Optimistic algorithms for function optimization

Rémi Munos

SequeL project: Sequential Learning
http://researchers.lille.inria.fr/∼munos/

INRIA Lille - Nord Europe

Workshop: Stochastic optimization and dynamic programming,
Cadarache 2012

Outline of this talk

The "optimism in the face of uncertainty" principle.

- The stochastic multi-armed bandit
- Optimization of a deterministic function
- ... and a noisy function
  - when its "local smoothness" is known,
  - and when it's not.
- Application to planning in MDPs

The stochastic multi-armed bandit problem

Setting:

- Set of K arms, defined by distributions ν_k (with support in $[0, 1]$), whose law is unknown,
- At each time t, choose an arm k_t and receive reward $x_t \overset{i.i.d.}{\sim} \nu_{k_t}$.
- **Goal**: find an arm selection policy such as to maximize the expected sum of rewards.

Exploration-exploitation tradeoff:

- **Explore**: learn about the environment
- **Exploit**: act optimally according to our current beliefs

The regret

Definitions:

- Let $\mu_k = \mathbb{E}[\nu_k]$ be the expected value of arm k,
- Let $\mu^* = \max_k \mu_k$ the best expected value,
- The cumulative expected **regret**:

$$R_n \stackrel{\text{def}}{=} \sum_{t=1}^{n} \mu^* - \mu_{k_t} = \sum_{k=1}^{K} (\mu^* - \mu_k) \sum_{t=1}^{n} \mathbf{1}\{k_t = k\} = \sum_{k=1}^{K} \Delta_k n_k,$$

where $\Delta_k \stackrel{\text{def}}{=} \mu^* - \mu_k$, and n_k the number of times arm k has been pulled up to time n.

Goal: Find an arm selection policy such as to minimize R_n.

Proposed solutions

This is an old problem! [Robbins, 1952] Maybe surprisingly, not fully solved yet!
Many proposed strategies:

- ϵ-**greedy exploration**: choose apparent best action with proba $1-\epsilon$, or random action with proba ϵ,
- **Bayesian exploration**: assign prior to the arm distributions and select arm according to the posterior distributions (Gittins index, Thompson strategy, ...)
- **Softmax exploration**: choose arm k with proba $\propto \exp(\beta \widehat{X}_k)$ (ex: EXP3 algo)
- **Follow the perturbed leader**: choose best perturbed arm
- **Optimistic exploration**: select arm with highest upper bound

The UCB algorithm

Upper Confidence Bound algorithm [Auer, Cesa-Bianchi, Fischer, 2002]: at each time n, select the arm k with highest $B_{k,n_k,n}$ value:

$$B_{k,n_k,n} \stackrel{\text{def}}{=} \underbrace{\frac{1}{n_k}\sum_{s=1}^{n_k} x_{k,s}}_{\widehat{x}_{k,n_k}} + \underbrace{\sqrt{\frac{3\log(n)}{2n_k}}}_{c_{n_k,n}},$$

with:

- n_k is the number of times arm k has been pulled up to time n
- $x_{k,s}$ is the s-th reward received when pulling arm k.

Note that

- Sum of an *exploitation term* and an *exploration term*.
- $c_{n_k,n}$ is a confidence interval term, so $B_{k,n_k,n}$ is a UCB.

Intuition of the UCB algorithm

Idea:

- **"Optimism in the face of uncertainty"** principle
- Select the arm with highest possible mean value, among all possible models that are compatible with the observations.
- The B-values $B_{k,t,n}$ are UCBs on μ_k. Indeed:

$$\mathbb{P}(B_{k,t,n} \geq \mu_k) \quad \geq \quad 1 - \frac{1}{n^3}$$

(and we also have $\mathbb{P}(\widehat{X}_{k,t} - \mu_k \geq \sqrt{\frac{3\log(n)}{2t}}) \leq \frac{1}{n^3}$)

This comes from Chernoff-Hoeffding inequality:

$$\mathbb{P}(\widehat{X}_{k,t} - \mu_k \geq \epsilon) \quad \leq \quad e^{-2t\epsilon^2}$$
$$\mathbb{P}(\widehat{X}_{k,t} - \mu_k \leq -\epsilon) \quad \leq \quad e^{-2t\epsilon^2}$$

Regret bound for UCB

Proposition 1.

Each sub-optimal arm k is visited in average, at most:

$$\mathbb{E}n_k(n) \leq 6\frac{\log n}{\Delta_k^2} + 1 + \frac{\pi^2}{3}$$

times (where $\Delta_k \overset{\text{def}}{=} \mu^* - \mu_k > 0$).

Thus the expected regret is bounded by:

$$\mathbb{E}R_n = \sum_k \mathbb{E}[n_k]\Delta_k \leq 6 \sum_{k:\Delta_k>0} \frac{\log n}{\Delta_k} + K(1 + \frac{\pi^2}{3}).$$

Lower-bounds: [Lai et Robbins, 1985]

$$\mathbb{E}R_n = \Omega\Big(\sum_{k:\Delta_k>0} \frac{\Delta_k}{KL(\nu_k||\nu^*)} \log n \Big)$$

Intuition of the proof

Let k be a sub-optimal arm, and k^* be an optimal arm. At time n, if arm k is selected, this means that

$$\begin{aligned} B_{k,n_k,n} &\geq B_{k^*,n_{k^*},n} \\ \widehat{X}_{k,n_k} + \sqrt{\frac{3\log(n)}{2n_k}} &\geq \widehat{X}_{k^*,n_{k^*}} + \sqrt{\frac{3\log(n)}{2n_{k^*}}} \\ \mu_k + 2\sqrt{\frac{3\log(n)}{2n_k}} &\geq \mu^*, \text{ with high proba} \\ n_k &\leq \frac{6\log(n)}{\Delta_k^2} \end{aligned}$$

Thus, if $n_k > \frac{6\log(n)}{\Delta_k^2}$, then there is only a small probability that arm k be selected.

Proof of Proposition 1

Write $u = \frac{6\log(n)}{\Delta_k^2} + 1$. We have:

$$\begin{aligned} n_k(n) &\leq u + \sum_{t=u+1}^{n} \mathbf{1}\{k_t = k; n_k(t) > u\} \\ &\leq u + \sum_{t=u+1}^{n} \Big[\sum_{s=u+1}^{t} \mathbf{1}\{\hat{X}_{k,s} - \mu_k \geq c_{t,s}\} + \sum_{s=1}^{t} \mathbf{1}\{\hat{X}_{k^*,s^*} - \mu_k \leq -c_{t,s^*}\} \Big] \end{aligned}$$

Now, taking the expectation of both sides,

$$\begin{aligned} \mathbb{E}[n_k(n)] &\leq u + \sum_{t=u+1}^{n} \Big[\sum_{s=u+1}^{t} \mathbb{P}\big(\hat{X}_{k,s} - \mu_k \geq c_{t,s}\big) + \sum_{s=1}^{t} \mathbb{P}\big(\hat{X}_{k^*,s^*} - \mu_k \leq -c_{t,s^*}\big) \Big] \\ &\leq u + \sum_{t=u+1}^{n} \Big[\sum_{s=u+1}^{t} t^{-3} + \sum_{s=1}^{t} t^{-3} \Big] \leq \frac{6\log(n)}{\Delta_k^2} + 1 + \frac{\pi^2}{3} \end{aligned}$$

Variants of UCB

- **UCB with variance estimate:** [Audibert, M., Szepesvári, 2008]:

$$B_{k,n_k,n} \stackrel{\text{def}}{=} \widehat{X}_{k,t} + \sqrt{2\frac{V_{k,n_k}\log(1.2n)}{n_k}} + \frac{3\log(1.2n)}{n_k}.$$

  Then the expected regret is bounded by:

$$\mathbb{E}R_n \leq 10\Big(\sum_{k:\Delta_k>0}\frac{\sigma_k^2}{\Delta_k} + 2\Big)\log(n).$$

- **PAC-UCB** Let $\beta > 0$. W.p. $1-\beta$,

$$R_n \leq 6\log(K\beta^{-1})\sum_{k:\Delta_k>0}\frac{1}{\Delta_k}.$$

- **KL-UCB** [Garivier & Cappé, 2011] and $\mathcal{K}_{\inf}$**-UCB** [Maillard, M., Stoltz, 2011]:

$$\mathbb{E}R_n = \sum_{k:\Delta_k>0}\frac{\Delta_k}{KL(\nu_k||\nu^*)}\log n + o(\log n).$$

The optimization problem

Goal: maximize function $f :\rightarrow \boldsymbol{R}$ given a finite budget n of (noisy or noiseless) evaluations.

Protocol:

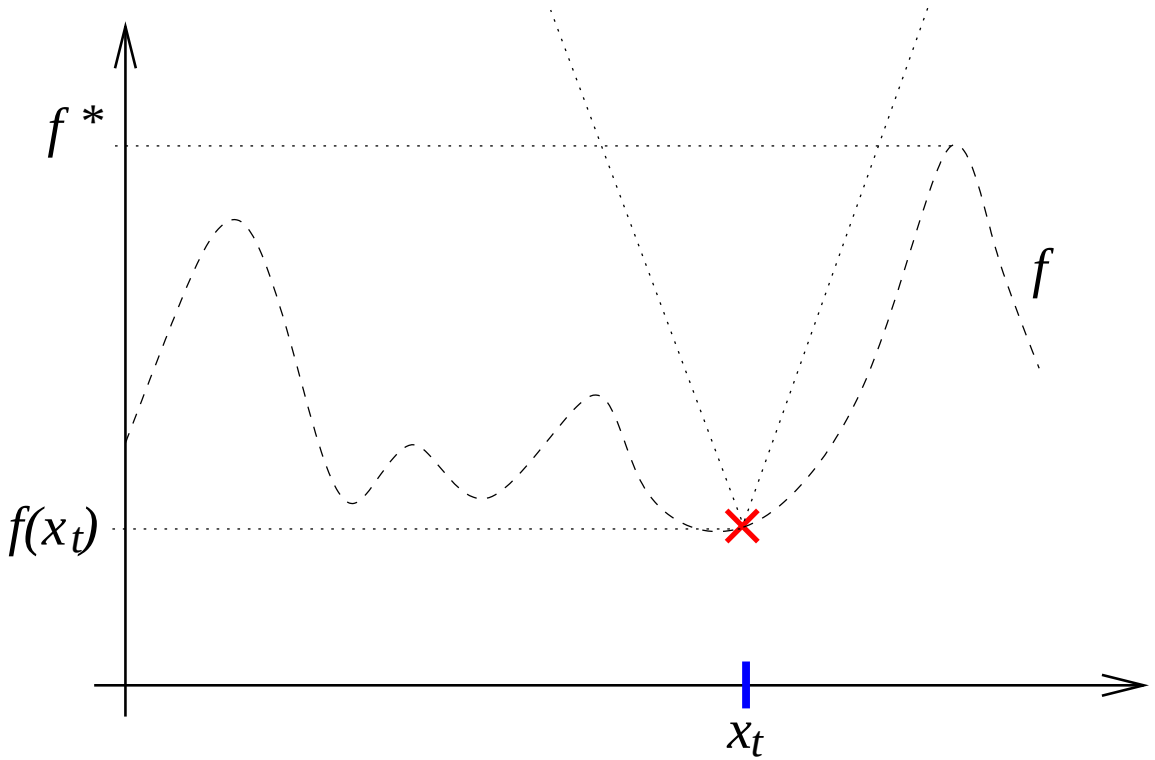
- For $t = 1$ to n, select state $x_t \in$ and observe
  - Deterministic case: $f(x_t)$
  - Stochastic case: $f(x_t) + \epsilon_t$, with $\mathbb{E}[\epsilon_t | x_t] = 0$
- Return a state $x(n)$.

Loss (or simple regret):

$$r_n = \sup_{x \in} f(x) - f(x(n)). \tag{1}$$

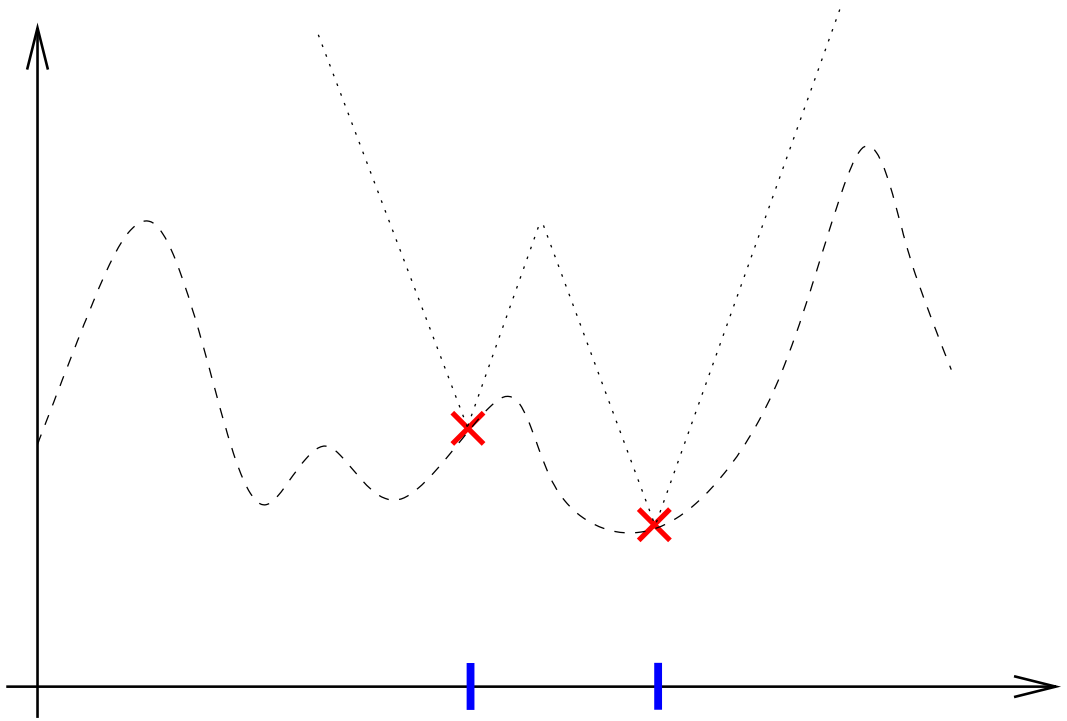
Optimistic optimization: illustration

Assume $f : X \to \mathbf{R}$ is Lipschitz: $|f(x) - f(y)| \leq \ell(x, y)$.



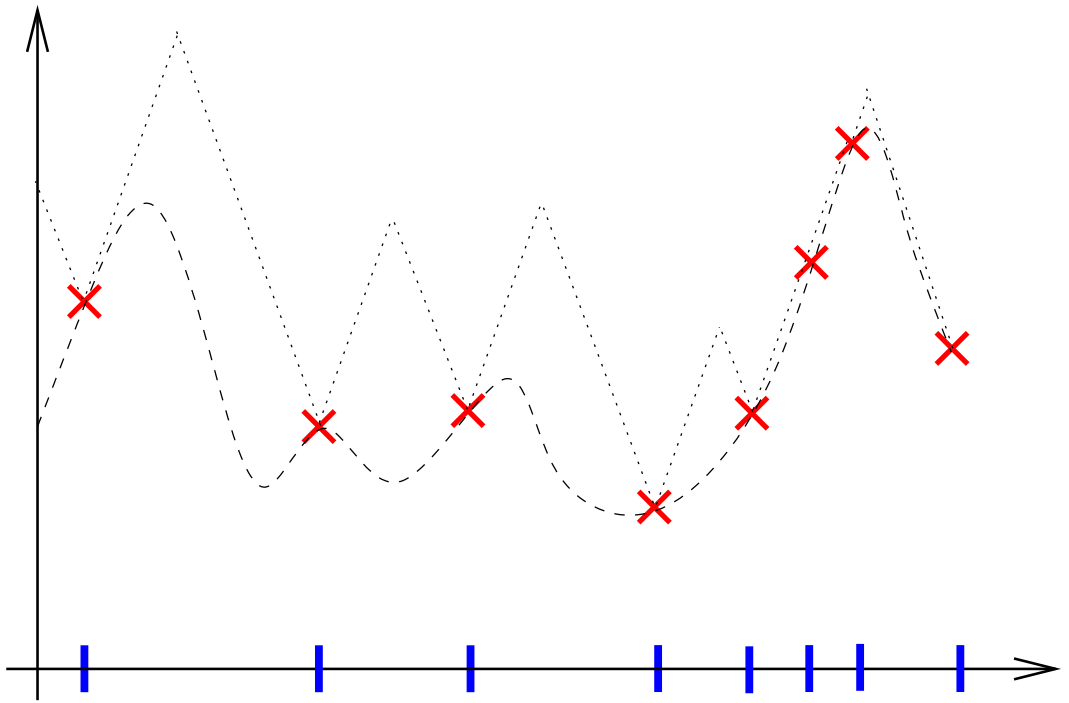
Lipschitz property $\to$ the evaluation of f at x_t provides a first upper-bound on f.

Example in 1d (continued)



New point $\rightarrow$ refined upper-bound on f.

Example in 1d (continued)

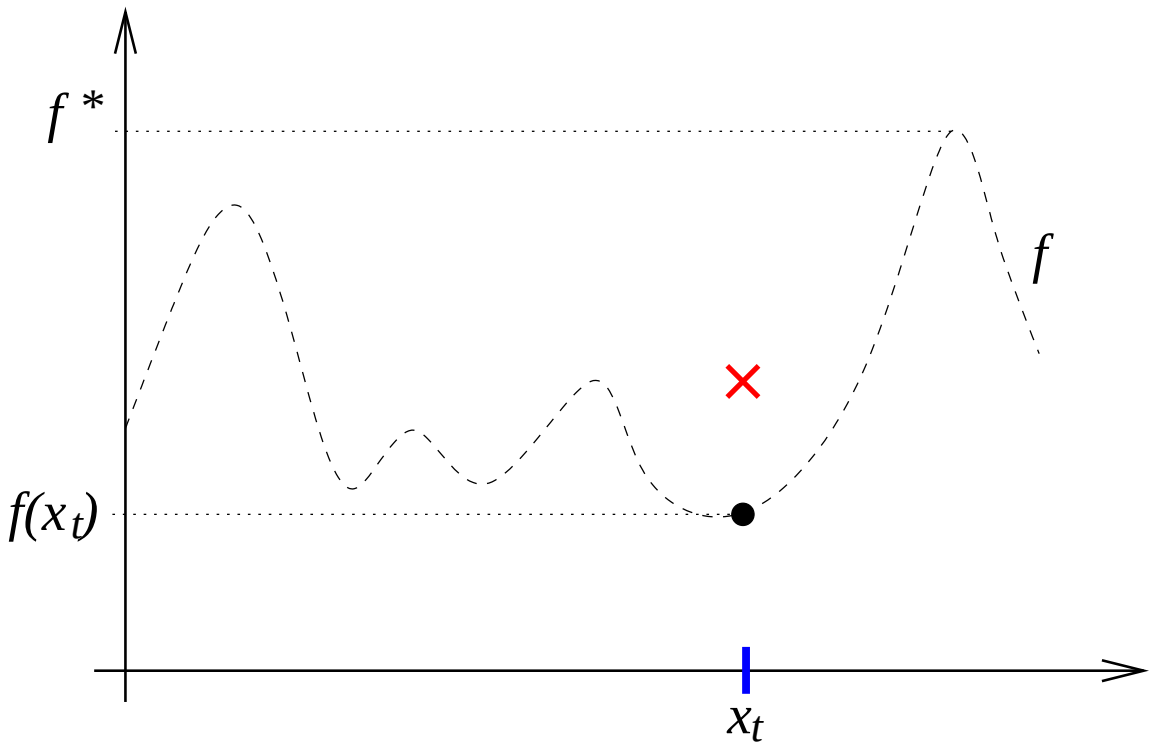


Optimistic optimization: Sample the point with highest upper bound.

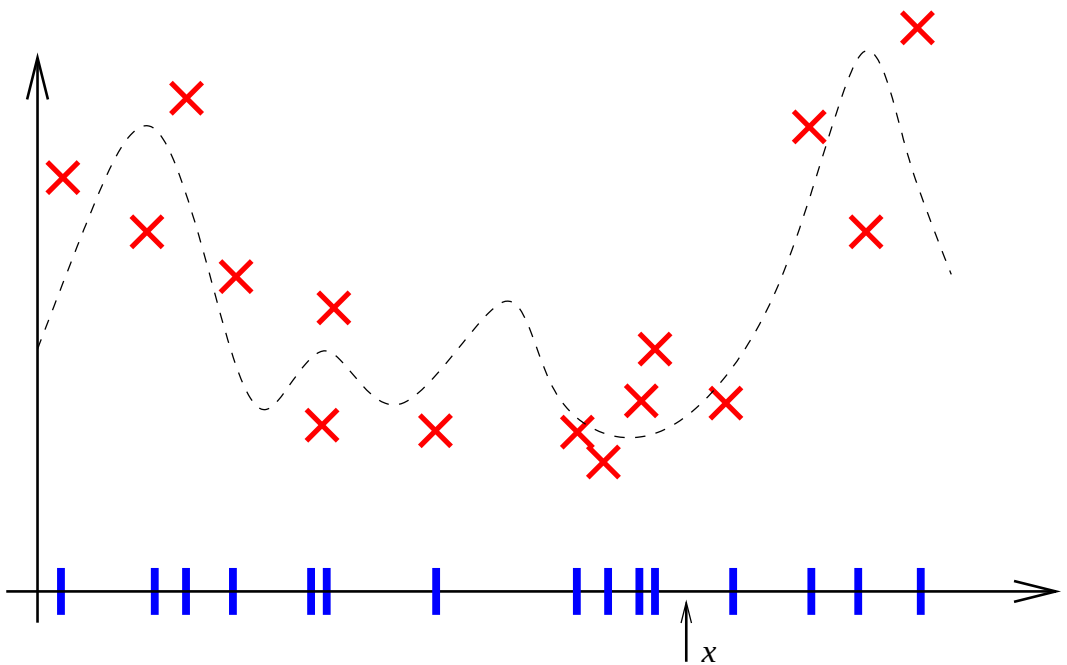
"Optimism in the face of computational uncertainty"

Lipschitz optimization with noisy evaluations

f is still Lipschitz, but now, the evaluation of f at x_t returns a noisy evaluation $r_t = f(x_t) + \epsilon_t$ where $\mathbb{E}[\epsilon_t | x_t] = 0$.

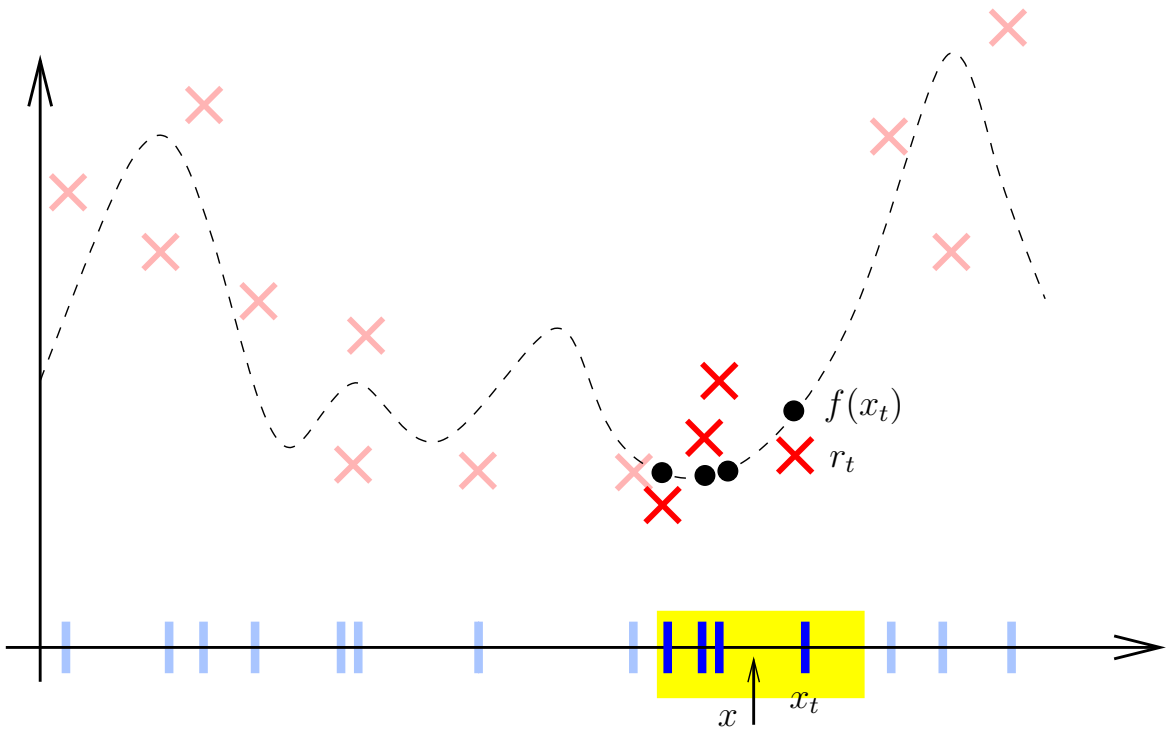


Where should one sample next?



How to define a high probability upper bound at any state x?

UCB in a given domain

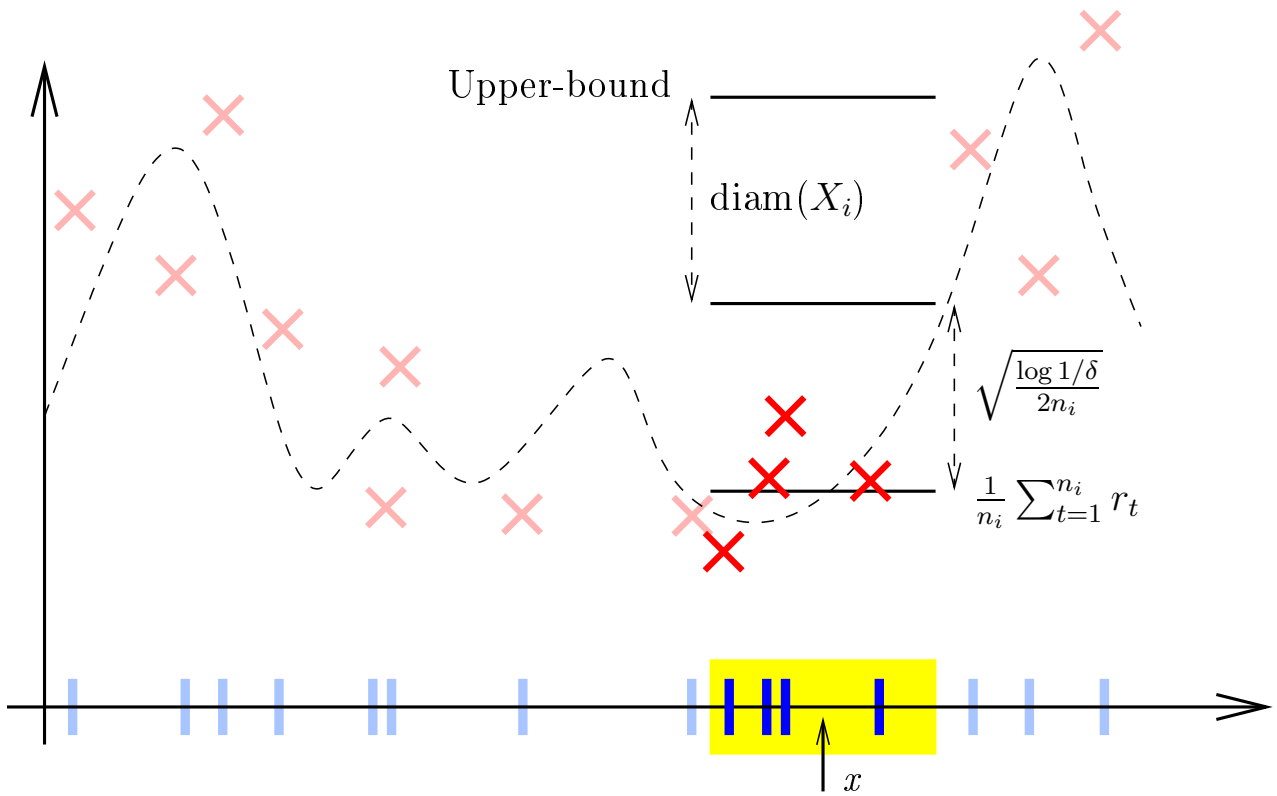


For a fixed domain $X_i \ni x$ containing n_i points $\{x_t\} \in X_i$, we have that $\sum_{t=1}^{n_i} r_t - f(x_t)$ is a Martingale. Thus by Azuma's inequality,

$$\frac{1}{n_i}\sum_{t=1}^{n_i} r_t + \sqrt{\frac{\log 1/\delta}{2n_i}} \geq \frac{1}{n_i}\sum_{t=1}^{n_i} f(x_t) \geq f(x) - diam(X_i),$$

since f is Lipschitz (where $diam(X_i) = \sup_{x,y \in X_i} \ell(x,y)$).

High probability upper bound



w.p. $1-\delta$, $\quad \frac{1}{n_i}\sum_{t=1}^{n_i} r_t + \sqrt{\frac{\log 1/\delta}{2n_i}} + diam(X_i) \geq \sup_{x \in X_i} f(x).$

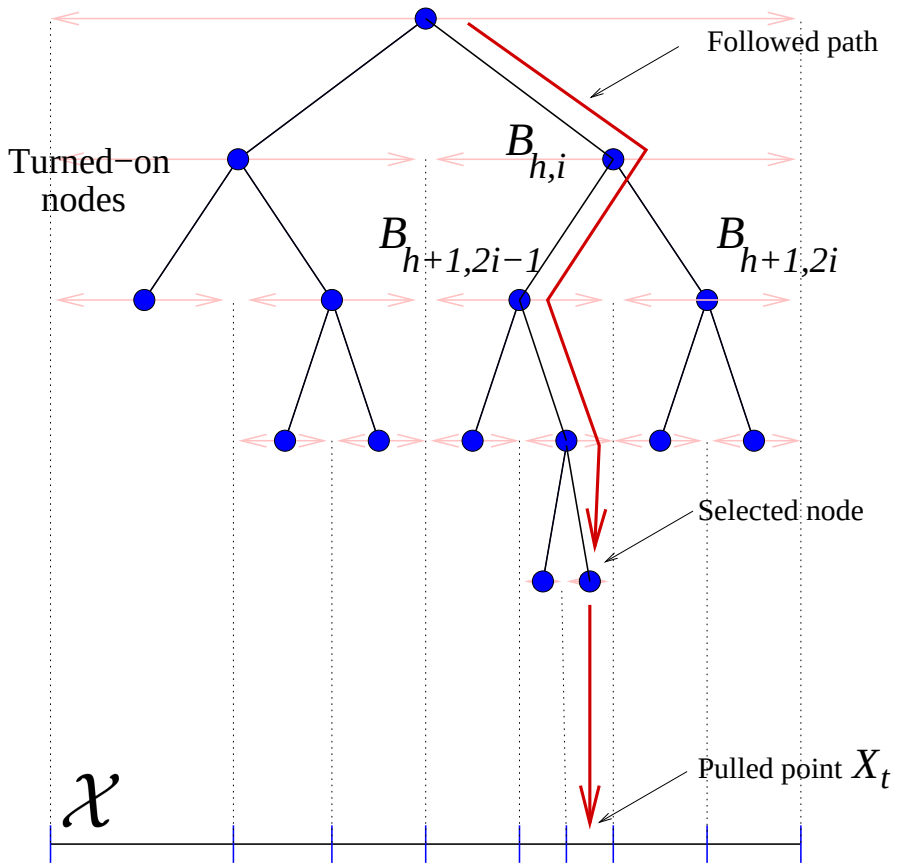
Tradeoff between size of the confidence interval and diameter.

By considering several domains we can derive a tighter upper bound.

Hierarchical Optimistic Optimization (HOO)

[Bubeck, M., Stoltz, Szepesvári, 2008]: Builds incrementally a partitions of X

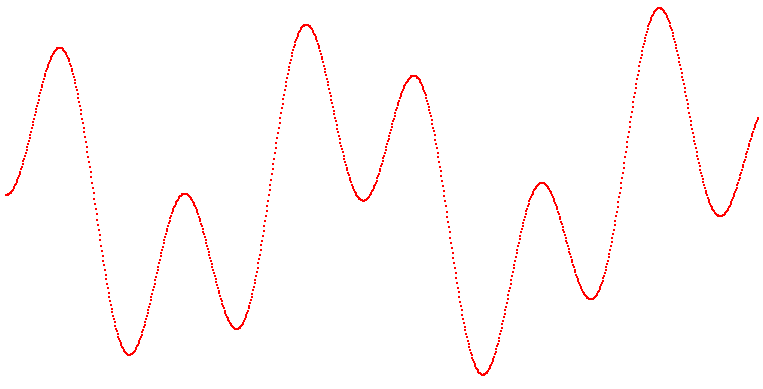
HOO Algorithm:
Let $\mathcal{T}_t$ denote the set of expanded nodes at round t.
- At t, select a leaf i_t of $\mathcal{T}_t$ by maximizing the B-values,
- $\mathcal{T}_{t+1} = \mathcal{T}_t \cup \{i_t\}$
- Select $x_t \in X_{i_t}$ (arbitrarily)
- Observe reward $r_t \sim \nu(x_t)$ and update the B-values:

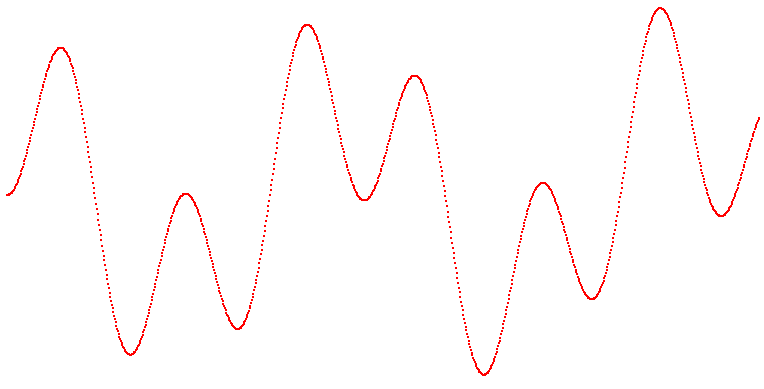


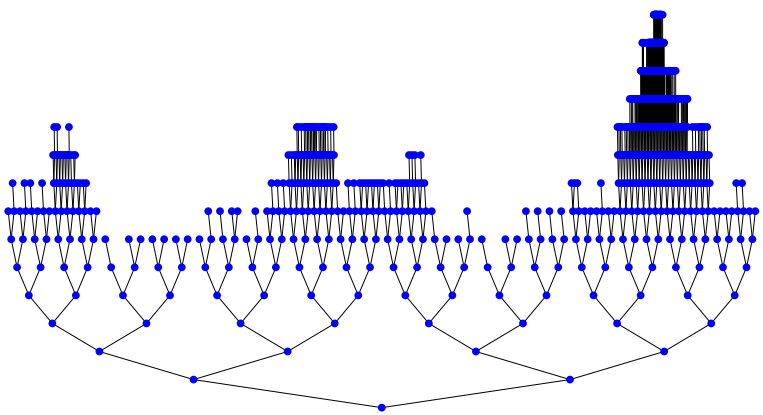
$$B_i \stackrel{\text{def}}{=} \min\left[\widehat{X}_{i,n_i} + \sqrt{\frac{2\log(n)}{n_i}} + diam(i), \max_{j\in\mathcal{C}(i)} B_j\right]$$

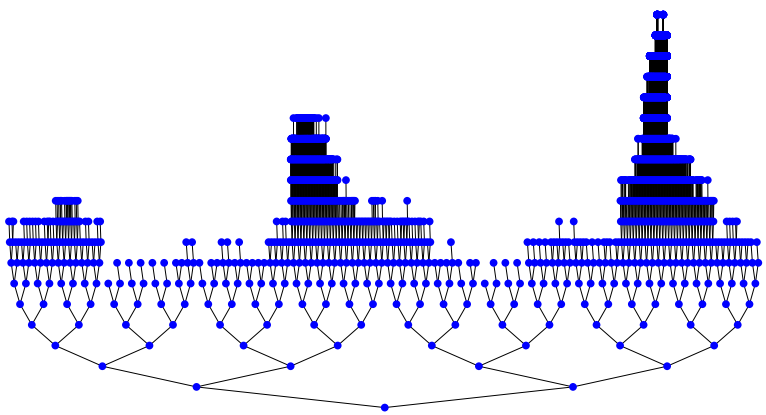
Example in 1d

$r_t \sim \mathcal{B}(f(x_t))$ a Bernoulli distribution with parameter $f(x_t)$







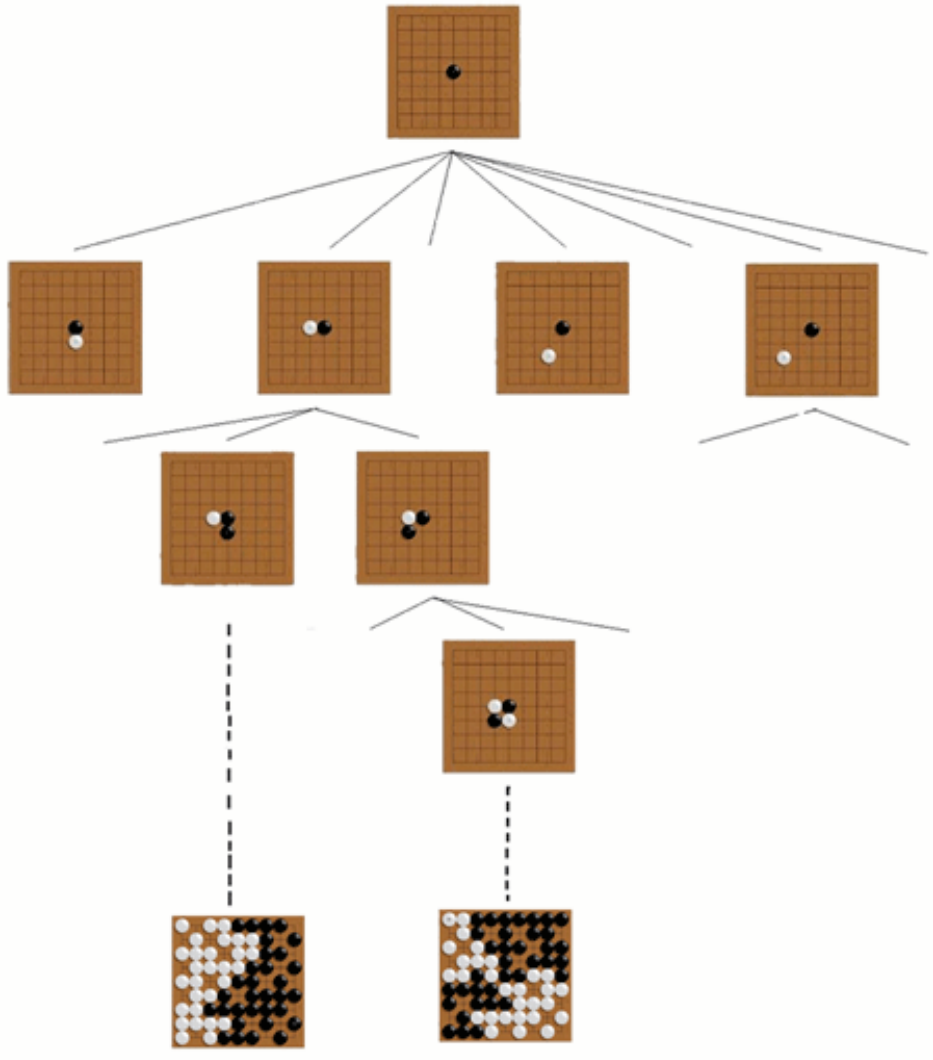


Resulting tree at time $n = 1000$ and at $n = 10000$.

Experiments in computer-go

MoGo program [Gelly, Wang, M., Teytaud, 2006] uses a modified version of HOO called UCT [Kocsis and Szepesvári, 2006].
Features:

- Hierarchical UCB bandits
- Asymmetric tree expansion
- Explore first the most promising branches
- Average converges to max
- Anytime algo
- Use of features
- Generalization among nodes
- Parallelization



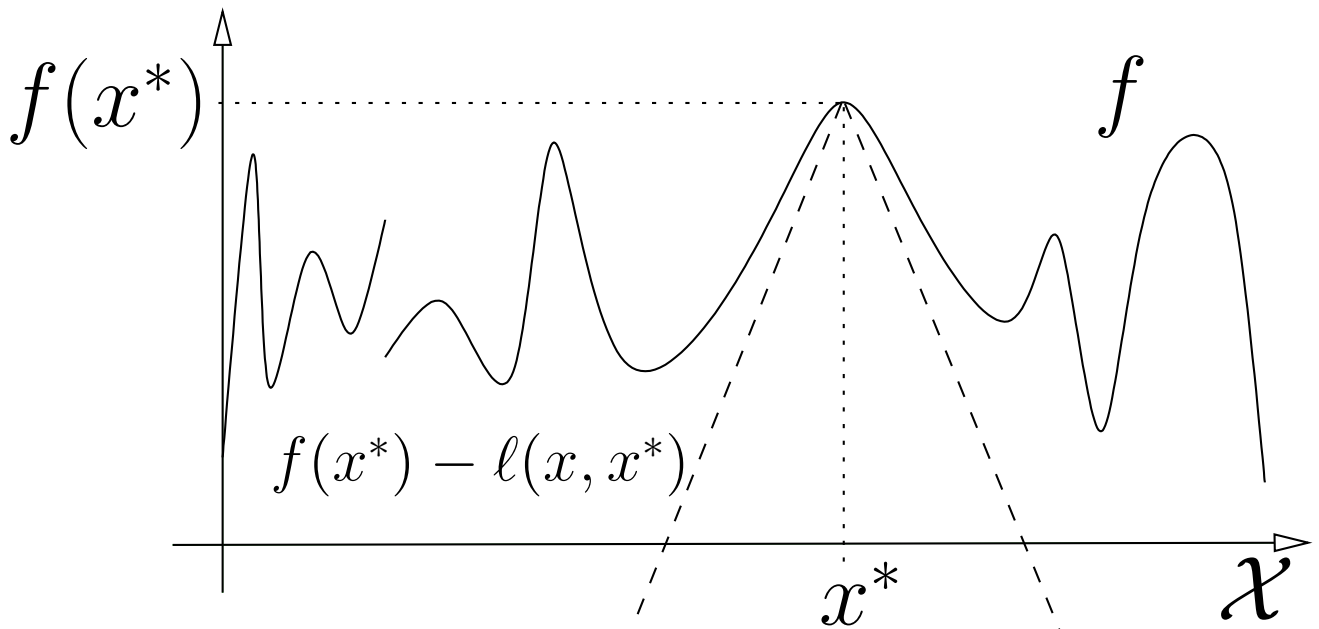
Among world best programs!

General problem

Assumptions:

1. $\ell(x, y)$ is a **semi-metric** ($\ell(x, y) = 0 \Leftrightarrow x = y$ and symmetric)
2. f is **locally "smooth" around its max.**: for all $x \in \mathcal{X}$,

$$f(x^*) - f(x) \leq \ell(x, x^*).$$



Analysis of HOO

Let d be the **near-optimality dimension** of f in X: i.e. such that the set of ε-optimal states

$$X_\varepsilon \stackrel{\text{def}}{=} \{x \in X, f(x) \geq f^* - \varepsilon\}$$

can be covered by $O(\varepsilon^{-d})$ balls of radius ε.

Theorem 1.

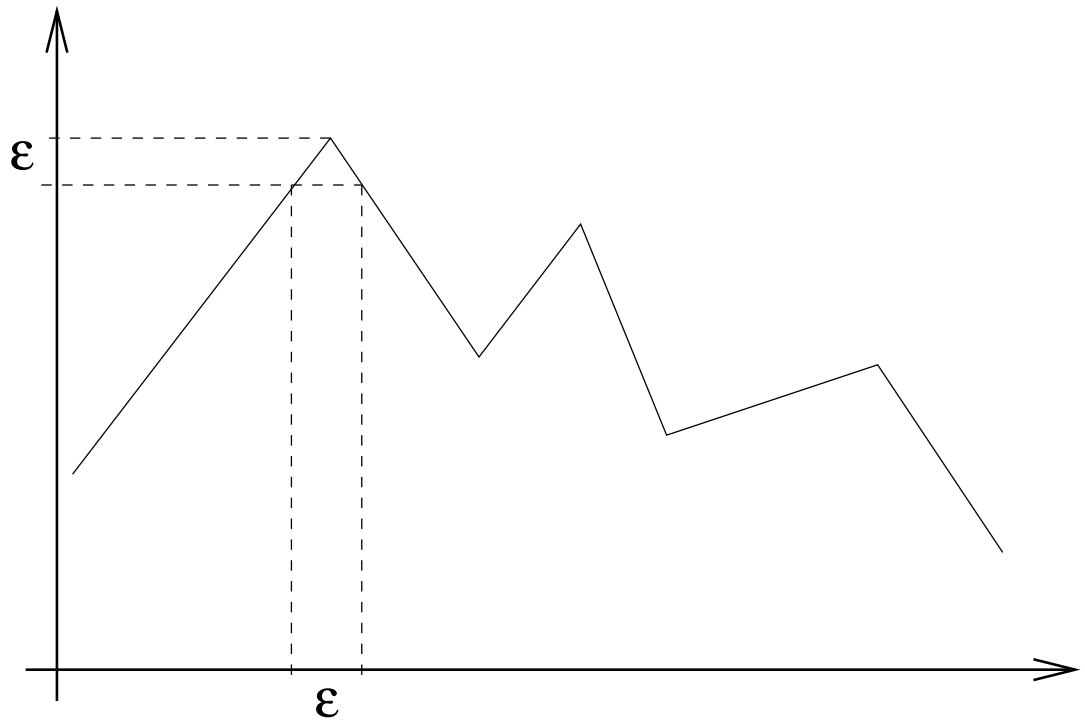
The loss of HOO is

- *In the stochastic case:* $\mathbb{E}r_n = \widetilde{O}(n^{-\frac{1}{d+2}})$.
- *In the deterministic case:* $r_n = \widetilde{O}(n^{-\frac{1}{d}})$ *for* $d > 0$, *and* $r_n = O(e^{-\frac{c}{D}n})$ *for* $d = 0$.

Example 1:

Assume the function is locally peaky around its maximum:

$$f(x^*) - f(x) = \Theta(||x^* - x||).$$

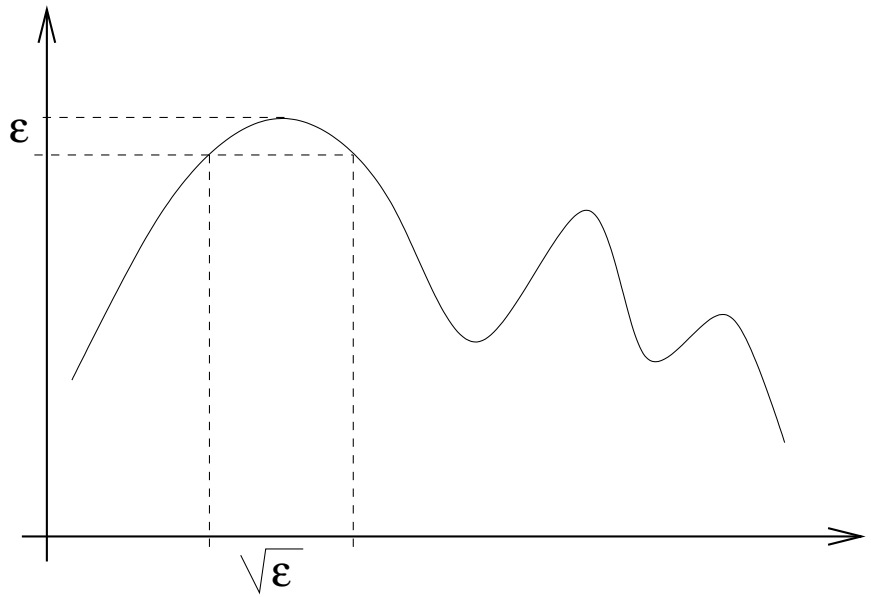


It takes $O(\epsilon^0)$ balls of radius ϵ to cover X_ε. Thus $d = 0$ and the regret is $1/\sqrt{n}$.

Example 2:

Assume the function is locally quadratic around its maximum:

$$f(x^*) - f(x) = \Theta(||x^* - x||^\alpha), \text{ with } \alpha = 2.$$



- For $\ell(x, y) = ||x - y||$, it takes $O(\epsilon^{-D/2})$ balls of radius ϵ to cover X_ε (of size $O(\epsilon^{D/2})$). Thus $d = D/2$.
- For $\ell(x, y) = ||x - y||^2$, it takes $O(\epsilon^0)$ ℓ-balls of radius ϵ to cover X_ε. Thus $d = 0$ and the regret is $1/\sqrt{n}$.

Known smoothness around the maximum

Consider $X = [0,1]^d$. Assume that f has a finite number of global maxima and is locally α-smooth around each maximum x^*, i.e.

$$f(x^*) - f(x) = \Theta(||x^* - x||^\alpha).$$

Then, by choosing $\ell(x,y) = ||x-y||^\alpha$, X_ε is covered by $O(1)$ balls of "radius" ε. Thus the near-optimality dimension $d = 0$, and the regret of HOO is:

$$\mathbb{E}R_n = \widetilde{O}(1/\sqrt{n}),$$

i.e. the rate of growth is **independent of the dimension**.

Discussion about smoothness

The near-optimality dimension may be seen as an excess order of smoothness of f (around its maxima) compared to what is known:

- **If the smoothness order of the function is known** then the regret of HOO algorithm is $\widetilde{O}(1/\sqrt{n})$
- **If the smoothness is underestimated**, for example f is α-smooth but we only use $\ell(x, y) = ||x - y||^{\beta}$, with $\beta < \alpha$, then the near-optimality dimension is $d = D(1/\beta - 1/\alpha)$ and the regret is $\widetilde{O}(n^{-1/(d+2)})$
- **If the smoothness is overestimated**, the weak-Lipschitz assumption is violated, thus there is no guarantee (e.g., UCT)

Assume that ℓ is unknown

f is locally smooth w.r.t. the semi-metric ℓ but now ℓ **is unknown!**

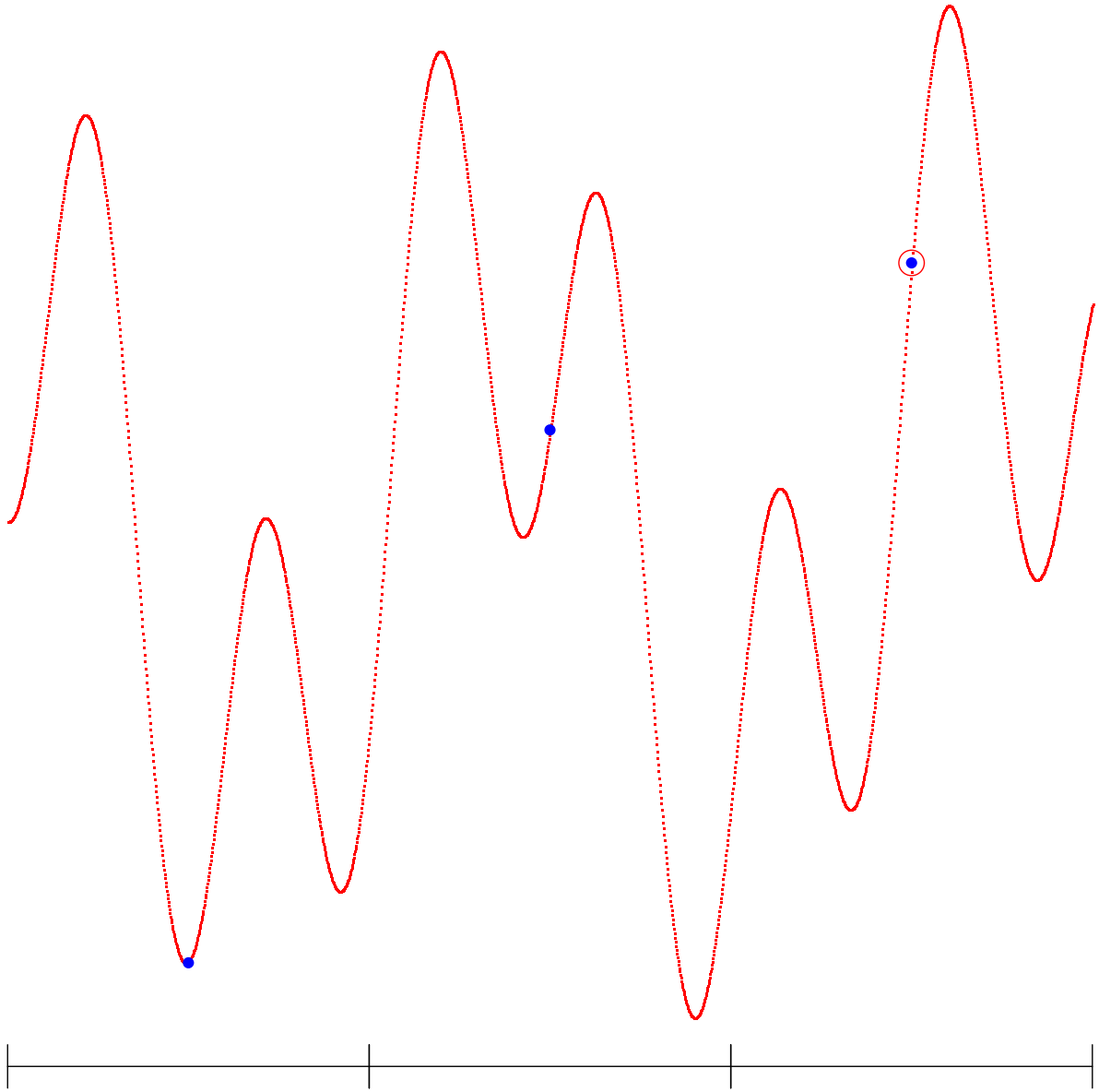
Is it possible to implement an optimistic algorithm with performance guarantees?

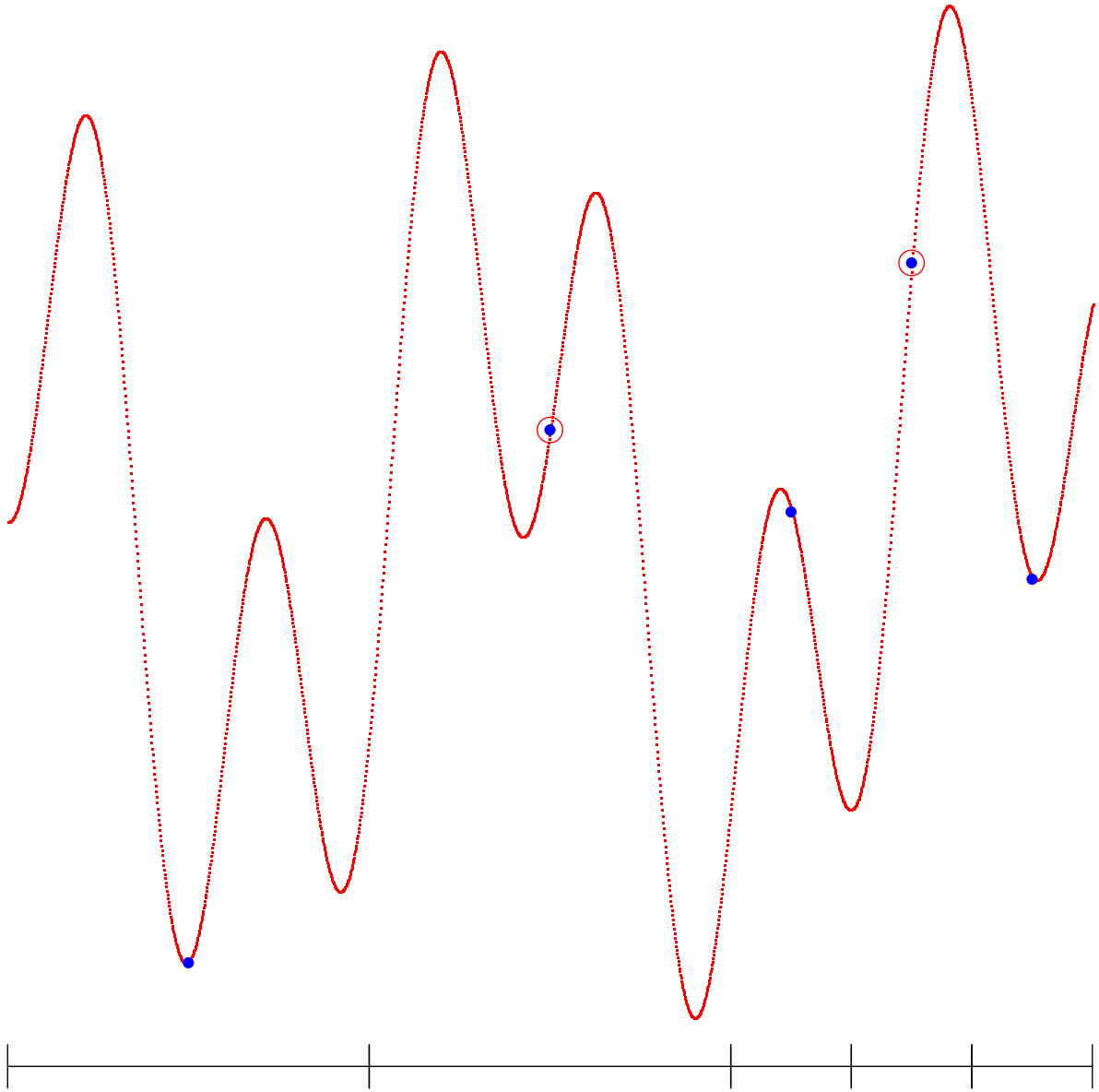
Simultaneous Optimistic Optimization (SOO) [M., 2011]

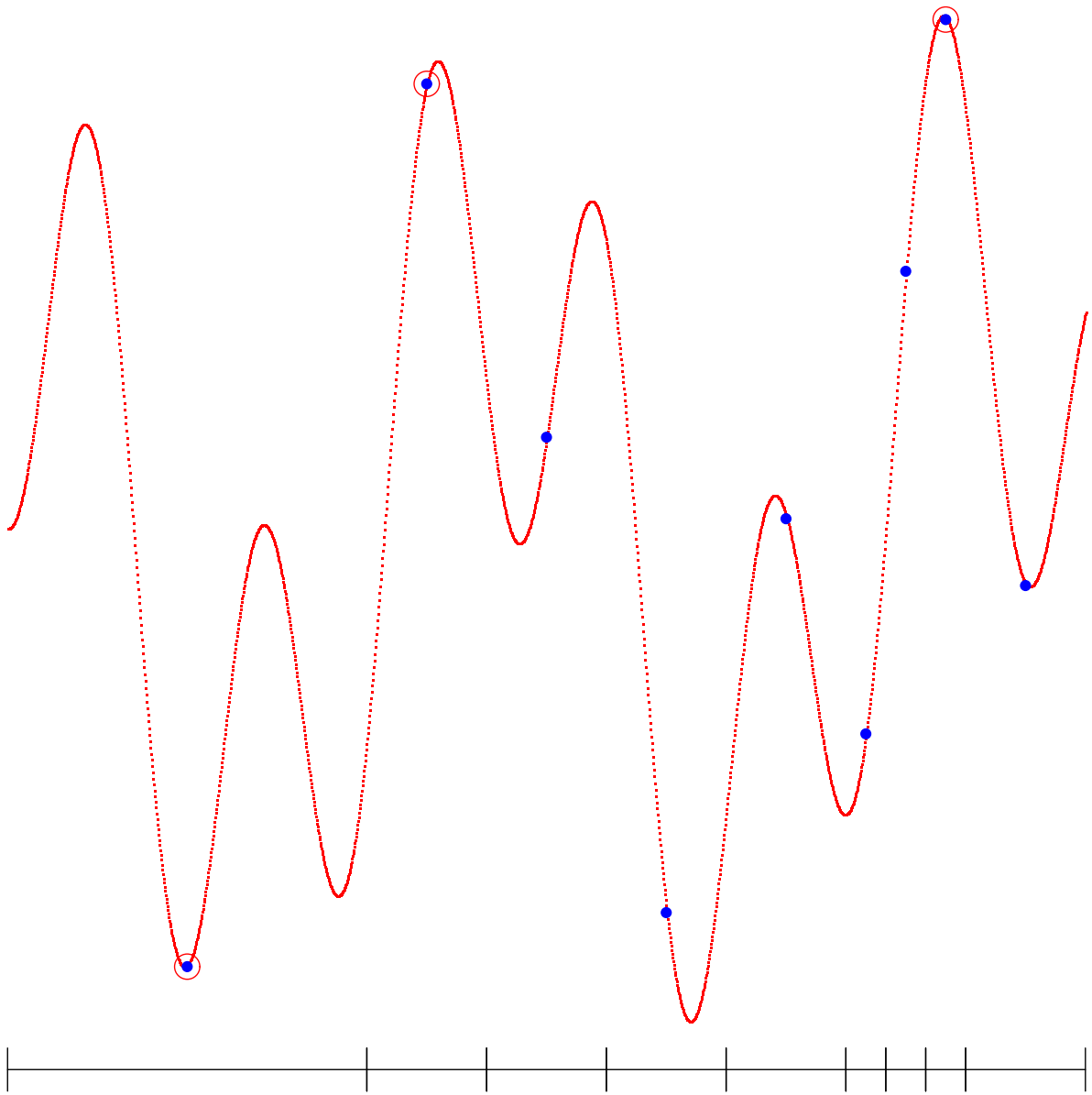
- Expand several leaves simultaneously!
- SOO expands at most one leaf per depth
- SOO expands a leaf only if it has the largest value among all leaves of same or lower depths.
- At round t, SOO does not expand leaves with depth larger than $h_{\max}(t)$

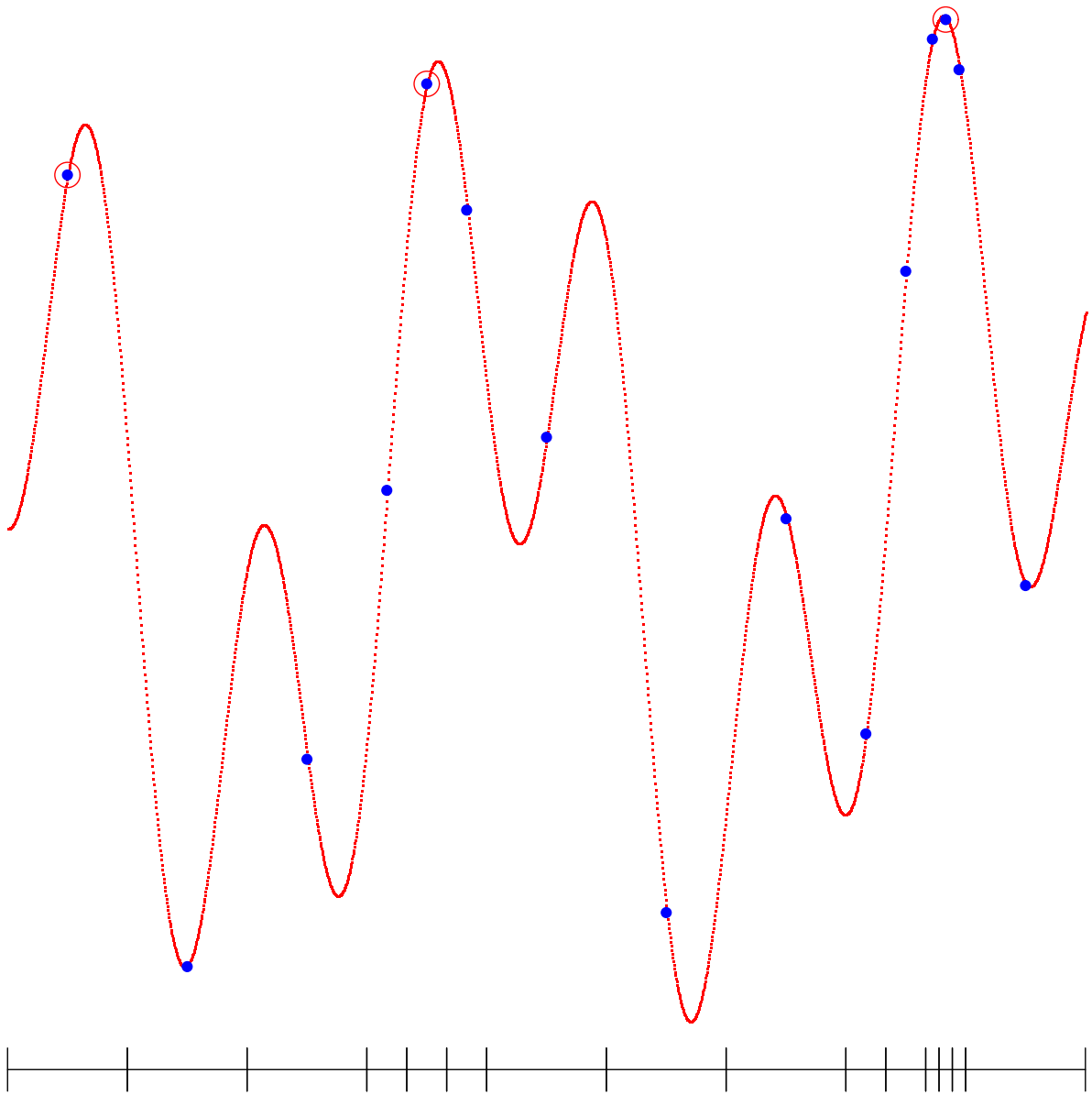
SOO algorithm

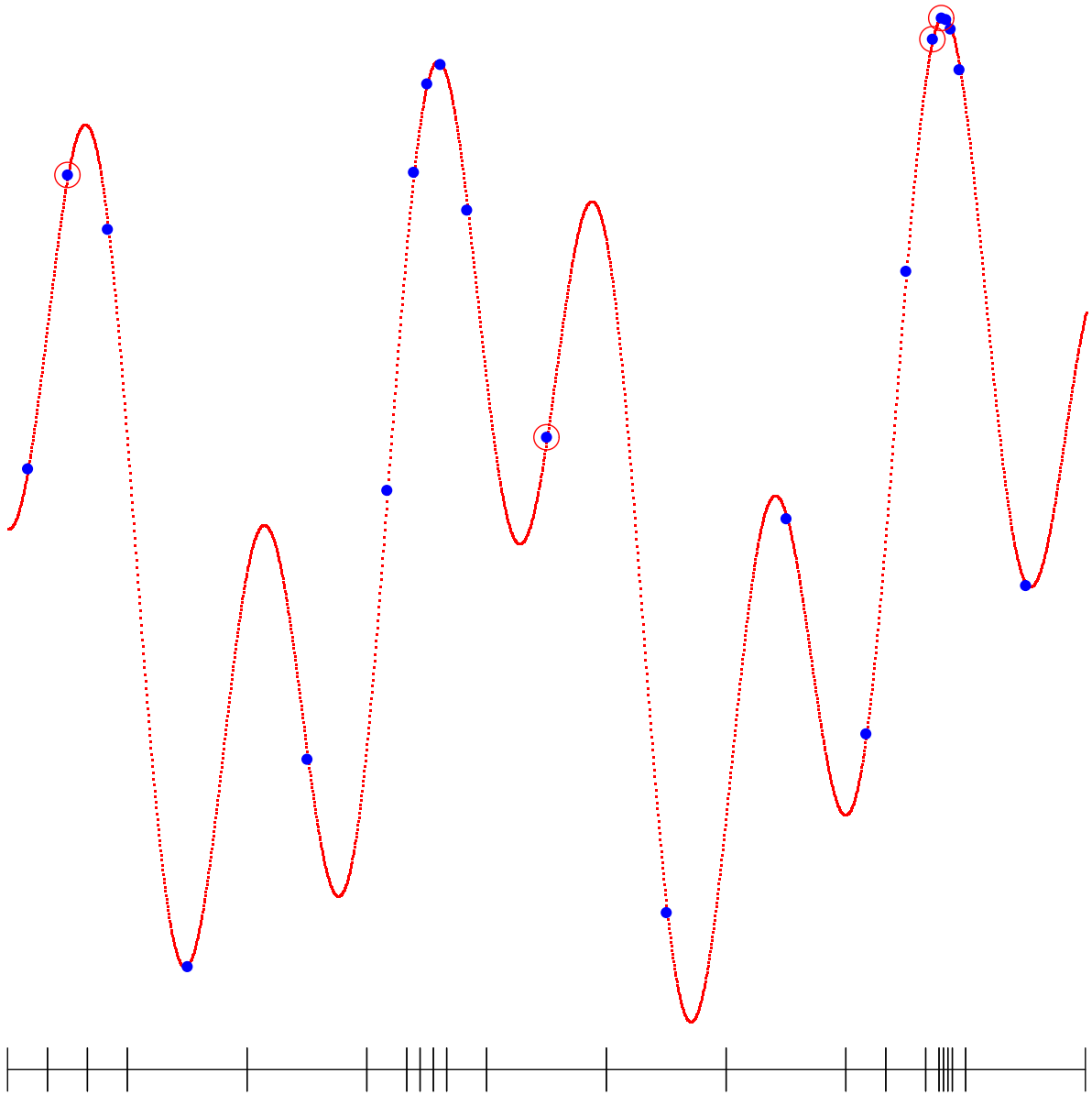
Input: the maximum depth function $t \mapsto h_{\max}(t)$
Initialization: $\mathcal{T}_1 = \{(0,0)\}$ (root node). Set $t = 1$.
while True **do**
  Set $v_{\max} = -\infty$.
  for $h = 0$ to $\min(\text{depth}(\mathcal{T}_t), h_{\max}(t))$ **do**
    Select the leaf $(h,j) \in \mathcal{L}_t$ of depth h with max $f(x_{h,j})$ value
    if $f(x_{h,i}) > v_{\max}$ **then**
      Expand the node (h,i), Set $v_{\max} = f(x_{h,i})$, Set $t = t+1$
      if $t = n$ **then** return $x(n) = \arg\max_{(h,i)\in\mathcal{T}_n} x_{h,i}$
    end if
  end for
end while.

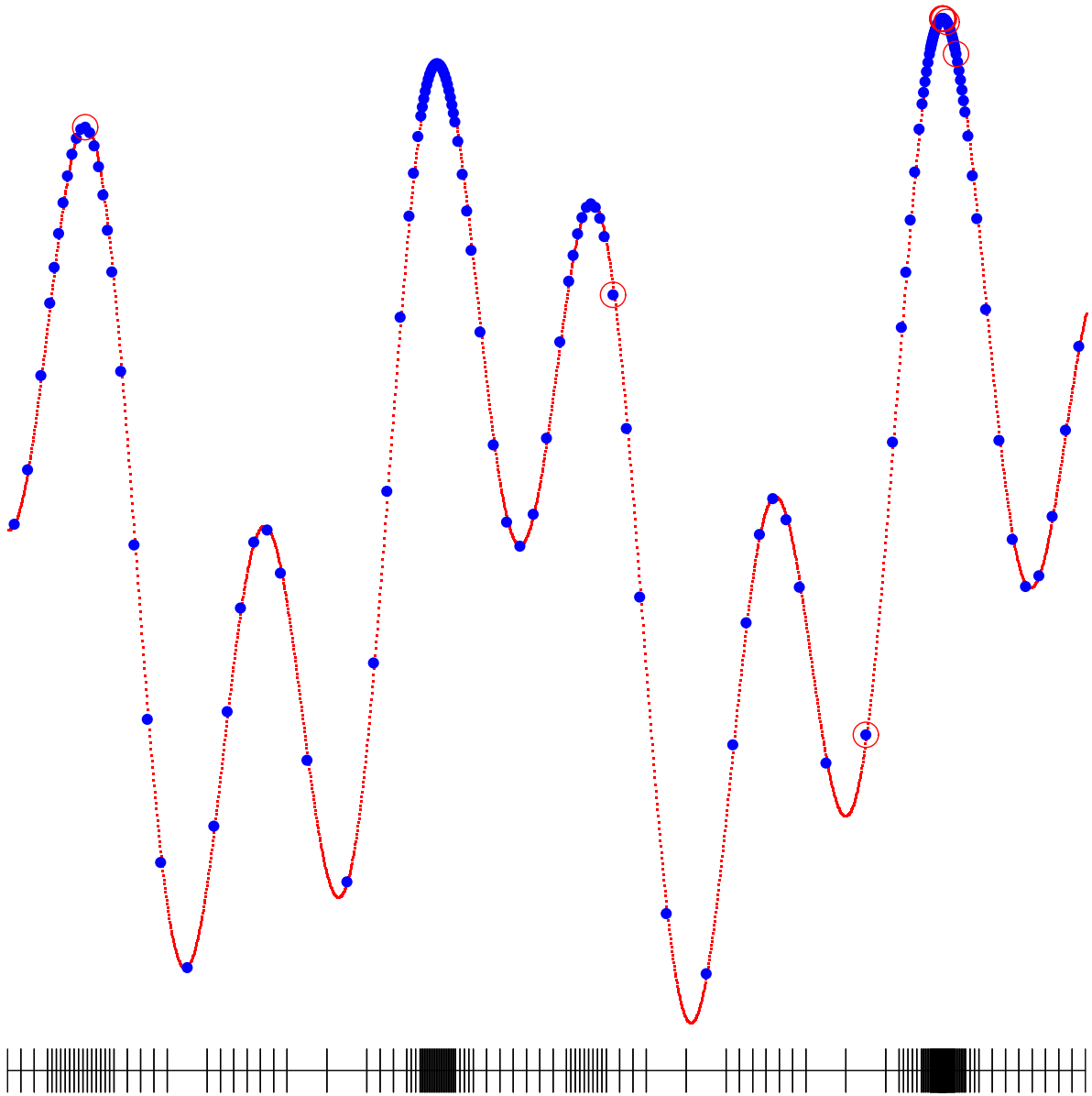


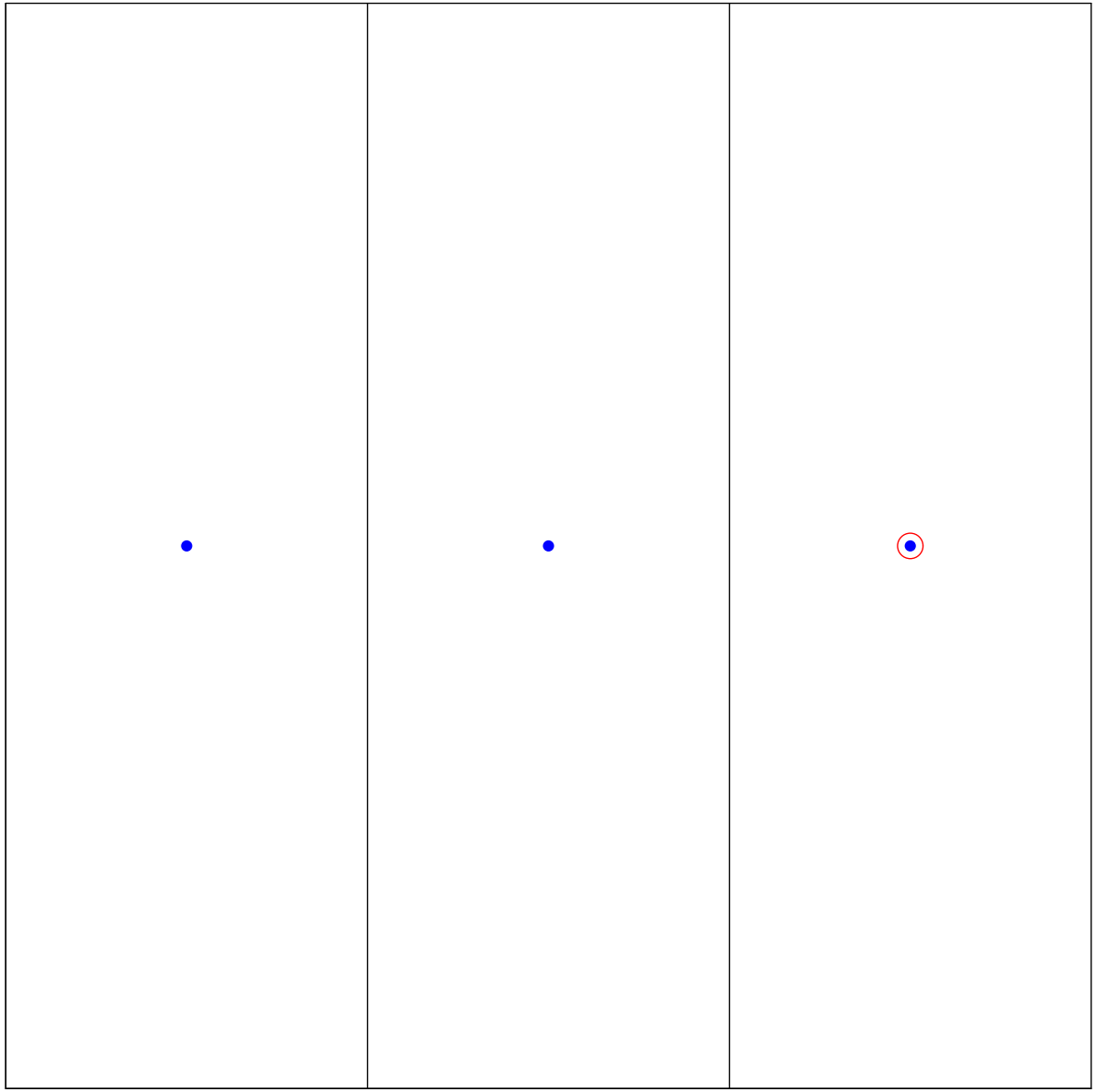


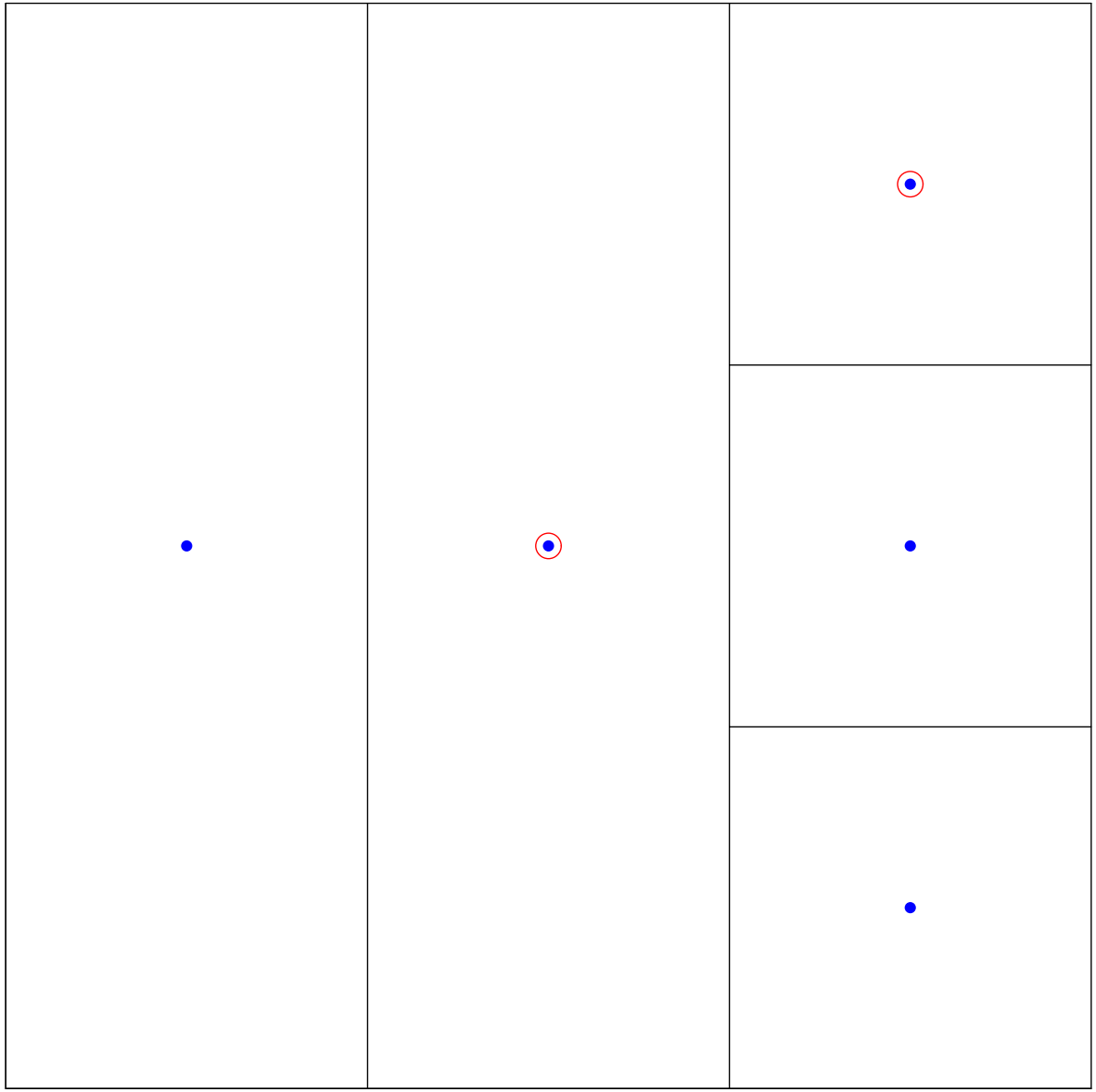


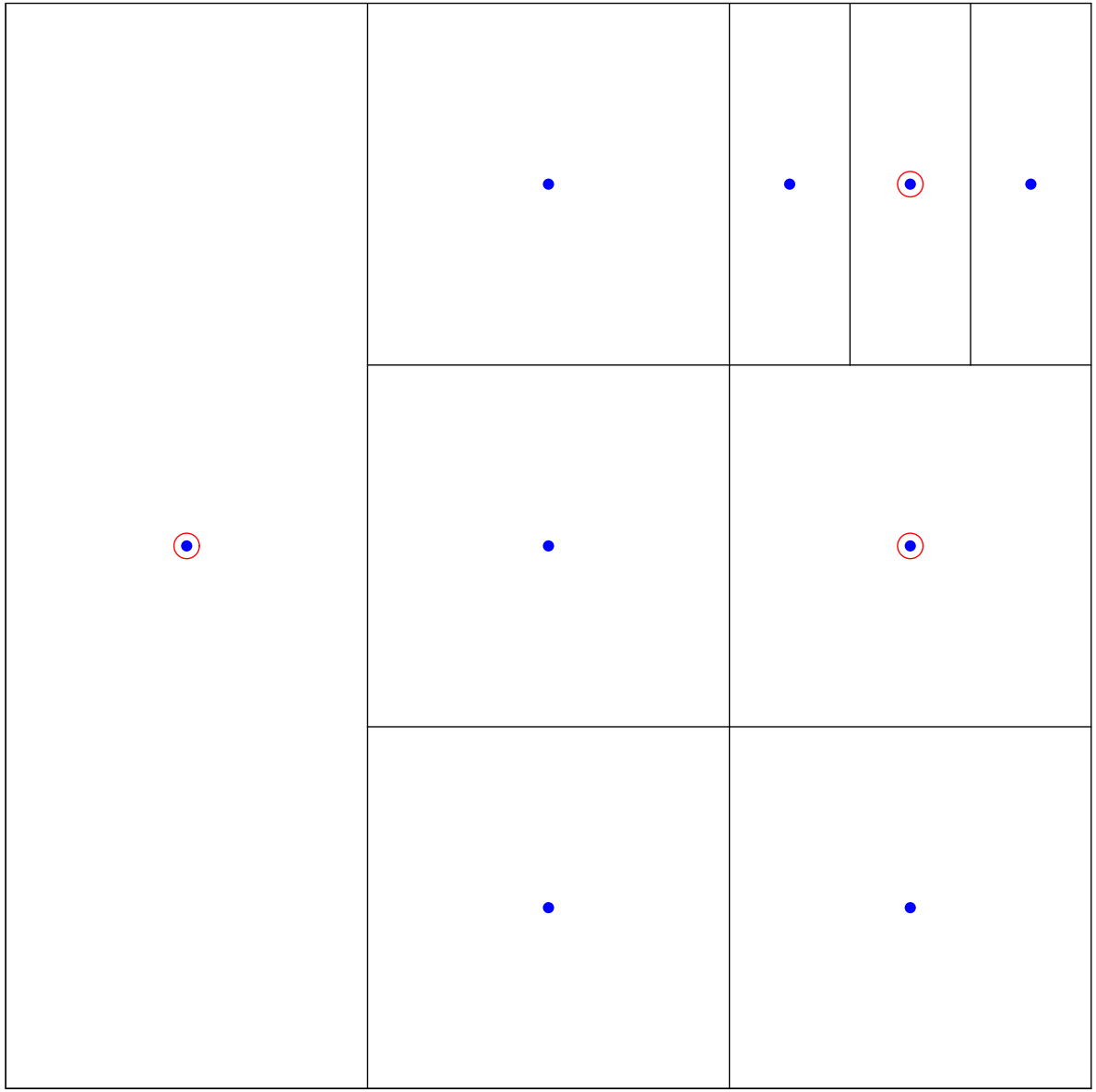


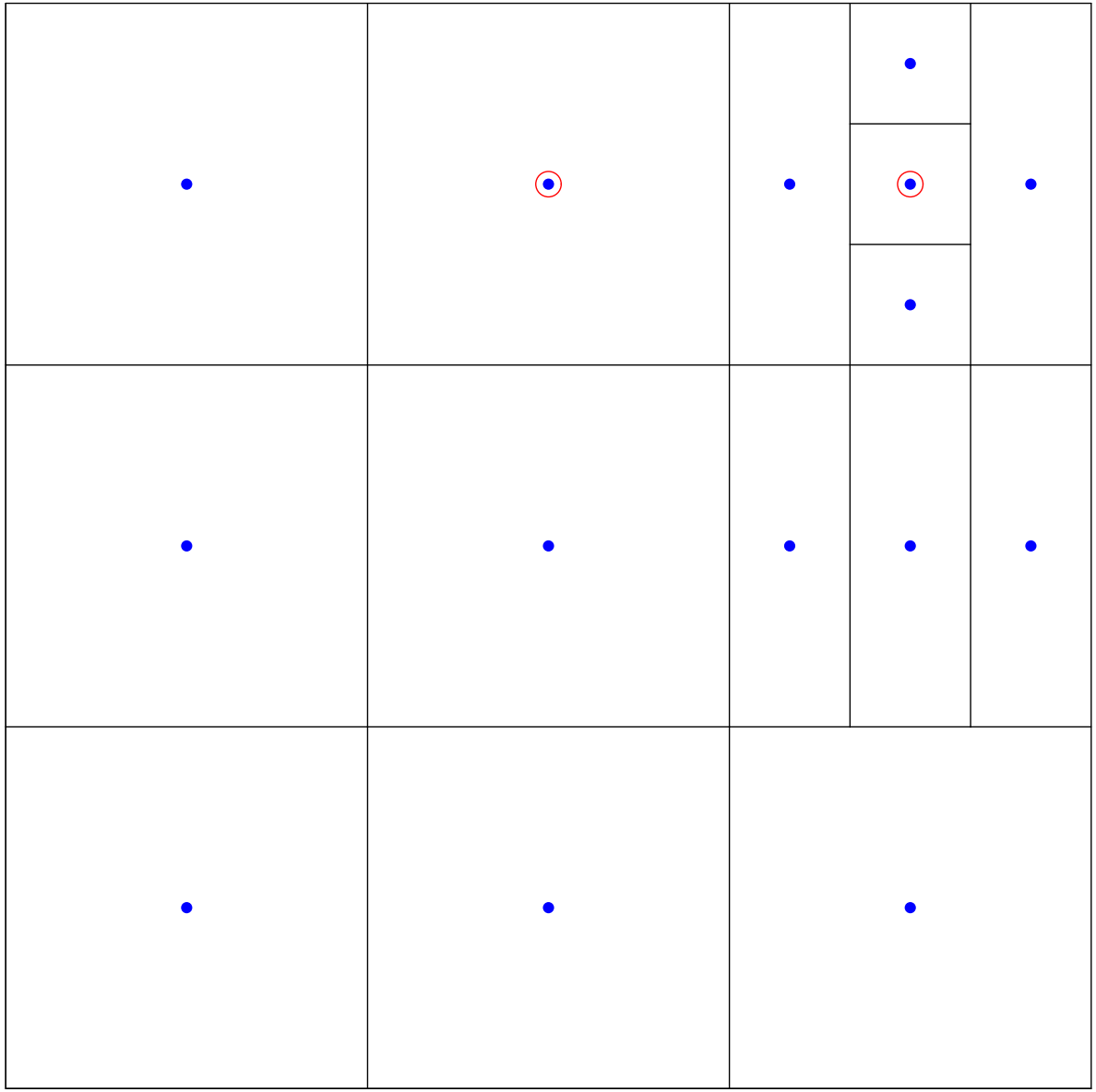


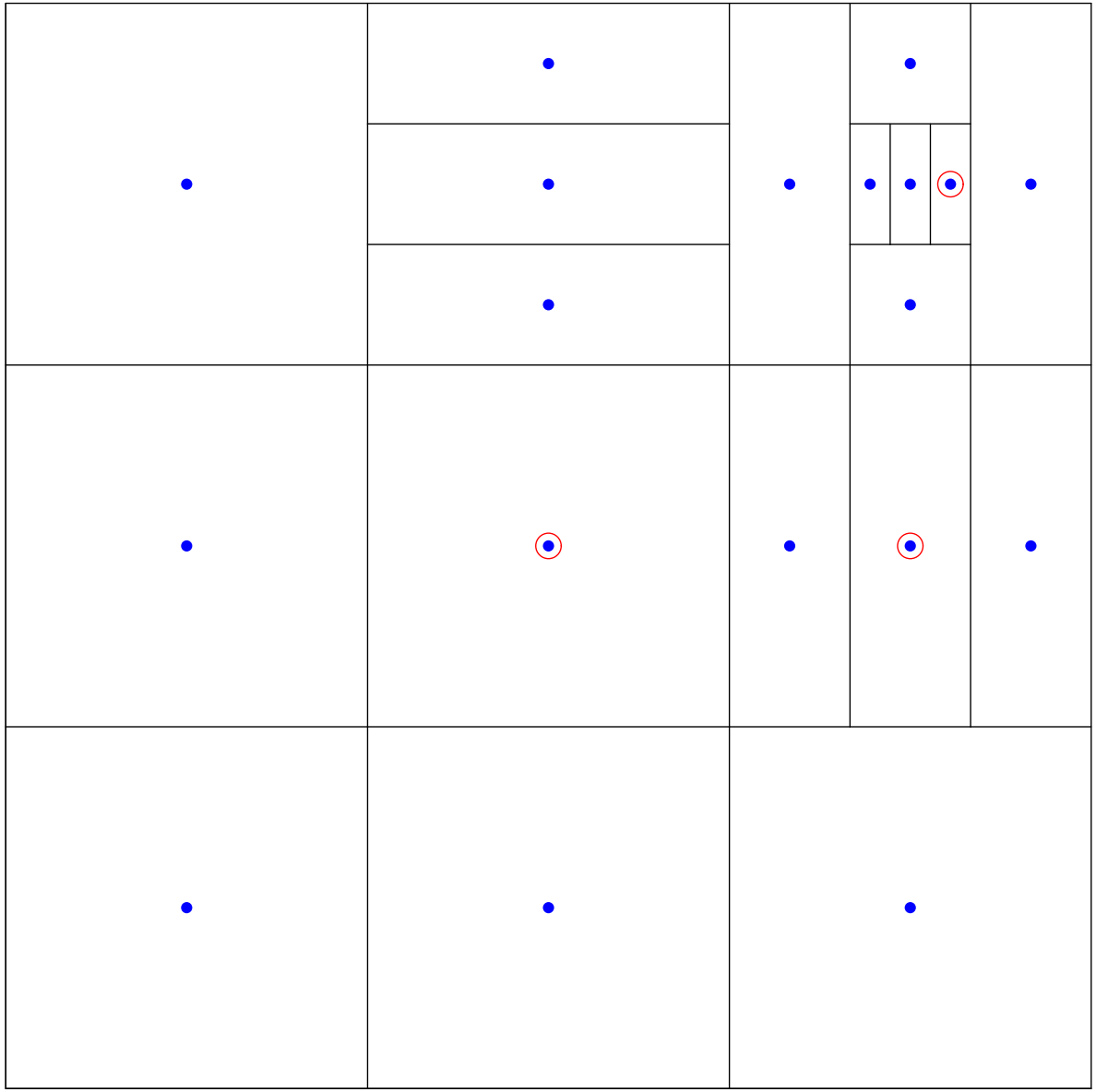


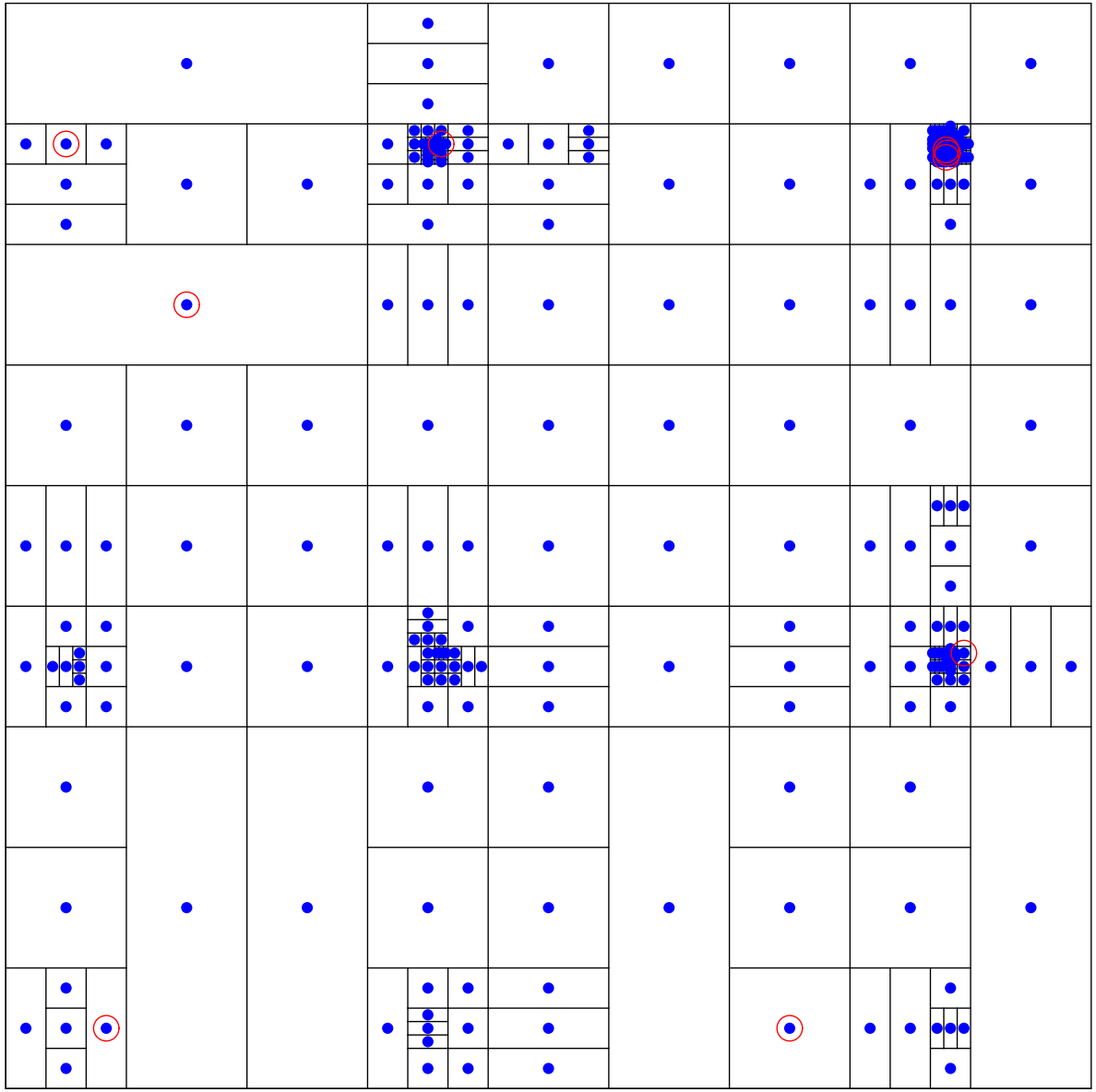












Extension of SOO to the stochastic case

- Play the leaves k according to SOO based on the values

$$\widehat{X}_{k,n_k} + c\sqrt{\frac{\log n}{n_k}},$$

- If a leaf has been played more than $\frac{n}{(\log n)^3}$ times, then expand it.

Performance of SOO

Theorem 2.

If there exists a semi-metric ℓ such that f is locally smooth w.r.t. ℓ and the near-optimality dimension $d = 0$, then

- *Stochastic case:* $\mathbb{E} r_n = \tilde{O}(1/\sqrt{n})$.
- *Deterministic case:* $r_n = O\left(e^{-\frac{c}{D}\sqrt{n}}\right)$.

This is almost as good as HOO optimally fitted!.

Remarks:

- Since the algorithm does not depend on ℓ, the analysis holds for the best possible choice of the semi-metric ℓ satisfying the assumptions.
- SOO adapts to the local unknown smoothness of f.

Example

Let f be such that $f^* - f(x) = \Theta(||x^* - x||^\alpha)$ for some **unknown** $\alpha \geq 0$.

- SOO algorithm does not require the knowledge of ℓ,
- thus the analysis holds for any ℓ satisfying Assumptions 1-4, for example $\ell(x, y) = \|x - y\|^\alpha$.
- Then the near-optimality dimension $d = 0$ and the loss of SOO is $r_n = O(2^{-\sqrt{n}\alpha})$ (stretched-exponential loss),
- This is almost as good as HOO optimally fitted!

Comparison with the DIRECT algorithm

The DIRECT (DIviding RECTangles) algorithm [Jones et al., 1993] is a Lipschitz optimization algorithm where the Lipschitz constant L of f is unknown.

DIRECT uses a similar optimistic splitting technique.

Comparison SOO versus DIRECT:

- Finite-time analysis of SOO (whereas only a consistency property $\lim_{n\to\infty} r_n = 0$ is available for DIRECT in [Finkel and Kelley, 2004])
- Setting of SOO much more general than DIRECT: the function is only locally smooth and the space is semi-metric.
- SOO (deterministic) is a rank-based algorithm
- And SOO is easier to implement...

Application to planning in MDPs

Setting:

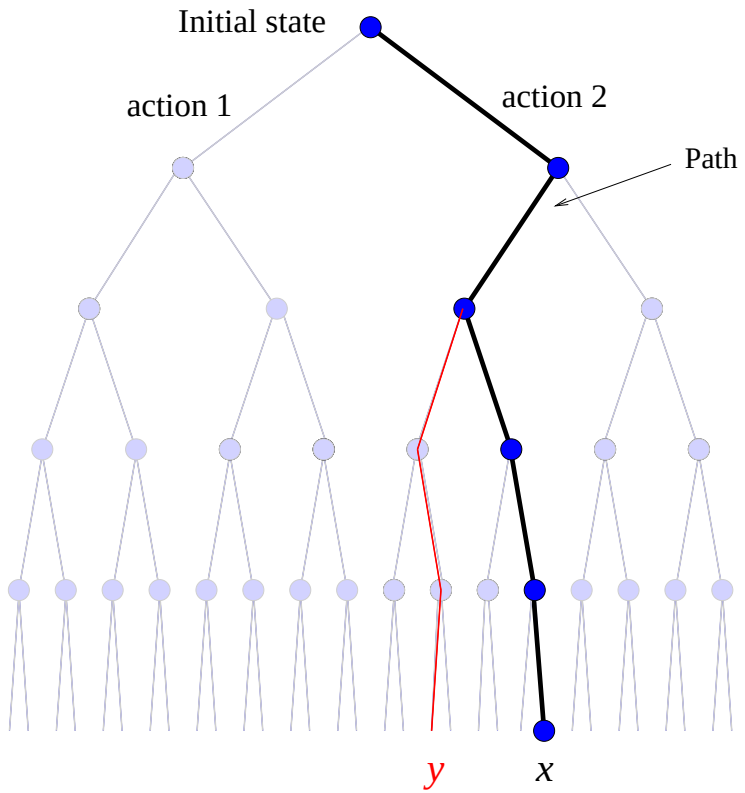
- Assume we have a model of an MDP.
- The state space is large: no way to represent the value function
- Search for the best policy given an initial state, given a computational budget.
- Ex: from current state s_t, given n calls to the model, return the action $a(n)$, play this action in the real environment, observe next state s_{t+1}, and repeat

Simple regret: $r_n \stackrel{\text{def}}{=} \max_{a \in A} Q^*(s_t, a) - Q^*(s_t, a(n))$.

Planning in deterministic systems

From the current state, build the look-ahead tree:

- From the current state s_t
- Search space X = set of paths (infinite sequence of actions)
- Value of any path x: $f(x) = \sum_{t \geq 0} \gamma^t r_t$
- Metric: $\ell(x, y) = \frac{\gamma^{h(x,y)}}{1-\gamma}$
- Prop: f is Lipschitz w.r.t. ℓ
- Use optimistic search to explore the tree with budget n resources



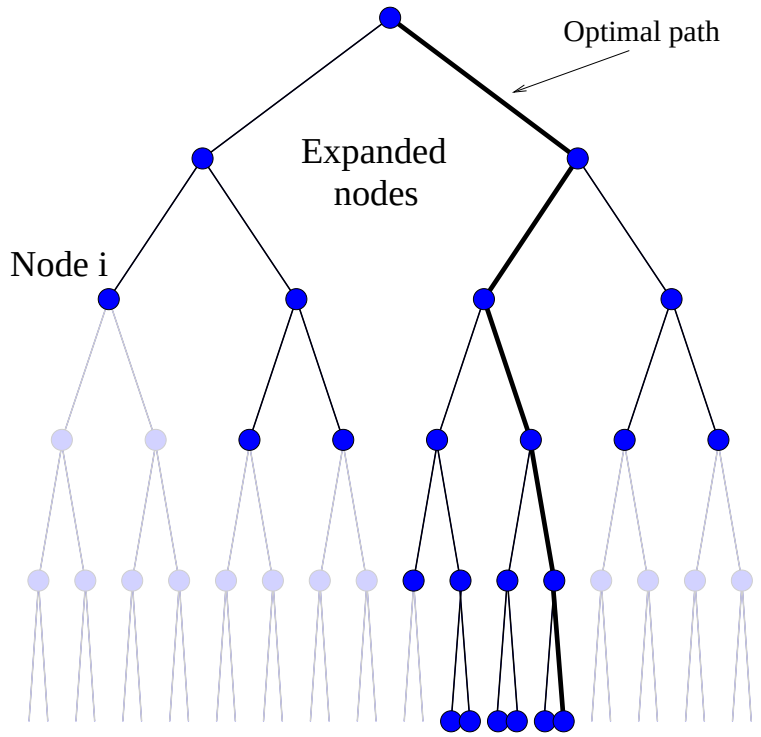
Optimistic exploration

(HOO algo in deterministic setting)

- For any node i of depth d, define the B-values:

$$B_i \stackrel{\text{def}}{=} \sum_{t=0}^{d-1} \gamma^t r_t + \frac{\gamma^d}{1-\gamma} \geq v_i$$

- At each round n, expand the node with highest B-value
- Observe reward, update B-values,
- Repeat until no more available resources
- Return immediate action



Analysis of the regret

[Hren and M., 2008] Define β such that the proportion of ϵ-optimal paths is $O(\epsilon^\beta)$ (this is related to the near-optimal dimension). Let

$$\kappa \stackrel{\text{def}}{=} K\gamma^\beta \in [1, K].$$

- If $\kappa > 1$, then
$$r_n = O\left(n^{-\frac{\log 1/\gamma}{\log \kappa}}\right).$$
(whereas for uniform planning $R_n = O\left(n^{-\frac{\log 1/\gamma}{\log K}}\right)$.)
- If $\kappa = 1$, then we obtain the exponential rate $r_n = O\left(\gamma^{\frac{(1-\gamma)^\beta}{c} n}\right)$, where c is such that the proportion of ϵ-path is bounded by $c\epsilon^\beta$.

Open Loop Optimistic Planning

Setting:

- **Rewards are stochastic** but depend on sequence of actions (and not resulting states)
- Goal : find the sequence of actions that maximizes the expected discounted sum of rewards
- Search space: open-loop policies (sequences of actions)

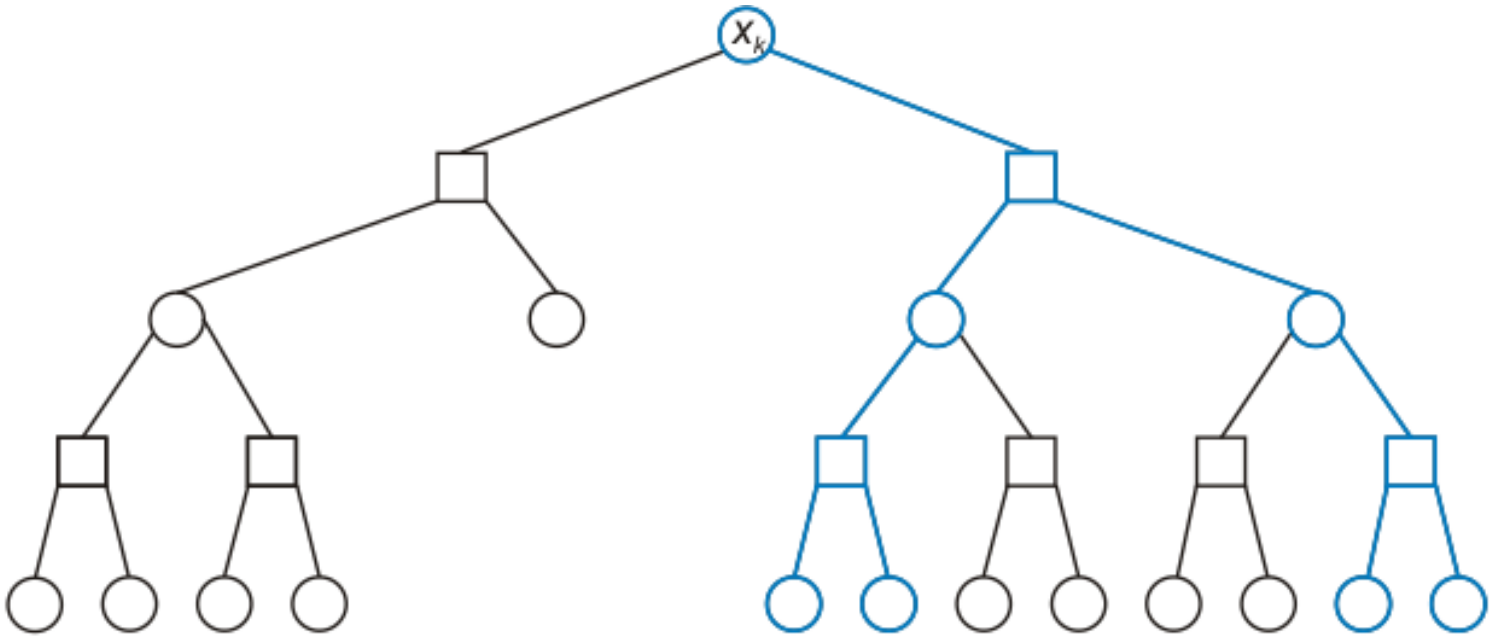
[Bubeck et M., 2010] OLOP algorithm has expected regret

$$\mathbb{E} r_n = \begin{cases} \tilde{O}\left(n^{-\frac{\log 1/\gamma}{\log \kappa}}\right) & \text{if } \gamma\sqrt{\kappa} > 1, \\ \tilde{O}\left(n^{-\frac{1}{2}}\right) & \text{if } \gamma\sqrt{\kappa} \leq 1. \end{cases}$$

Remarks:

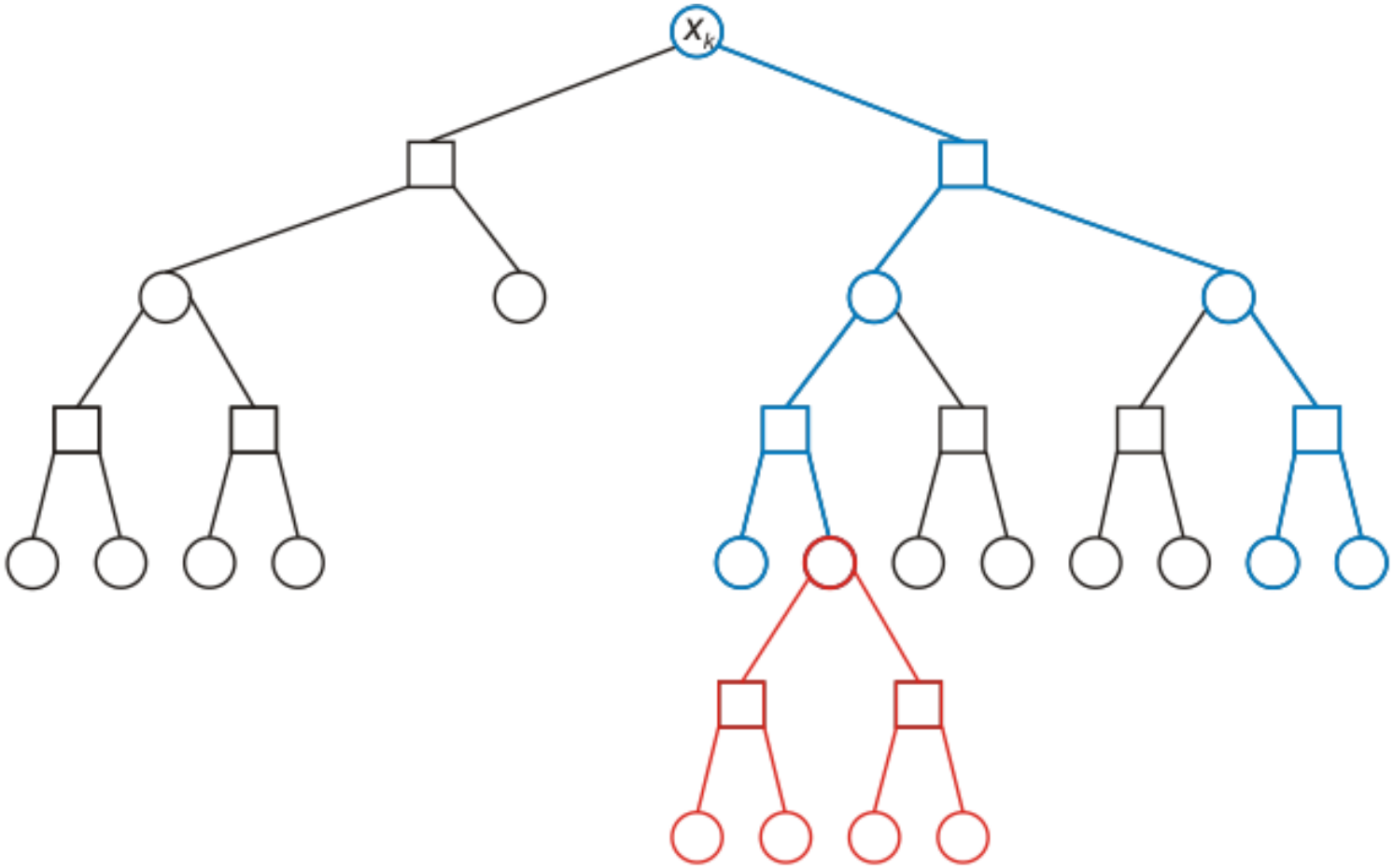
- For $\gamma\sqrt{\kappa} > 1$, this is the same rate as for deterministic systems!
- This is not a consequence of HOO

[Buşoniu and M., 2012]



B-values: upper-bounds on the optimal Q-values Qto B(s) = $1\frac{}{1-\gamma \text{ for leaves}}$
B(s) = $\max_a \sum_{s'} p(s'|s,a)\left[r(s,a,s') + \gamma \max_{a'} b(s',a')\right]$Compute the **optimistic policy**.

Optimistic Planning in MDPs



Expand leaf in the optimistic policy with largest contribution:

$$\arg\max_{s \in \mathcal{L}} P(s) \frac{\gamma^{h(s)}}{1-\gamma}.$$

Performance analysis of OP-MDP

Define X_ϵ the set of states

- whose "contribution" is at least ϵ
- and that belong to an ϵ-optimal policy

Define the measure of complexity of planning in the MDP as the smallest $\beta \geq 0$ such that $|X_\epsilon| = O(\epsilon^{-\beta})$.

Theorem 3.
The performance of OP-MDP is $r_n = O(n^{-1/\beta})$.

Remarks: β is small when

- Structured rewards
- Transition probabilities are heterogeneous

Conclusions on optimistic planning

- Can be seen as applications of hierarchical bandits
- Perform optimistic search in policy space.
- Interesting when the state-space is large (e.g., continuous), and the MDP has structured rewards and transition probabilities.
- Possible extentions to planning in POMDPs

General conclusion

Optimism in the face of uncertainty principle seems successful in several decision making problems:

- Multi-armed bandit problems
- Optimization of deterministic and stochastic functions in general spaces
- For example in planning

Regret analysis = how fast an algorithm converges to the optimal solutions.

Key ingredients of the analysis:

- measure of the quantity of near-optimal solutions,
- and its knowledge
- or design adaptive strategies (SOO).

Related references

- J.Y. Audibert, R. Munos, and C. Szepesvari, *Tuning bandit algorithms in stochastic environments*, ALT, 2007.
- P. Auer, N. Cesa-Bianchi, and P. Fischer, *Finite time analysis of the multiarmed bandit problem*, Machine Learning, 2002.
- S. Bubeck and R. Munos, *Open Loop Optimistic Planning*, COLT 2010.
- S. Bubeck, R. Munos, G. Stoltz, Cs. Szepesvari, *Online Optimization in X-armed Bandits*, NIPS 2008. Long version *X-armed Bandits* JMLR 2011.
- L. Buşoniu, R. Munos, *Optimistic Planning for Markov Decision Processes*, AISTATS 2012.
- P.-A. Coquelin and R. Munos, *Bandit Algorithm for Tree Search*, UAI 2007.
- S. Gelly, Y. Wang, R. Munos, and O. Teytaud, *Modification of UCT with Patterns in Monte-Carlo Go*, RR INRIA, 2006.

Related references (cont'ed)

- J.-F. Hren and R. Munos, *Optimistic planning in deterministic systems*. EWRL 2008.
- M. Kearns, Y. Mansour, A. Ng, *A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes*, Machine Learning, 2002.
- R. Kleinberg, A. Slivkins, and E. Upfal, *Multi-Armed Bandits in Metric Spaces*, ACM Symposium on Theory of Computing, 2008.
- L. Kocsis and Cs. Szepesvri, *Bandit based Monte-Carlo Planning*, ECML 2006.
- T. L. Lai and H. Robbins, *Asymptotically Efficient Adaptive Allocation Rules*, Advances in Applied Mathematics, 1985.