Monte-Carlo Graph Search:

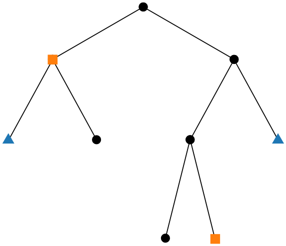
the Value of Merging Similar States

Edouard Leurent[1,2],
Odalric-Ambrym Maillard[1]

[1]Univ. Lille, Inria, CNRS,
Centrale Lille, UMR 9189 – CRIStAL,
[2]Renault Group

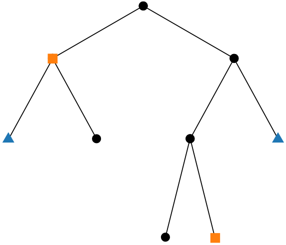
Monte-Carlo Tree Search algorithms

- rely on a tree structure to represent their value estimates.



Monte-Carlo Tree Search algorithms

- rely on a tree structure to represent their value estimates.

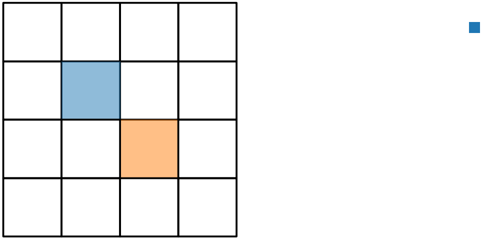


- performance independent of the size S of the state space

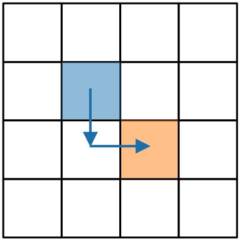
Tabular RL	(UCBVI)	$\sqrt{HSAn}$
MCTS	(OPD)	$n^{-\log\frac{1}{\gamma}/\log A}$

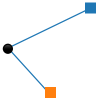
Limitation

- There can be several paths to the same state s

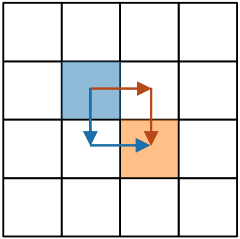


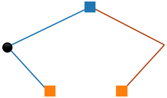
- There can be several paths to the same state s



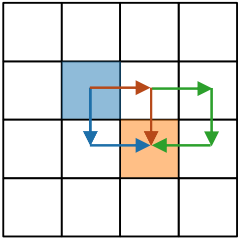


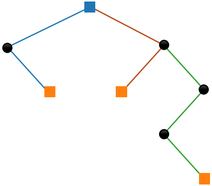
- There can be several paths to the same state s



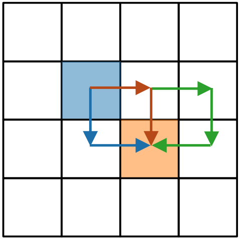


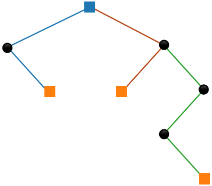
- There can be several paths to the same state *s*





- There can be several paths to the same state s

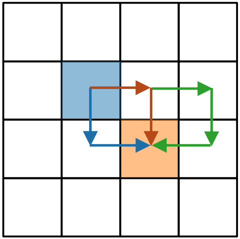


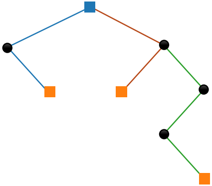


- s is represented several times in the tree

Limitation

- There can be several paths to the same state s



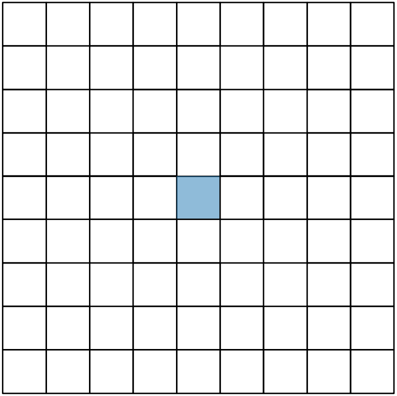


- s is represented several times in the tree
- No information is shared between these paths

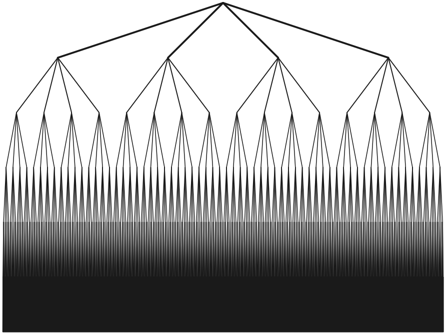
Not accounting for state similarity hinders exploration

Not accounting for state similarity hinders exploration

Sparse gridworld: reward of 0 everywhere



↳ Uniform planning in the space of sequences of actions



`OPD`, budget of $n = 5460$ moves

Concentration

- Does not lead to uniform exploration of the state space

Concentration

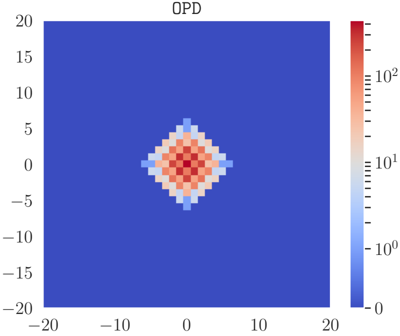
- Does not lead to uniform exploration of the state space
- 2D random walk $\sim$ Rayleigh distribution $P(d) = \frac{2d}{H} e^{-\frac{-d^2}{H}}$

Concentration

- Does not lead to uniform exploration of the state space
- 2D random walk $\sim$ Rayleigh distribution $P(d) = \frac{2d}{H}e^{-\frac{-d^2}{H}}$

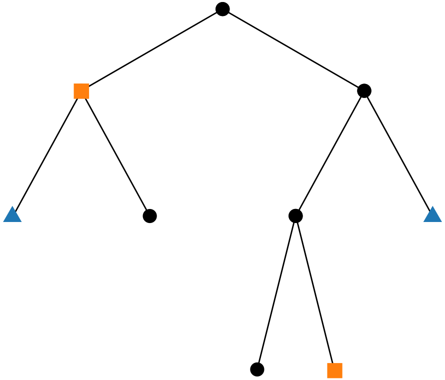
Concentration

- Does not lead to uniform exploration of the state space
- 2D random walk $\sim$ Rayleigh distribution $P(d) = \frac{2d}{H} e^{-\frac{-d^2}{H}}$



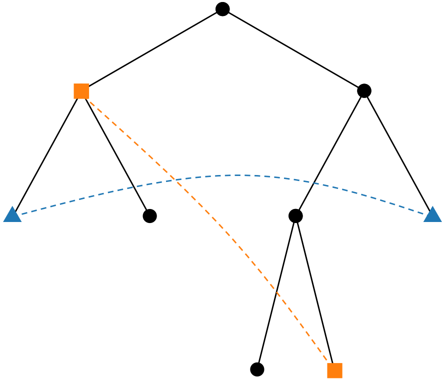
budget of 5460 samples, maximum distance $d = 6$

Better exploit this wasted information



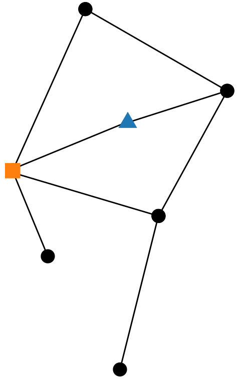
Better exploit this wasted information

- By merging similar states



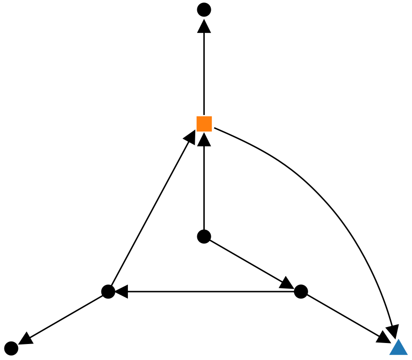
Better exploit this wasted information

- By merging similar states into a graph

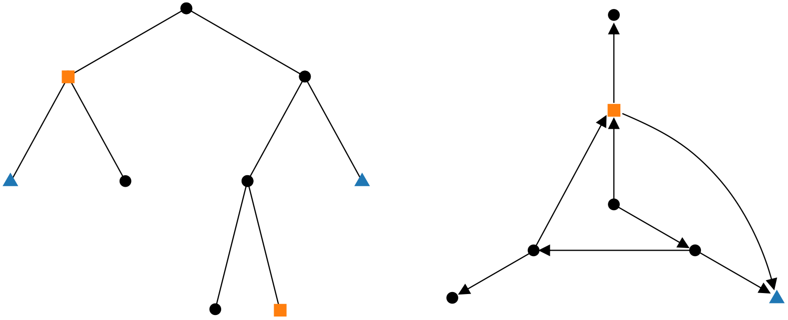


Better exploit this wasted information

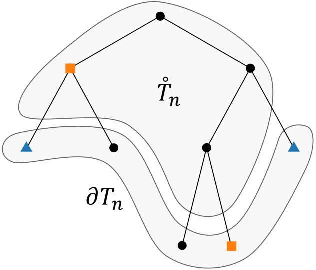
- By merging similar states



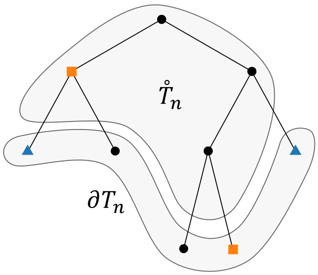
Questions



- How to adapt MCTS algorithms to work on graphs?
- Can we quantify the benefit of using graphs over trees?

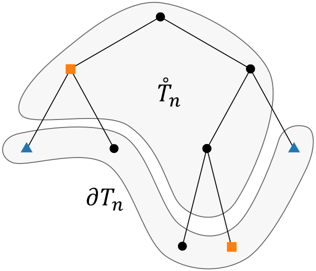


Optimism in the Face of Uncertainty: `OPD`



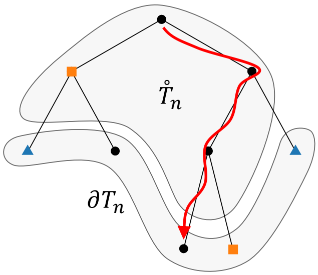
Optimism in the Face of Uncertainty: `OPD`

1. Build confidence bounds $L(a) \leq V(a) \leq U(a)$



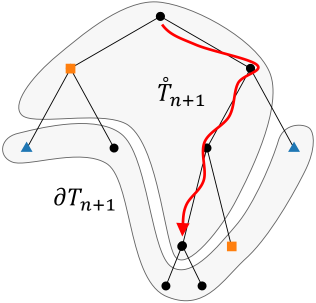
Optimism in the Face of Uncertainty: `OPD`

1. Build confidence bounds $L(a) \leq V(a) \leq U(a)$
2. Follow optimistic actions, from the root down to a leaf b



Optimism in the Face of Uncertainty: OPD

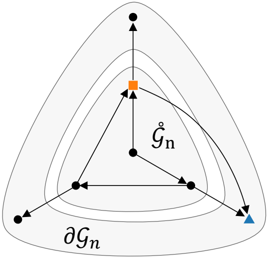
1. Build confidence bounds $L(a) \leq V(a) \leq U(a)$
2. Follow optimistic actions, from the root down to a leaf b
3. Expand the leaf $b \in \partial T_n$



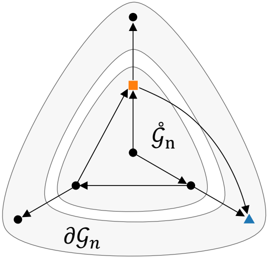
Optimism in the Face of Uncertainty: OPD

1. Build confidence bounds $L(a) \leq V(a) \leq U(a)$
2. Follow optimistic actions, from the root down to a leaf b
3. Expand the leaf $b \in \partial T_n$

On a graph...

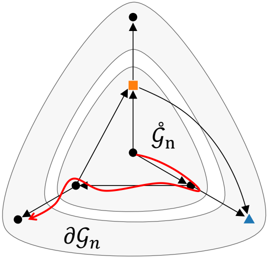


Same principle: `GBOP-D`



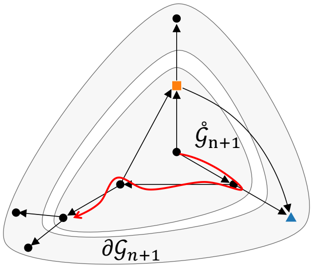
Same principle: `GBOP-D`

1. Build confidence bounds $L(s) \leq V(s) \leq U(s)$



Same principle: GBOP-D

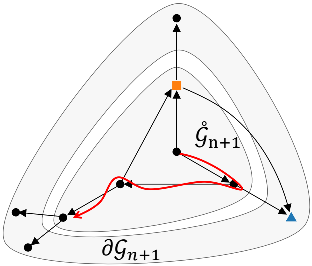
1. Build confidence bounds $L(s) \leq V(s) \leq U(s)$
2. Follow optimistic actions until an external node s is reached



Same principle: `GBOP-D`

1. Build confidence bounds $L(s) \leq V(s) \leq U(s)$
2. Follow optimistic actions until an external node s is reached
3. Expand the external node $s \in \partial\mathcal{G}_n$

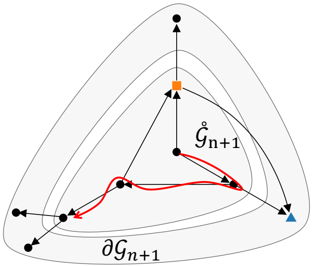
On a graph...



Same principle: `GBOP-D`

1. Build confidence bounds $L(s) \leq V(s) \leq U(s)$
2. Follow optimistic actions until an external node s is reached
3. Expand the external node $s \in \partial\mathcal{G}_n$
   - We are guaranteed to expand any state only once.

On a graph...



Same principle: GBOP-D

1. Build confidence bounds $L(s) \leq V(s) \leq U(s)$
2. Follow optimistic actions until an external node s is reached
3. Expand the external node $s \in \partial\mathcal{G}_n$
   > We are guaranteed to expand any state only once.

How to bound $V(s) = \sup \sum_{t=0}^{\infty} \gamma^t r_t$?

- Initialize with trivial bounds: $L = 0, \quad U = \frac{1}{1-\gamma}$

How to bound $V(s) = \sup \sum_{t=0}^{\infty} \gamma^t r_t$?

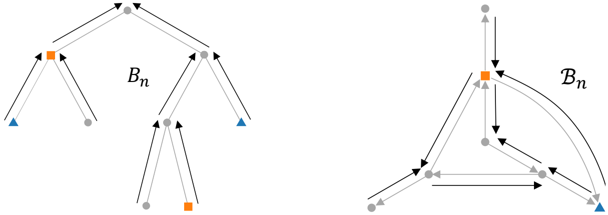
- Initialize with trivial bounds: $L = 0, \quad U = \frac{1}{1-\gamma}$
- Apply the Bellman operator $B : V \to max_a R(s, a) + \gamma V(s')$

$$L(a) \leq B(L)(a) \leq V(a) \leq B(U)(a) \leq U(a)$$

How to bound $V(s) = \sup \sum_{t=0}^{\infty} \gamma^t r_t$?

- Initialize with trivial bounds: $L = 0, \quad U = \frac{1}{1-\gamma}$
- Apply the Bellman operator $B : V \to max_a R(s,a) + \gamma V(s')$

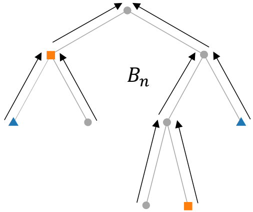
$$L(a) \leq B(L)(a) \leq V(a) \leq B(U)(a) \leq U(a)$$

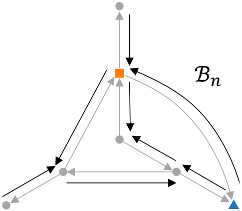


How to bound $V(s) = \sup \sum_{t=0}^{\infty} \gamma^t r_t$?

- Initialize with trivial bounds: $L = 0, \quad U = \frac{1}{1-\gamma}$
- Apply the Bellman operator $B : V \to max_a R(s,a) + \gamma V(s')$

$$L(a) \leq B(L)(a) \leq V(a) \leq B(U)(a) \leq U(a)$$



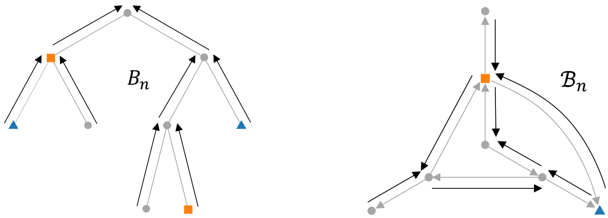


- **Trees**: Converges in d_n steps

How to bound $V(s) = \sup \sum_{t=0}^{\infty} \gamma^t r_t$?

- Initialize with trivial bounds: $L = 0, \quad U = \frac{1}{1-\gamma}$
- Apply the Bellman operator $B : V \to max_a R(s,a) + \gamma V(s')$

$$L(a) \leq B(L)(a) \leq V(a) \leq B(U)(a) \leq U(a)$$



- **Trees**: Converges in d_n steps
- **Graphs**: May converge in ∞ steps when there is a loop

Is `GBOP-D` more efficient than `OPD`?

Is GBOP-D more efficient than OPD?

Performance: $r_n = V^\star - V(a_n)$

Is GBOP-D more efficient than OPD?

Performance: $r_n = V^\star - V(a_n)$

Theorem (Sample complexity of OPD, Hren and Munos, 2008)

$$r_n = \widetilde{\mathcal{O}}\left(n^{-\log\frac{1}{\gamma}/\log\kappa}\right),$$

where κ is a problem-dependent difficulty measure.

Is `GBOP-D` more efficient than `OPD`?

Performance: $r_n = V^\star - V(a_n)$

Theorem (Sample complexity of `OPD`, Hren and Munos, 2008)

$$r_n = \widetilde{\mathcal{O}}\left(n^{-\log\frac{1}{\gamma}/\log\kappa}\right),$$

where κ is a problem-dependent difficulty measure.

Theorem (Sample complexity of `GBOP-D`)

$$r_n = \widetilde{\mathcal{O}}\left(n^{-\log\frac{1}{\gamma}/\log\kappa_\infty}\right),$$

where κ_∞ is a tighter problem-dependent difficulty measure:

$$\kappa_\infty \leq \kappa$$

- $\kappa_\infty = \kappa$ if the MDP has a tree structure

Comparing difficulty measures

- $\kappa_\infty = \kappa$ if the MDP has a tree structure
- $\kappa_\infty < \kappa$ when trajectories intersect a lot

Comparing difficulty measures

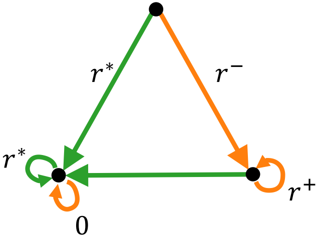
- $\kappa_\infty = \kappa$ if the MDP has a tree structure
- $\kappa_\infty < \kappa$ when trajectories intersect a lot
  - actions cancel each-other out (e.g. moving left or right)

Comparing difficulty measures

- $\kappa_\infty = \kappa$ if the MDP has a tree structure
- $\kappa_\infty < \kappa$ when trajectories intersect a lot
  - actions cancel each-other out (e.g. moving left or right)
  - actions are commutative (e.g. placing pawns on a board)

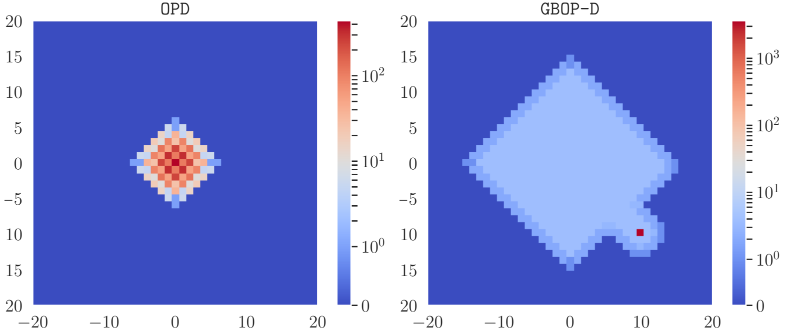
- $\kappa_\infty = \kappa$ if the MDP has a tree structure
- $\kappa_\infty < \kappa$ when trajectories intersect a lot
  - actions cancel each-other out (e.g. moving left or right)
  - actions are commutative (e.g. placing pawns on a board)

Illustrative example: 3 states, $K > 2$ actions



$$\kappa_\infty = 1 < \kappa = K - 1$$

Rewards in a ball around $(10, 10)$ of radius 5, with quadratic decay

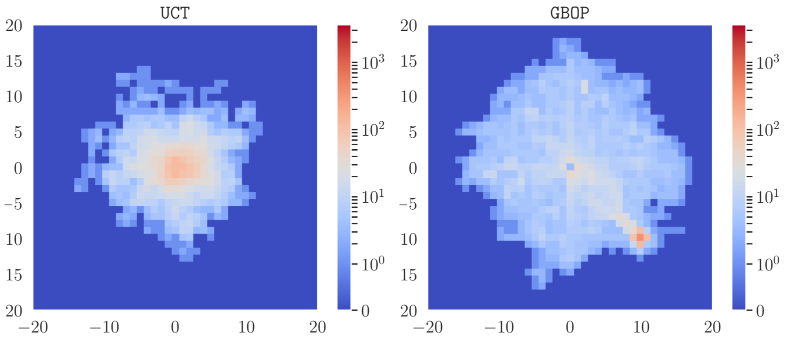


$n = 5640$ samples

Extension to stochastic MDPs

- Use state similarity to tighten the bounds $L \leq V \leq U$.
- We adapt `MDP-GapE` (Jonsson et al., 2020) to obtain `GBOP`

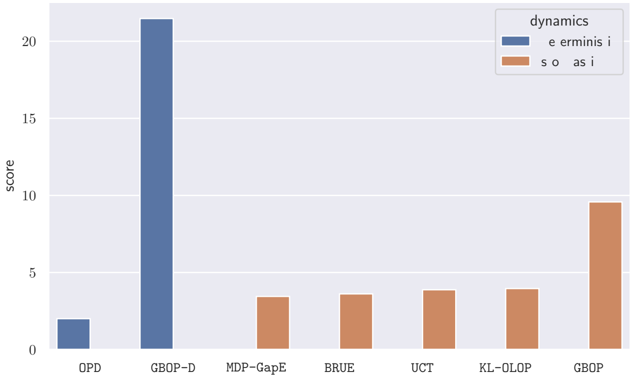
Noisy transitions with probability $p = 10\%$



$n = 5640$ samples

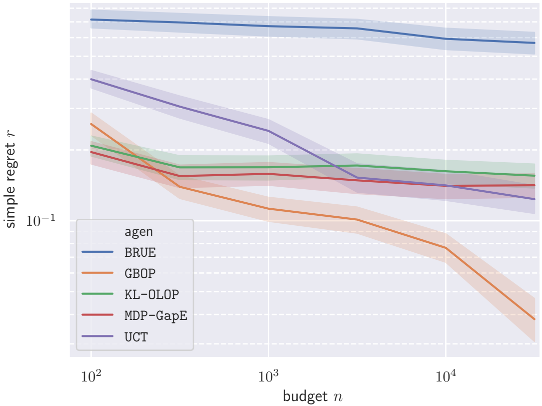
Exploration-Exploitation score

$$S = \sum_{t=1}^{n} \underbrace{d(s_t, s_0)}_{\text{Exploration}} - \underbrace{d(s_t, s_g)}_{\text{Exploitation}}$$

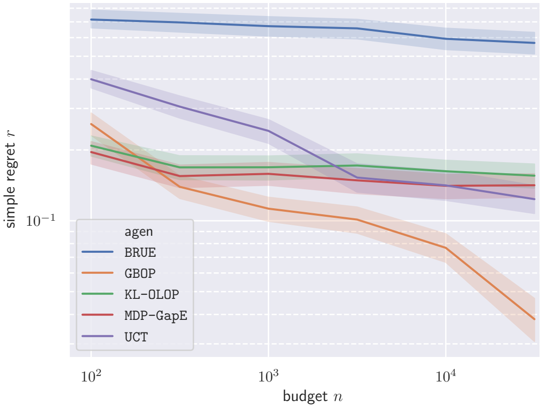


$n = 5640$ samples

Sailing Domain (Vanderbei, 1996)



Sailing Domain (Vanderbei, 1996)



Effective branching factor κ_e:

- $\kappa_e \approx 3.6$, for BRUE, KL-OLOP, MDP-GapE, UCT
- $\kappa_e \approx 1.2$ for GBOP, which suggests our results may still hold