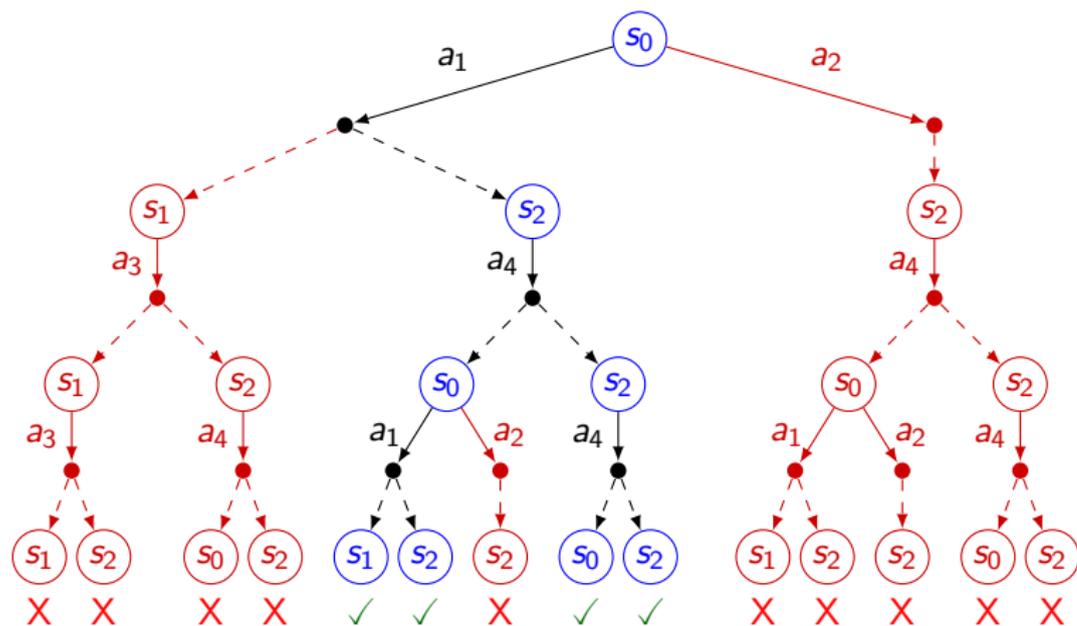
Symbolic advice

Symbolic advice



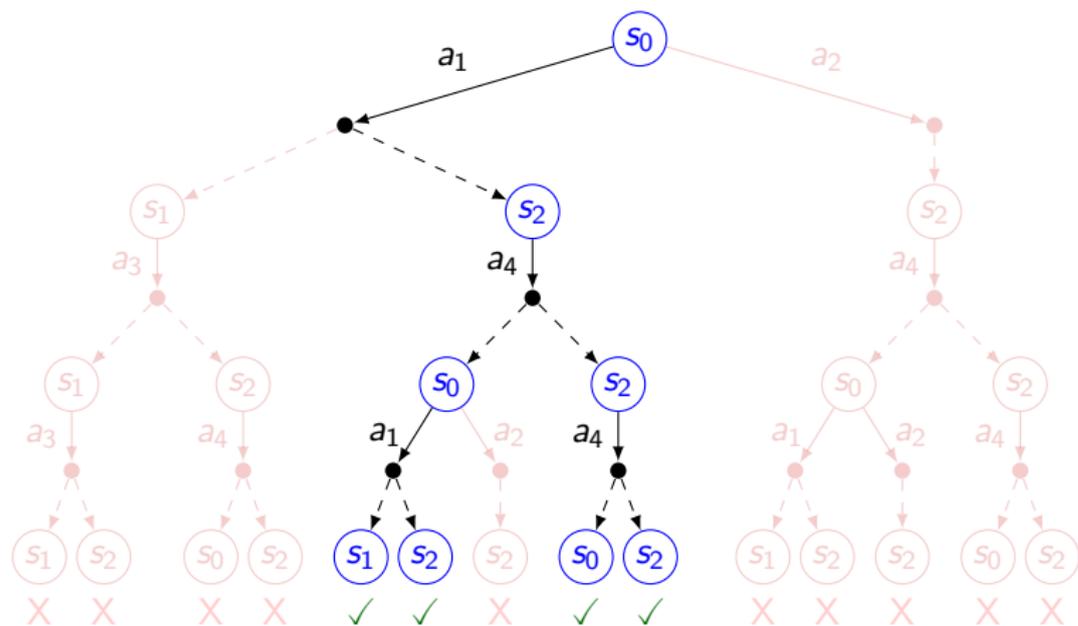
- ▶ An **advice** is a subset of $\text{Paths}^H(s_0)$
- ▶ Defined symbolically as a logical formula φ (reachability or safety property, LTL formula over finite traces, regular expression ...)

Symbolic advice



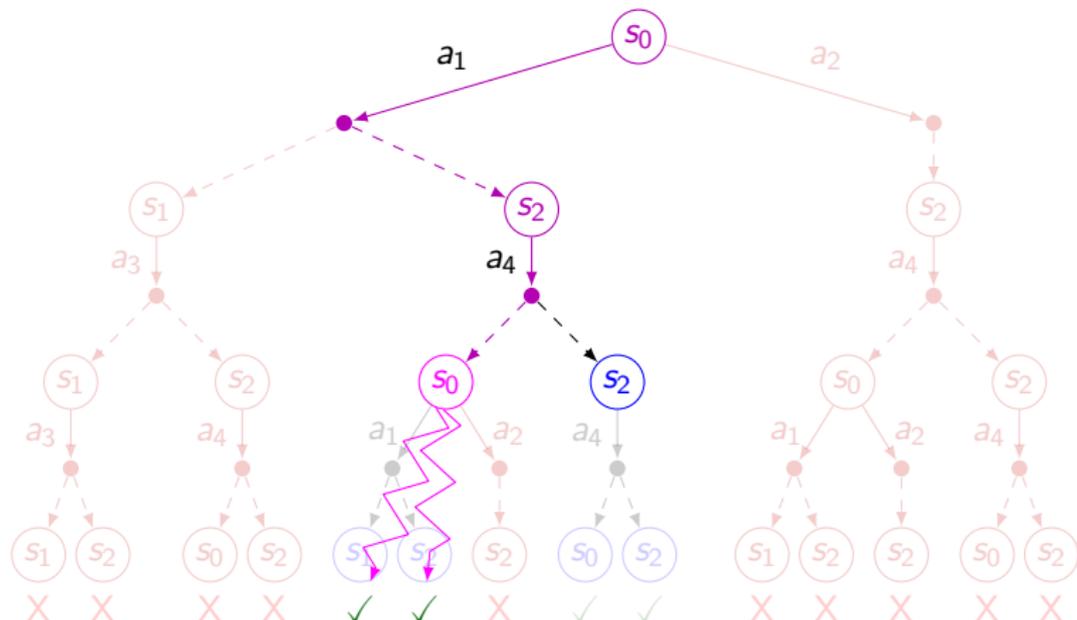
- ▶ An **advice** is a subset of $\text{Paths}^H(s_0)$
- ▶ Defined symbolically as a logical formula φ (reachability or safety property, LTL formula over finite traces, regular expression ...)

Symbolic advice



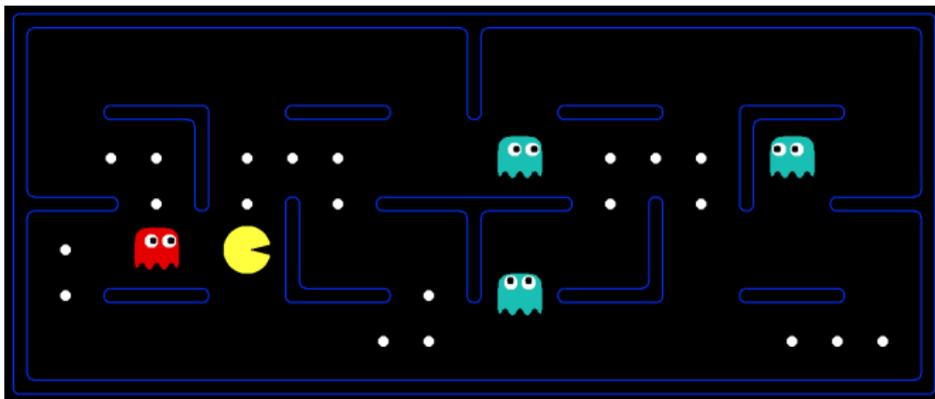
- ▶ An **advice** is a subset of $\text{Paths}^H(s_0)$
- ▶ Defined symbolically as a logical formula φ (reachability or safety property, LTL formula over finite traces, regular expression ...)
- ▶ φ defines a pruning of the unfolded MDP

MCTS under advice



- ▶ **Select** actions in the unfolding pruned by a **selection advice** φ
- ▶ **Simulation** is restricted according to a **simulation advice** ψ

Safety property



- ▶ Some states are unsafe and should be avoided
- ▶ **Advice** ψ : set of safe paths **G.** $(x, y)_p \neq (x, y)_g$
- ▶ Stronger property: safety is **ensured** no matter what stochastic transitions are taken
- ▶ **Enforceable advice** φ : set of paths so that every action chosen is compatible with a strategy that enforces safety with horizon H

Boolean Solvers

- ▶ The **safety property** ψ can be encoded as a Boolean Formula

Boolean Solvers

- ▶ The **safety property** ψ can be encoded as a Boolean Formula

QBF solver

- ▶ A first action a_0 is compatible with φ iff

$$\forall s_1 \exists a_1 \forall s_2 \dots, s_0 a_0 s_1 a_1 s_2 \dots \models \psi$$

- ▶ Inductive way of constructing paths that satisfy the **enforceable advice** φ
- ▶ Alternation of quantifiers \rightsquigarrow guarantee safety for $h < H$

Boolean Solvers

- ▶ The **safety property** ψ can be encoded as a Boolean Formula

QBF solver

- ▶ A first action a_0 is compatible with φ iff

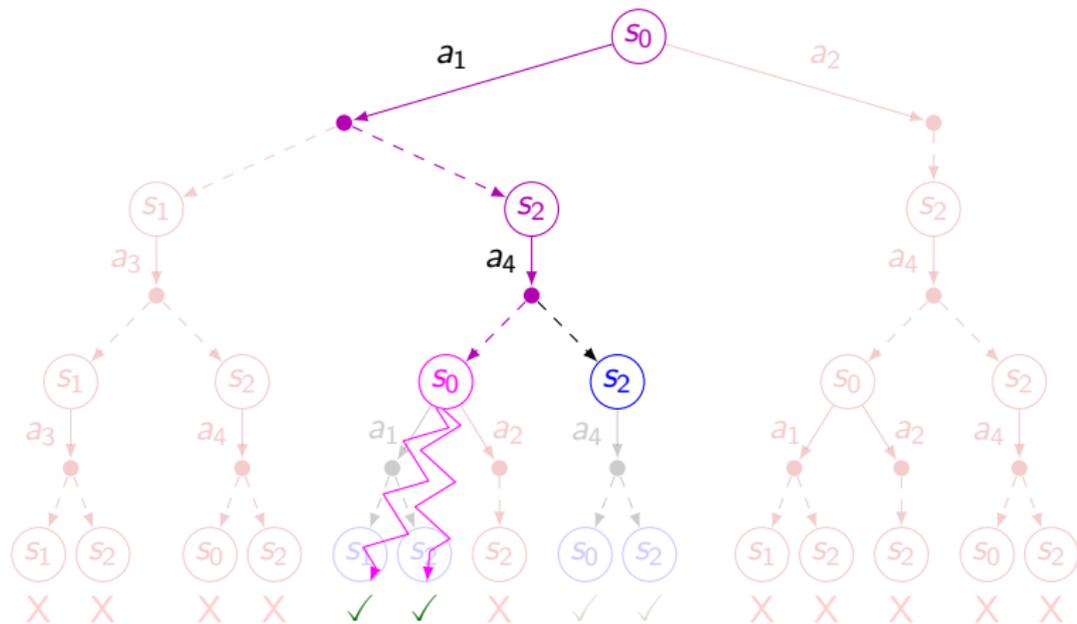
$$\forall s_1 \exists a_1 \forall s_2 \dots, s_0 a_0 s_1 a_1 s_2 \dots \models \psi$$

- ▶ Inductive way of constructing paths that satisfy the **enforceable advice** φ
- ▶ Alternation of quantifiers \rightsquigarrow guarantee safety for $h < H$

Weighted sampling

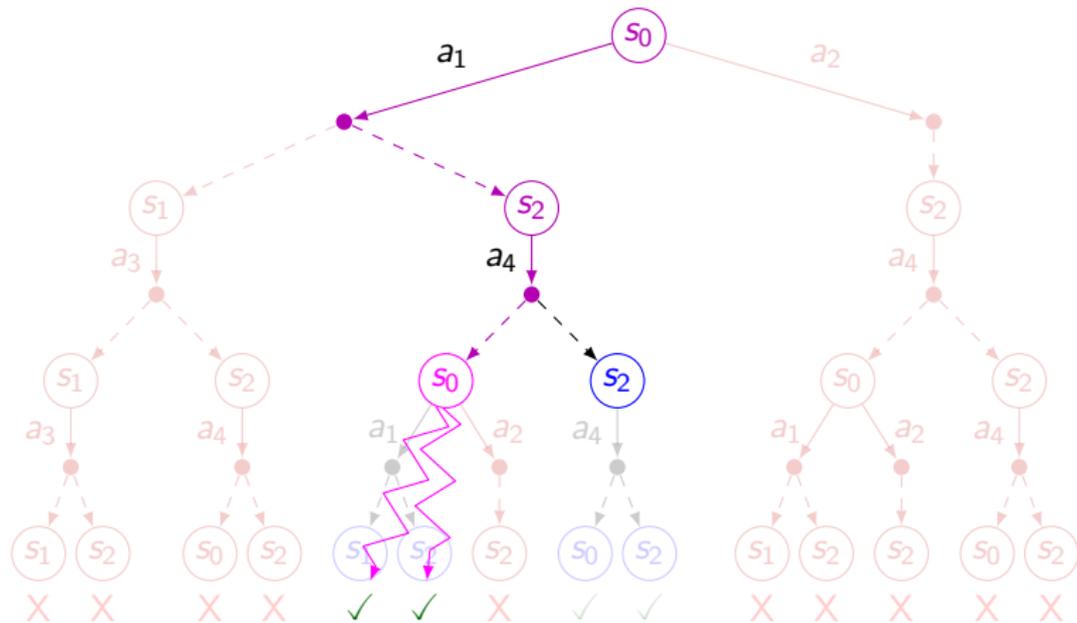
- ▶ Simulation of safe paths according to ψ
- ▶ Weighted SAT sampling (Chakraborty, Fremont, Meel, Seshia, & Vardi, 2014)

MCTS under advice



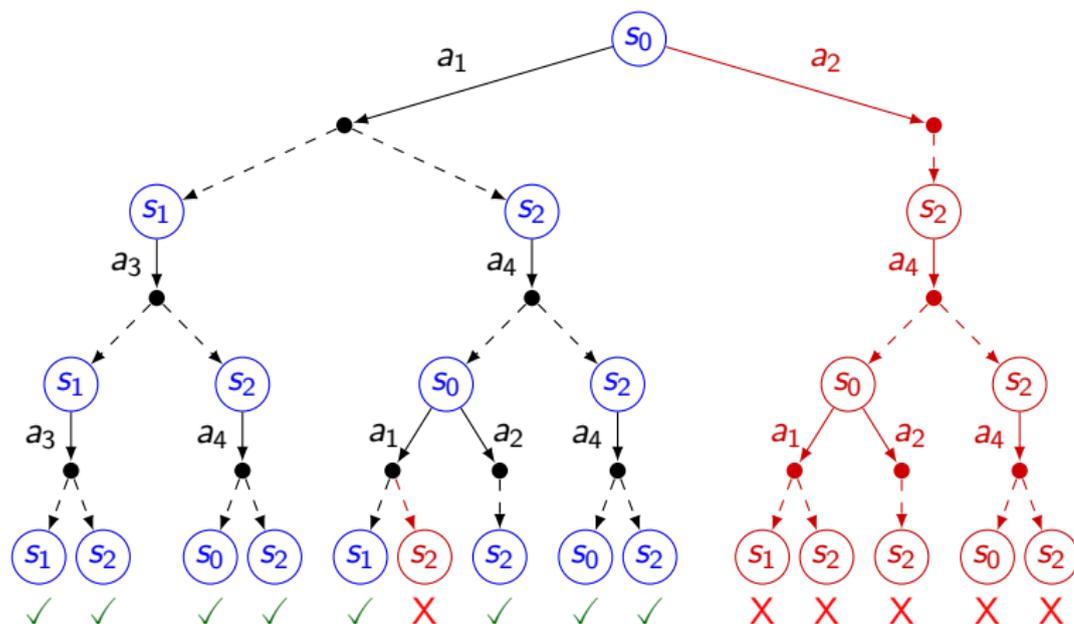
- ▶ **Select** actions in the unfolding pruned by a **selection advice** φ
- ▶ **Simulation** is restricted according to a **simulation advice** ψ

MCTS under advice



- ▶ **Select** actions in the unfolding pruned by a **selection advice** φ
- ▶ **Simulation** is restricted according to a **simulation advice** ψ
- ▶ We show that the convergence properties are maintained:
 - ▶ for a **selection advice** that satisfies some **assumptions**,
 - ▶ for **all simulation advice**.

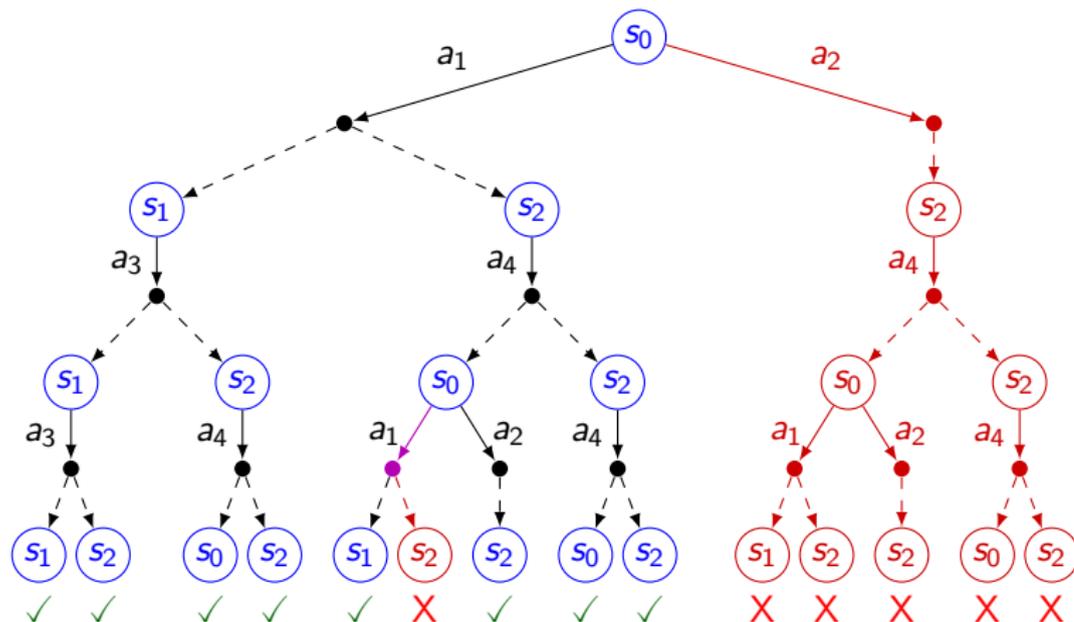
Assumptions on the selection advice



The selection advice must

- ▶ be **strongly enforceable**: can be enforced by controller if the MDP is seen as a game \rightsquigarrow does not partially prune stochastic transitions

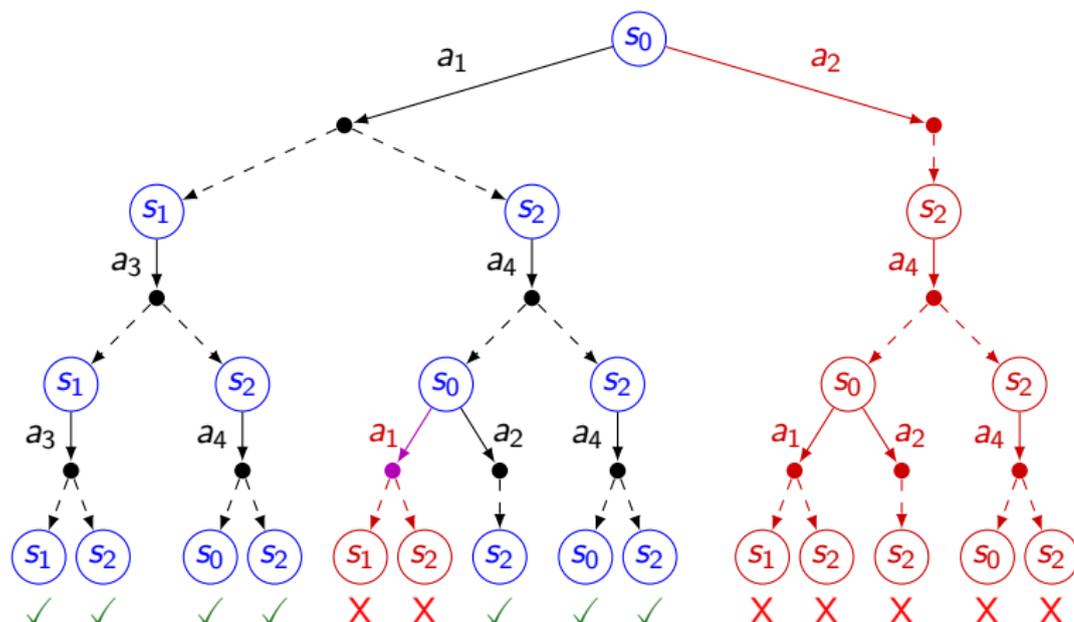
Assumptions on the selection advice



The selection advice must

- ▶ be **strongly enforceable**: can be enforced by controller if the MDP is seen as a game \rightsquigarrow does not partially prune stochastic transitions

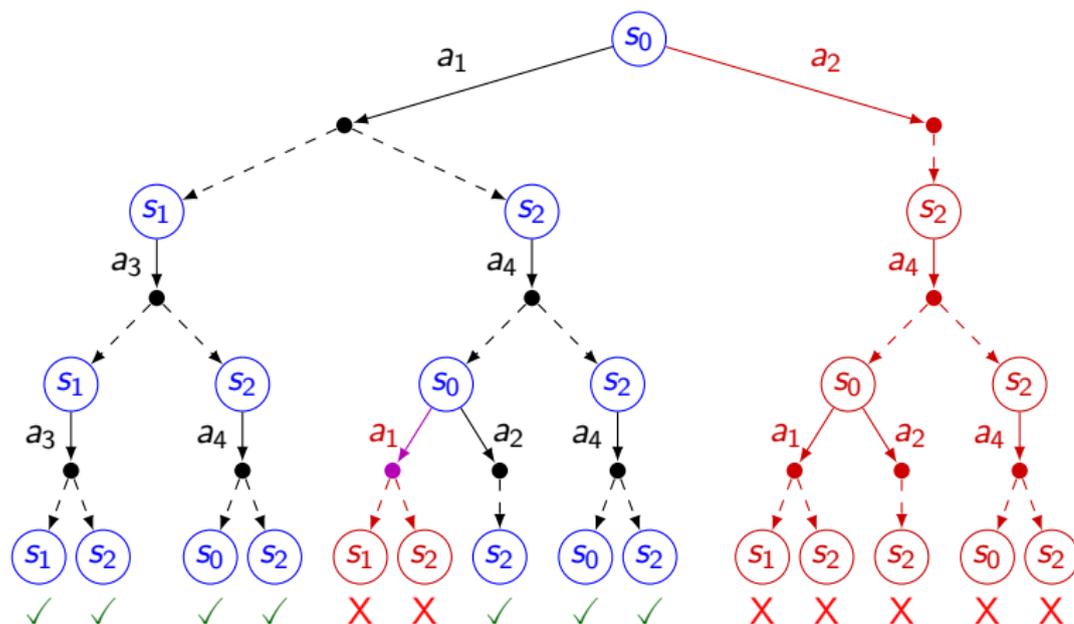
Assumptions on the selection advice



The selection advice must

- ▶ be **strongly enforceable**: can be enforced by controller if the MDP is seen as a game \rightsquigarrow does not partially prune stochastic transitions

Assumptions on the selection advice

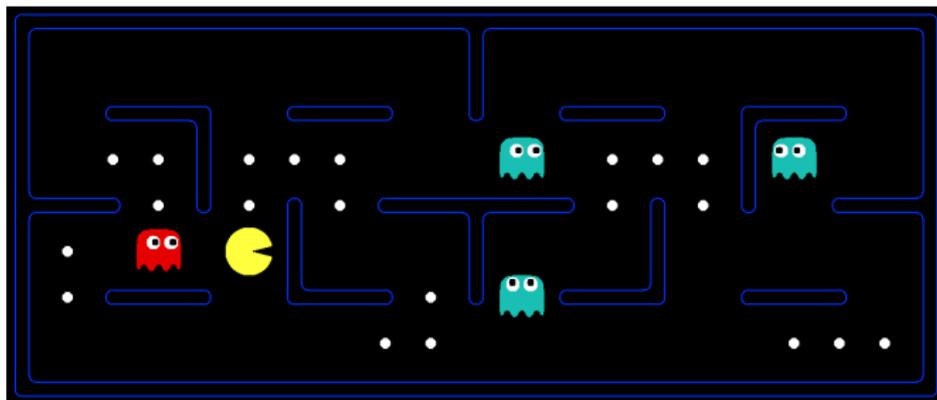


The **selection advice** must

- ▶ be **strongly enforceable**: can be enforced by controller if the MDP is seen as a game \rightsquigarrow does not partially prune stochastic transitions
- ▶ satisfy an **optimality assumption**: does not prune all optimal actions

Experimental results

Experimental results



9 × 21 maze, 4 random ghosts

Algorithm	win	loss	no result after 300 steps	food
MCTS	17	59	24	67
MCTS+Selection advice	25	54	21	71
MCTS+Simulation advice	71	29	0	88
MCTS+both advice	85	15	0	94
Human	44	56	0	75

Conclusion

Contributions

- ▶ How to inject domain knowledge in MCTS?
 - ▶ symbolic **advice** for selection and simulation
- ▶ How to preserve the convergence guarantees of MCTS?
 - ▶ strongly **enforceable** advice with an optimality assumption
- ▶ How to implement them?
 - ▶ symbolic solutions using **SAT** and QBF solvers
- ▶ Does it work on large MDPs?
 - ▶ good results with safety advice on the **Pac-Man** domain
- ▶ What if the MDP is not known?
 - ▶ learn it?
 - ▶ paper on a scheduling problem to appear in QEST '21

Conclusion

Contributions

- ▶ How to inject domain knowledge in MCTS?
 - ▶ symbolic **advice** for selection and simulation
- ▶ How to preserve the convergence guarantees of MCTS?
 - ▶ strongly **enforceable** advice with an optimality assumption
- ▶ How to implement them?
 - ▶ symbolic solutions using **SAT** and QBF solvers
- ▶ Does it work on large MDPs?
 - ▶ good results with safety advice on the **Pac-Man** domain
- ▶ What if the MDP is not known?
 - ▶ learn it?
 - ▶ paper on a scheduling problem to appear in QEST '21

Current and future works

- ▶ Support prism format for MDPs, LTL advice
- ▶ Study interactions with **reinforcement learning** techniques (and neural networks)

Thank you