Monte Carlo Tree Search based Variable Selection for High Dimensional Bayesian Optimization

Lei Song[*], Ke Xue[*], Xiaobin Huang, Chao Qian[†]
Email: qianc@lamda.nju.edu.cn

School of Artificial Intelligence
Nanjing University, China

Black-box Optimization

We consider the following problem formulation for black-box optimization (BBO):

$$\max_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})$$

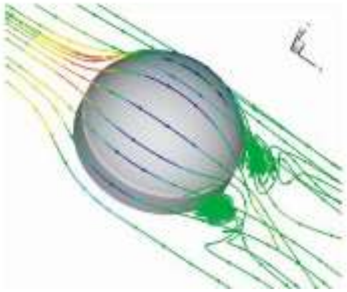
where only the evaluation $f(\boldsymbol{x})$ is available and **no additional information** is known

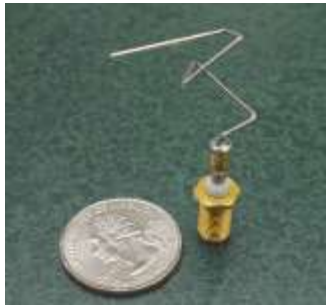
The methods based on first or second order information (e.g., gradient descent) can not be used

Traditional BBO algorithms:

- Evolutionary algorithms
- Evolutionary strategies
- Bayesian optimization

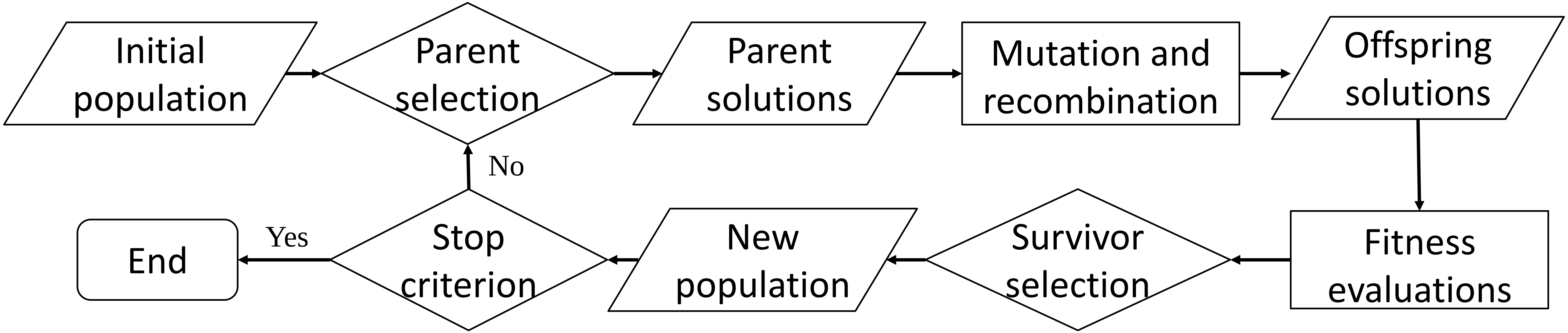
Application:





Evolutionary Algorithms

Evolutionary algorithms (EAs) are a kind of randomized heuristic optimization algorithms, inspired by nature evolution (reproduction with variation + nature selection)



- Population-based and easy to be parallelized
- Discrete inputs

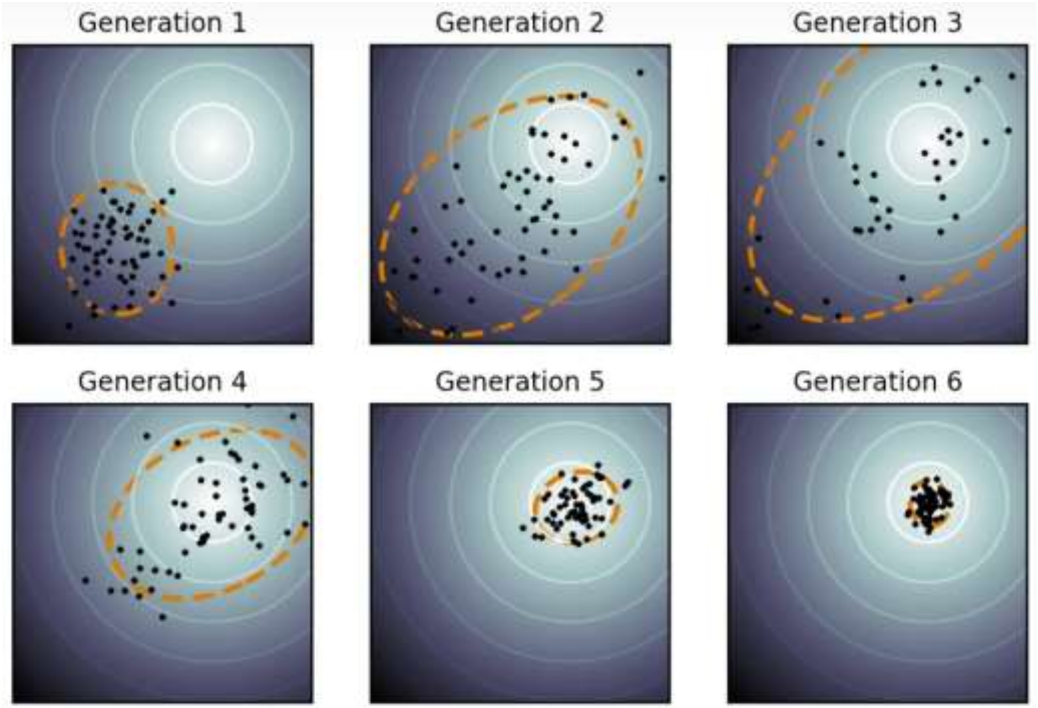
Evolutionary Strategies

Evolutionary strategies (ES) are a popular variant of EAs for continuous problems

CMA-ES is one of the most popular algorithms in ES

CMA-ES samples offspring from $N(m, \Sigma)$ and uses the adaptive covariance matrix to balance exploration and exploitation

- Hundreds of continuous parameters



Bayesian Optimization

Bayesian optimization (BO) uses surrogate model to approximate f and obtains the next query point via acquisition function

Algorithm 1 BO Framework

Input: iteration budget T

Process:

let $D_0 = \emptyset$;

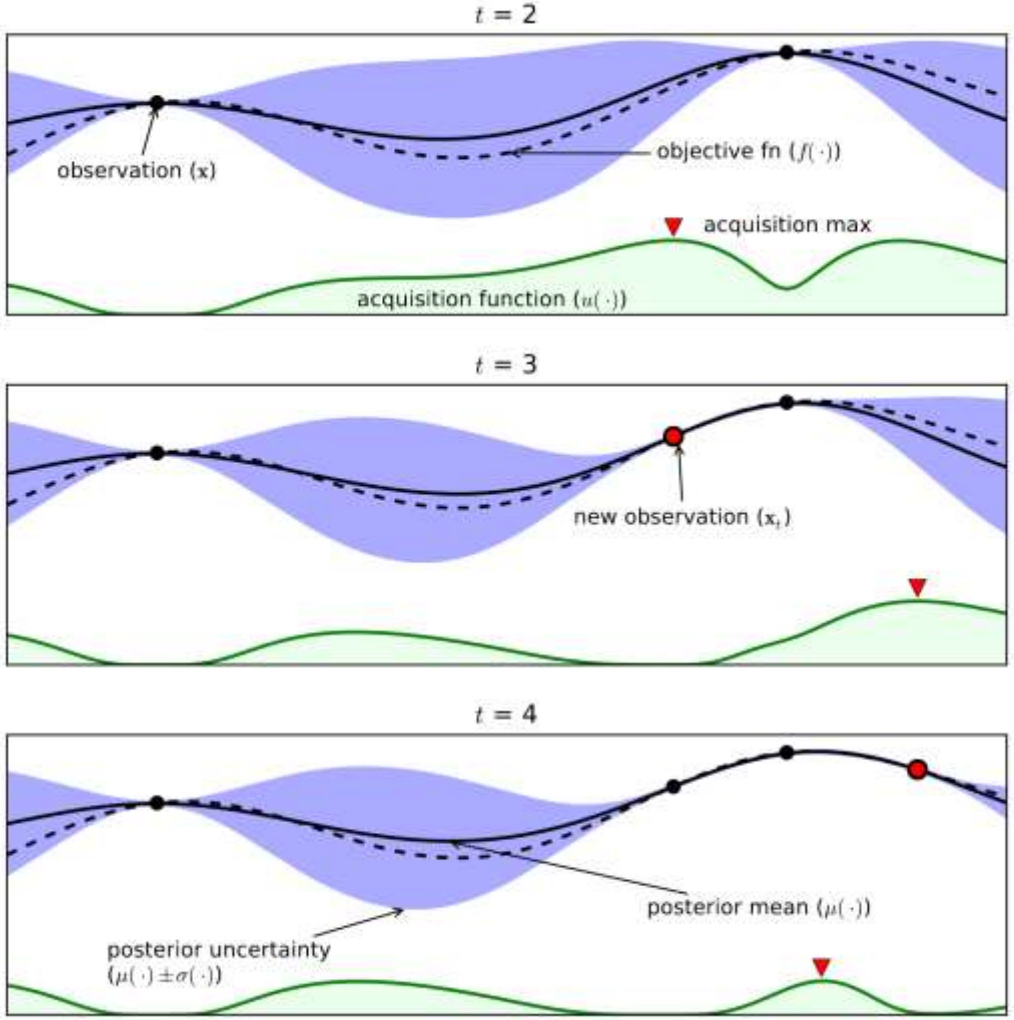
for $t = 1 : T$ **do**

  $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in \mathcal{X}} acq(\boldsymbol{x})$;   obtain next point

  evaluate f at $\boldsymbol{x}_t$ to obtain y_t;

  augment the data $D_t = D_{t-1} \cup \{(\boldsymbol{x}_t, y_t)\}$ and update the GP model   update model

end for



- Less than 30 parameters
- Expensive evaluation

Black-box Optimization

Algorithms	Runtime	Problem scale	Type of variables
EAs	short	large	discrete
ES	short	large	continuous
BO	long	small	continuous

There are still many challenges for BBO in real-world problems, e.g.,

- Multi-objective problems
- High-dimensional problems
- Complex constraints

High-dimensional Bayesian Optimization

Scaling BO to high-dimensional problems is a challenge:

- Search space increases exponentially
- Computation cost of fitting GP and optimizing the acquisition function is time-consuming

Current approaches usually solve high-dimensional BO in a low-dimensional subspace:

1. Obtain a low-dimensional subspace
2. Optimize in the low-dimensional subspace
3. Project the low-dimensional solution back to the high-dimensional space

High-dimensional Bayesian Optimization

Different approaches are based on different assumptions to obtain the low-dimensional subspace:

- **Decomposition**: f can be decomposed into the sum of low-dimensional functions
- **Embedding**: only a few dimensions affect f significantly
- **Variable selection**: only a few *axis-aligned* dimensions affect f significantly

Decomposition

Add-GP-UCB assumes that f can be decomposed into the sum of disjoint low-dimensional functions

$$f(\boldsymbol{x}) = f^1\left(\boldsymbol{x}^{(1)}\right) + \cdots + f^k\left(\boldsymbol{x}^{(k)}\right), \forall i, j, \boldsymbol{x}^{(i)} \cap \boldsymbol{x}^{(j)} = \emptyset$$

- Maximize the likelihood to learn a low-dimensional decomposition
- Optimize the low-dimensional functions separately
- Concatenate variables of low-dimensional functions

$$\begin{aligned} &f(x) \\ &= f^{(1)}(x_1, x_3, x_4) \\ &+ f^{(2)}(x_2) \\ &+ f^{(3)}(x_5, x_6) \end{aligned}$$

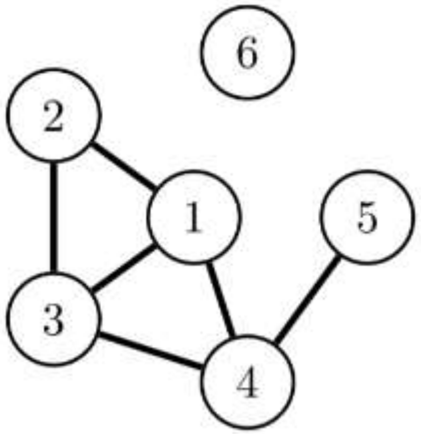
Decomposition

Overlapping generalizes Add-GP-UCB to overlapping conditions

$$f(\boldsymbol{x}) = f^1\left(\boldsymbol{x}^{(1)}\right) + \cdots + f^k\left(\boldsymbol{x}^{(k)}\right)$$

- Maximize the likelihood to learn a low-dimensional decomposition
- Optimize the low-dimensional functions on the graph similar to message passing
- Concatenate variables of low-dimensional functions

$$\begin{aligned} &f(x) \\ &= f^{(1)}(x_1, x_2, x_3) \\ &+ f^{(2)}(x_1, x_3, x_4) \\ &+ f^{(3)}(x_4, x_5) \\ &+ f^{(4)}(x_6) \end{aligned} \quad \Longleftrightarrow$$



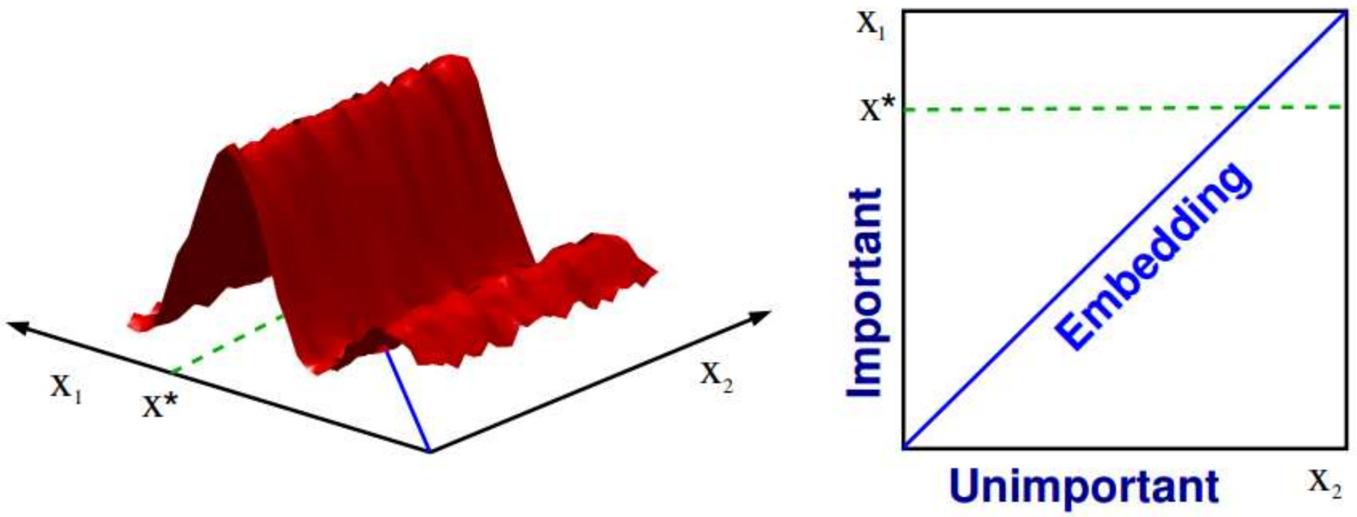
Embedding

Assuming low effective dimensionality, **REMBO** uses a random embedding matrix to obtain the low-dimensional subspace:

$$\mathbf{M} \in \mathbb{R}^{D*d}, \mathbf{M}_{ij} \sim N(0,1)$$

Then, the optimization problem is:

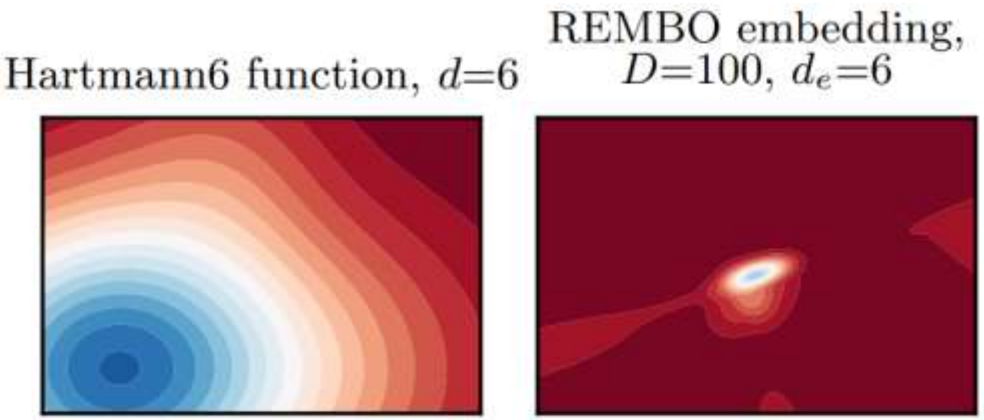
$$\max_{z \in \mathbb{R}^d} f(\mathbf{M}z)$$

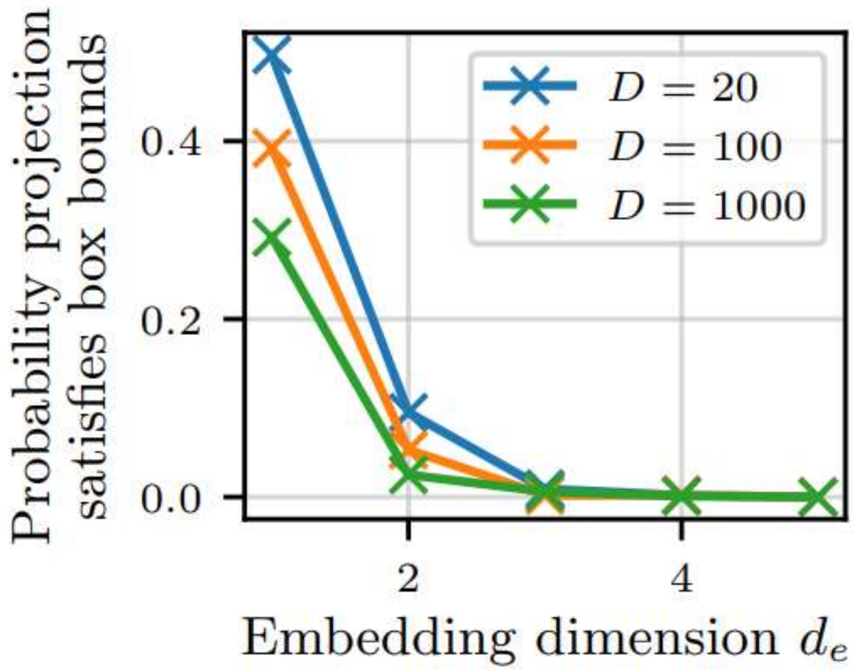


Embedding

ALEBO improves several misconceptions in previous embedding methods, e.g.,

- The box bounds result in a nonlinear distortion of the search space
- Many points map to the facets





Methods:

- A Mahalanobis kernel
- A constrained acquisition function optimization

Variable Selection

Dropout: select d variables randomly and optimize the selected variables

- Select d variables randomly
- Optimize the selected variables
- Use “fill-in” strategy to obtain the unselected variables

Advantage:

- Variable selection is much simper than embedding and can reduce the runtime

How can we do better than random selection?

Motivation

The importance of different variables are different

Thus, randomly select variables is inefficient

We should pay more attention to the important variables!

~~Select variables randomly~~ ⇒ Select important variables

- A metric for the importance of variables
- An algorithm to select variables

MCTS-VS

Monte Carlo Tree Search based Variable Selection (MCTS-VS) uses MCTS to iteratively select and optimize a subset of important variables, and uses “fill-in” strategy to obtain the unselected variables

- Variable score $\boldsymbol{s}$ is the metric of the importance of variables
- MCTS is employed to partition the variables into important and unimportant ones, select and optimize the important variables
- “Fill-in” strategy for unselected variables

Variable Score

Variable score $\boldsymbol{s} \in \mathbb{R}^D$ is a D-dimensional vector, where the i-th element represents the importance of the i-th variable

$$\boldsymbol{s} = \left(\sum_{(\mathbb{M},\boldsymbol{\mathcal{D}})\in\mathbb{D}} \boxed{\sum_{(\boldsymbol{x}^i,y^i)\in\boldsymbol{\mathcal{D}}} y^i \cdot g(\mathbb{M})} \right) \Big/ \boxed{\left(\sum_{(\mathbb{M},\boldsymbol{\mathcal{D}})\in\mathbb{D}} |\boldsymbol{\mathcal{D}}| \cdot g(\mathbb{M}) \right)}$$

The sum of query evaluations optimizing the variables indexed by $\mathbb{M}$

The number of queries using each variable

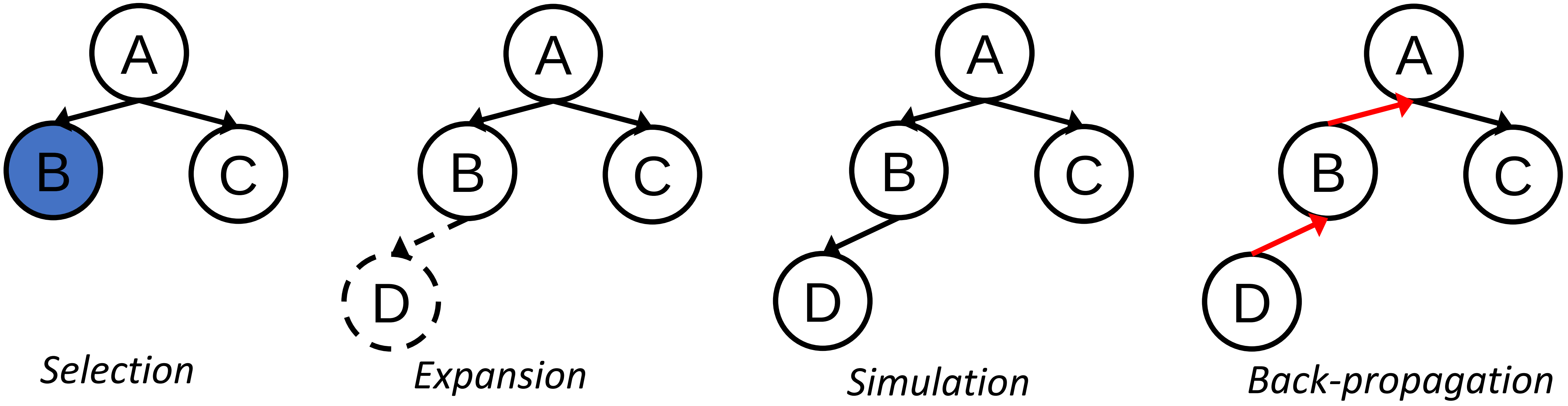
- $(\mathbb{M}, \boldsymbol{\mathcal{D}})$ represents the indices of selected variables and the corresponding samples
  - E.g., $\mathbb{M} = \{2, 5, 7\}$ and $\boldsymbol{\mathcal{D}}$ is obtained by optimizing $\left\{\left([x_2, x_5, x_7]^i, y^i\right)\right\}_{i=1}^{t-1}$
- $g: 2^{[D]} \to \{0, 1\}^D$, and the i-th element is 1 if $i \in \mathbb{M}$, and 0 otherwise

A Brief Introduction to MCTS

Tree node X represents the state, and stores v_X representing its goodness and the number n_X of visits

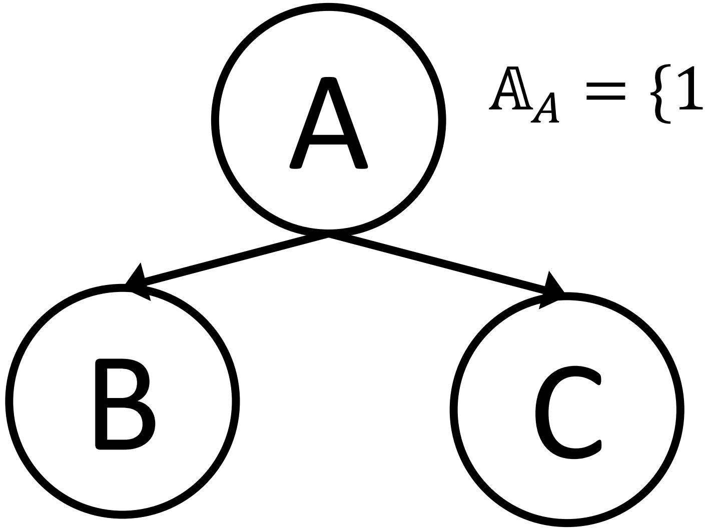
UCB is used to select node, balancing the exploitation and exploration:

$$v_X + 2C_p\sqrt{2(\log n_p)/n_X}$$



Example Illustration of MCTS-VS

Current state of MCTS-VS



$\mathbb{A}_A = \{1, 2, \dots, 9\}$

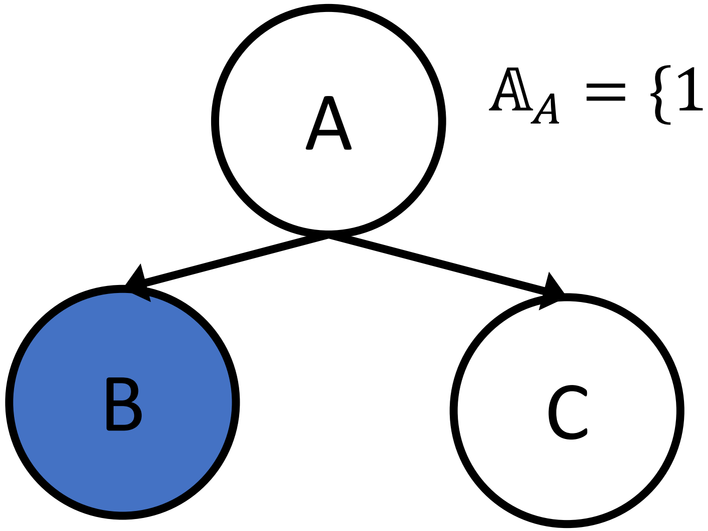
$\mathbb{A}_B = \{1, 2, 4, 7, 8\}$ $\mathbb{A}_C = \{3, 5, 6, 9\}$

Tree node X represents a subset of variables, denoted by index set $\mathbb{A}_X \subseteq [D]$

- The root node represents all variables
- v_X is defined as the average score (i.e., importance) of variables contained by X, which is calculated by $\boldsymbol{s} \cdot g(\mathbb{A}_X)/|\mathbb{A}_X|$
- n_X is the number of visits

Example Illustration of MCTS-VS

Select node B based on UCB value

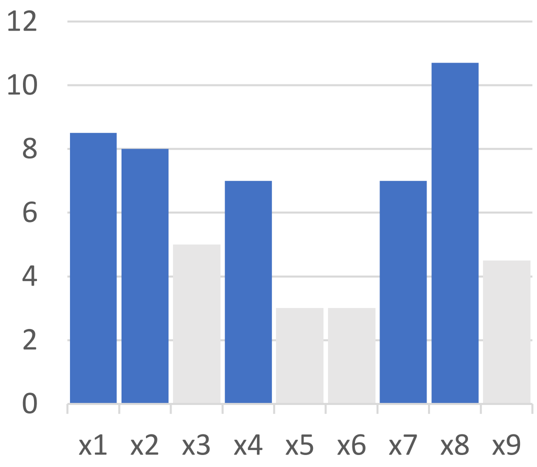


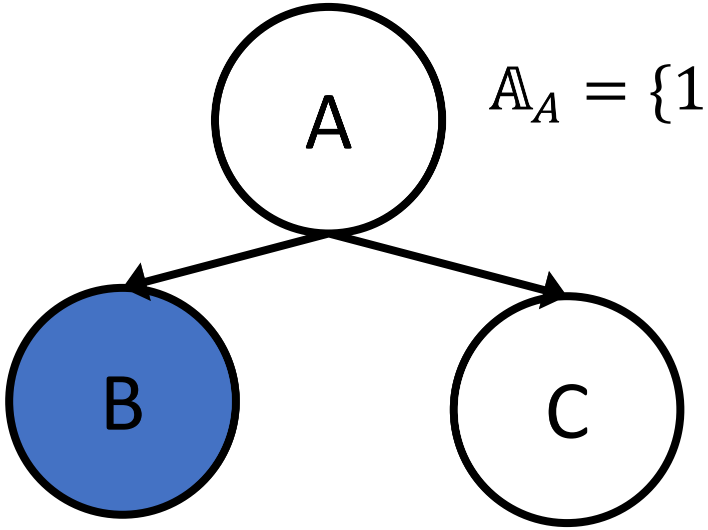
$\mathbb{A}_A = \{1, 2, \dots, 9\}$

$\mathbb{A}_B = \{1, 2, 4, 7, 8\}$ $\mathbb{A}_C = \{3, 5, 6, 9\}$

Example Illustration of MCTS-VS

Variable score $\boldsymbol{s}$



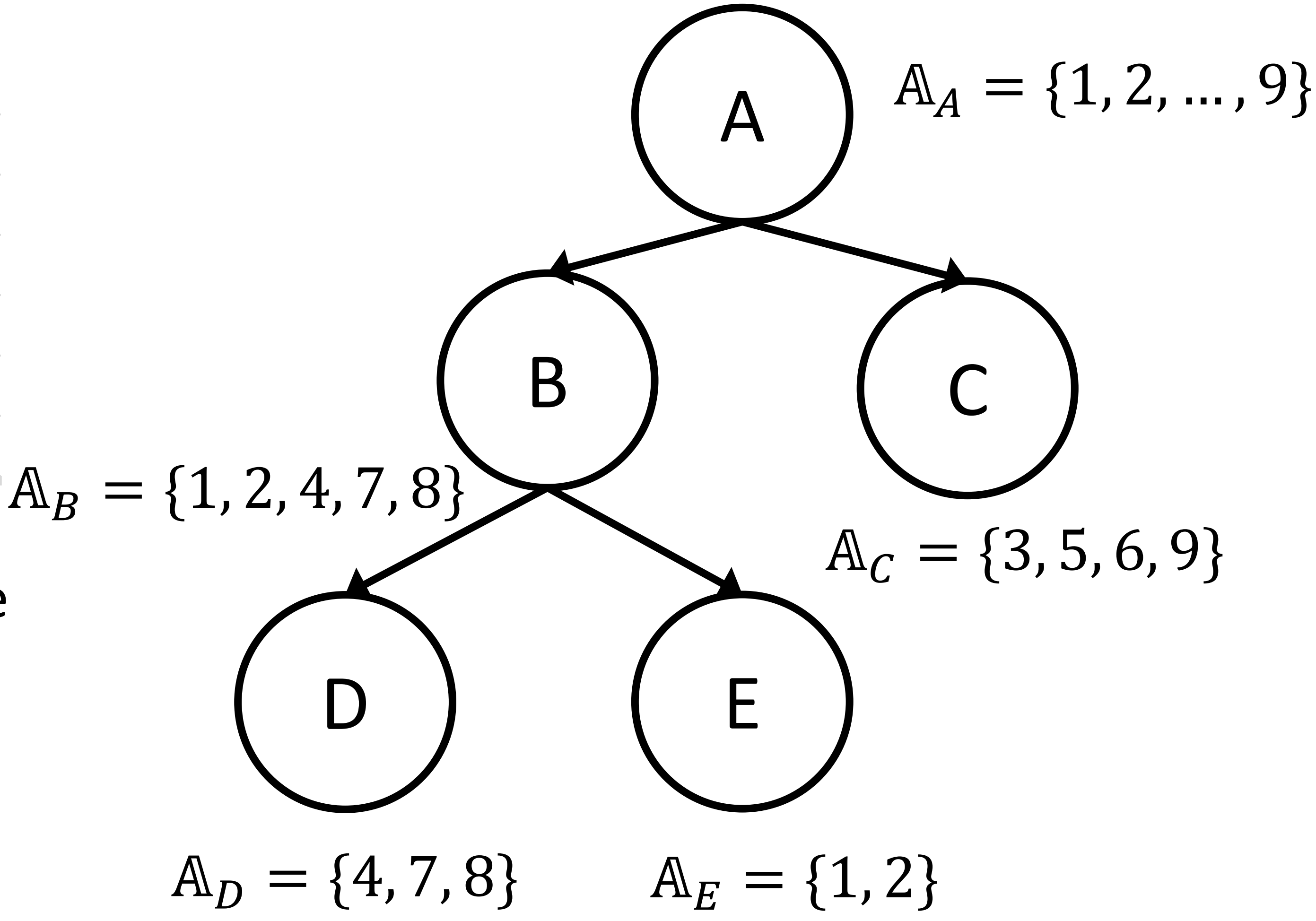


$\mathbb{A}_A = \{1, 2, \ldots, 9\}$

$\mathbb{A}_B = \{1, 2, 4, 7, 8\}$ $\mathbb{A}_C = \{3, 5, 6, 9\}$

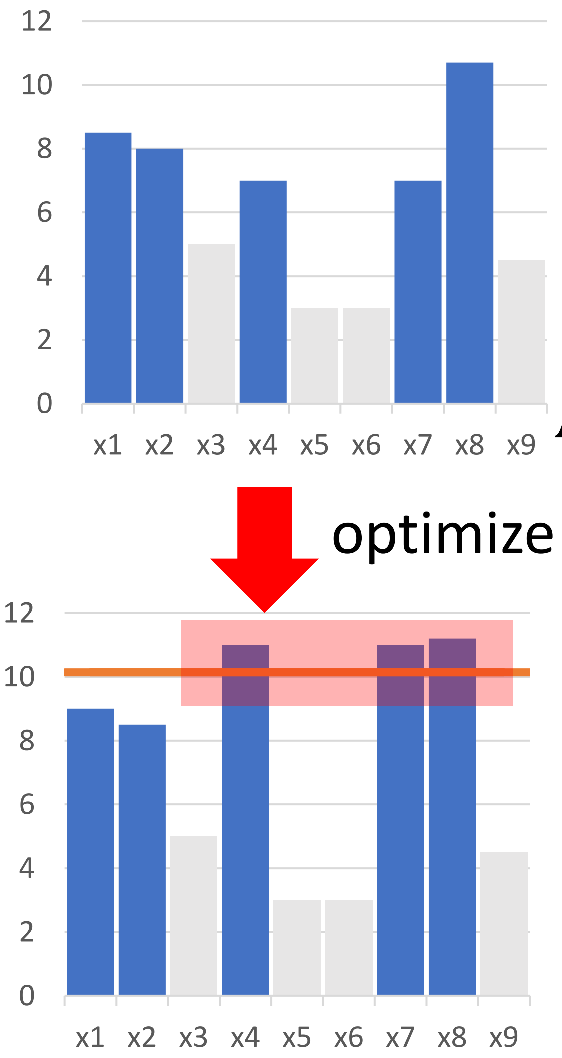
Example Illustration of MCTS-VS

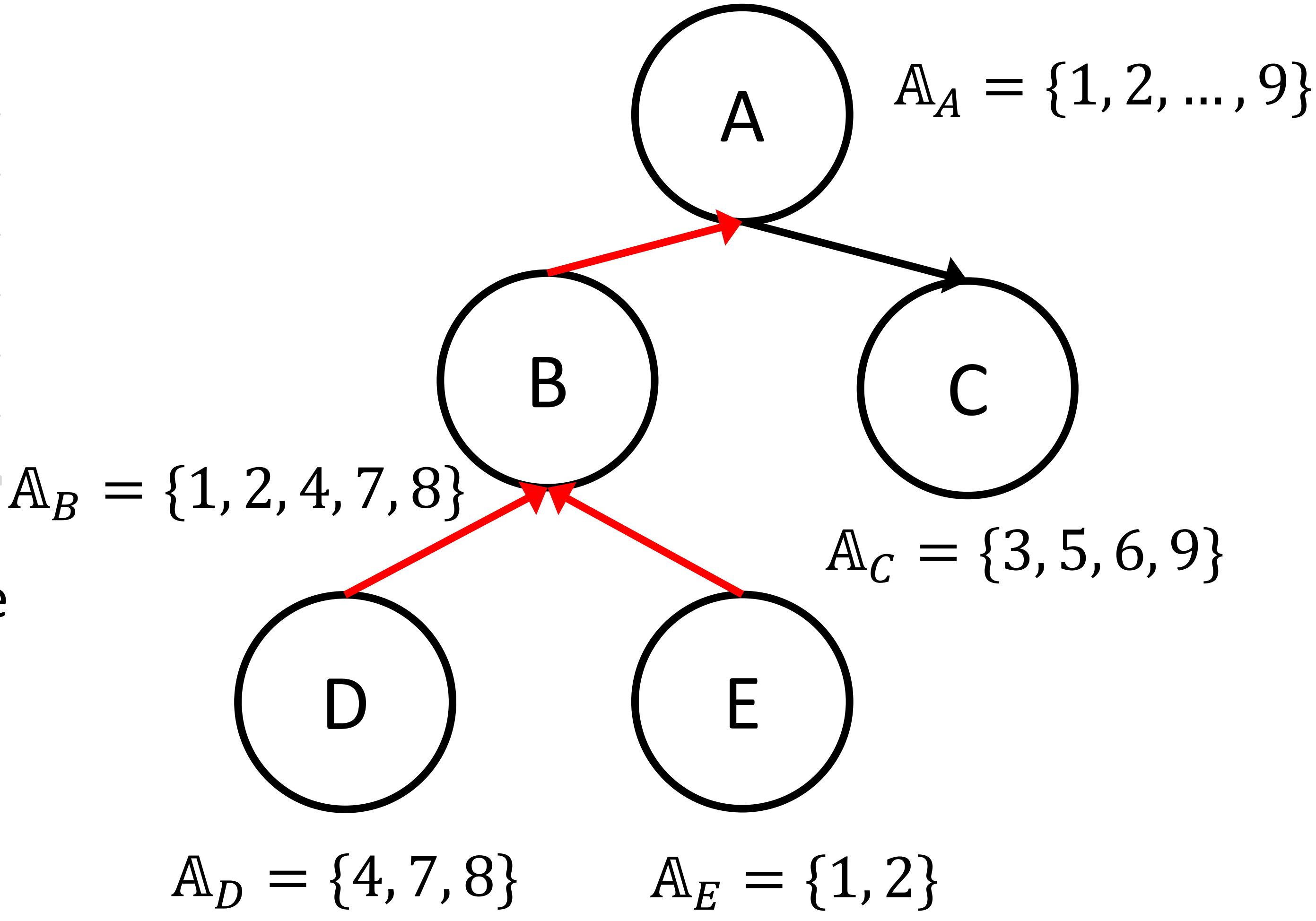
Variable score $\boldsymbol{s}$



Example Illustration of MCTS-VS

Variable score $\boldsymbol{s}$





“Fill-in” Strategy

The best k samples: $\{(\boldsymbol{x}^{*j}, y^{*j})\}_{j=1}^{k}$, for unselected variable x_i:

- The best-k strategy: x_i is uniformly selected from $\left\{x_i^{*j}\right\}_{j=1}^{k}$ at random
- The average best-k strategy: x_i is the average of $\left\{x_i^{*j}\right\}_{j=1}^{k}$
- The random strategy: x_i is sampled from the domain randomly

Theoretical Analysis

- Cumulative regret $R_T = \sum_{t=1}^{T}(f(x^*) - f(x^t))$
- Assumption: The function f is a GP sample path. For some $a, b > 0$, given $L > 0$, the partial derivatives of f satisfy that $\forall i \in [D], \exists \alpha_i \geq 0$,

$$P\left(\sup_{\boldsymbol{x}\in\mathcal{X}}\left|\frac{\partial f}{\partial x_i}\right| < \alpha_i L\right) \geq 1 - ae^{-\left(\frac{L}{b}\right)^2}$$

- Theorem: $\forall\delta \in (0,1)$, let $\beta_t = 2\log\left(\frac{4\pi_t}{\delta}\right) + 2d_t \log(d_t t^2 br\sqrt{\log(\frac{4Da}{\delta})})$ and $L = b\sqrt{\log\frac{4Da}{\delta}}$, and $\{\pi_t\}_{t\geq 1}$ satisfies $\sum_{t\geq 1}\pi_t^{-1} = 1$ and $\pi_t > 0$. Let $\beta_T^* = \max_{1\leq i\leq T}\beta_t$. At iteration T,

$$R_T \leq \sqrt{C_1 T\beta_T^*\gamma_T} + 2\alpha_{max} + 2\sum_{t=1}^{T}\sum_{i\in[D]\setminus\mathbb{M}_t}\alpha_i^* Lr$$

Theoretical Analysis

$$R_T \leq \boxed{\sqrt{C_1 T \beta_T^* \gamma_T} + 2\alpha_{max}} + \boxed{2\sum_{t=1}^{T} \sum_{i \in [D] \setminus \mathbb{M}_t} \alpha_i^* Lr}$$

- The regret from optimization
- The regret from unselected variables

Insight:

- The variable selection can reduce the computational complexity while increasing the regret
- A good variable selection algorithm should select as important variables as possible, i.e., variables with as larger α_i^* as possible

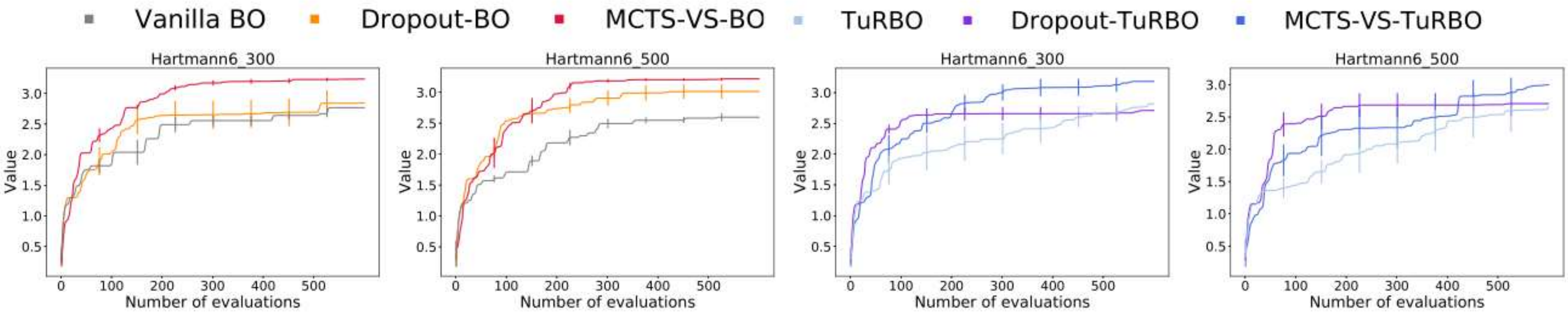
Experiments

We want to know the following research questions (RQs):

- RQ1: Can BO benefit from variable selection?
- RQ2: How does MCTS-VS perform compared with state-of-the-art methods?
- RQ3: How about the runtime of MCTS-VS?
- RQ4: Can MCTS-VS select more important variables than Dropout? (why)
- RQ5: Is MCTS-VS sensitive to the hyper-parameters?

Experiments – RQ1

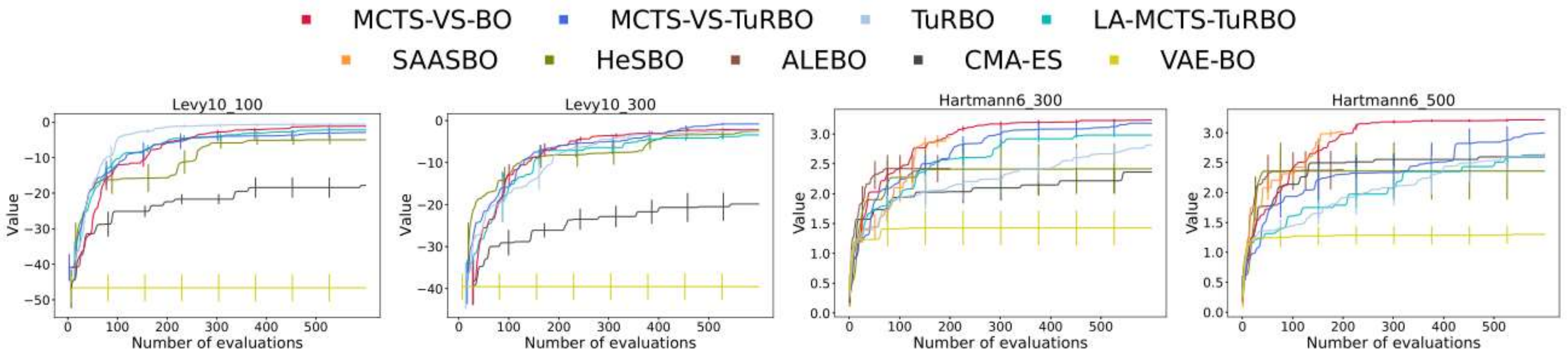
Effectiveness of variable selection:



- Dropout is better than BO without variable selection
- MCTS-VS is better than Dropout

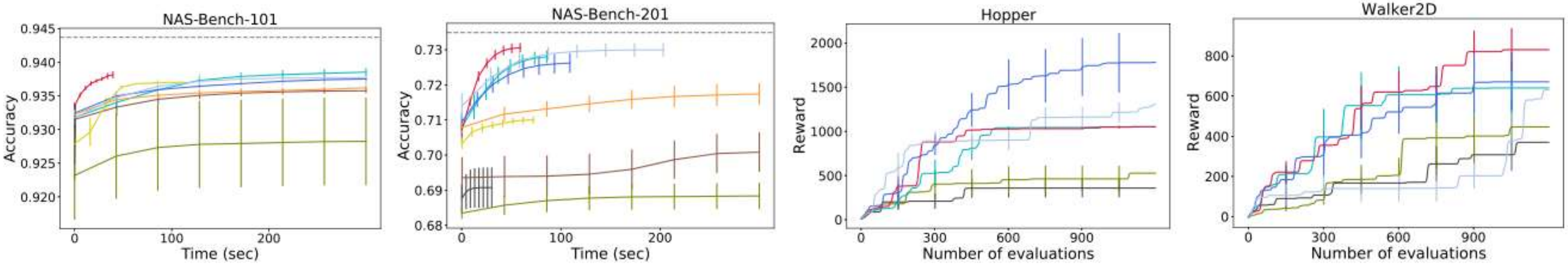
Experiments – RQ2

Synthetic functions:



Real-world problems:

MCTS-VS is comparable with sota methods



Experiments – RQ3

Runtime comparison:

Method	Levy10_100	Levy10_300	Hartmann6_300	Hartmann6_500
Vanilla BO	3.190	4.140	4.844	5.540
Dropout-BO	2.707	3.225	3.237	3.685
MCTS-VS-BO	2.683	3.753	3.711	4.590
TuRBO	8.621	9.206	9.201	9.754
LA-MCTS-TuRBO	14.431	22.165	25.853	34.381
MCTS-VS-TuRBO	4.912	5.616	5.613	5.893
SAASBO	/	/	2185.678	4163.121
HeSBO	220.459	185.092	51.678	55.699
ALEBO	/	/	470.714	512.641
CMA-ES	0.030	0.043	0.043	0.045

Variable selection can reduce the runtime

Experiments – RQ4

Recall comparison:

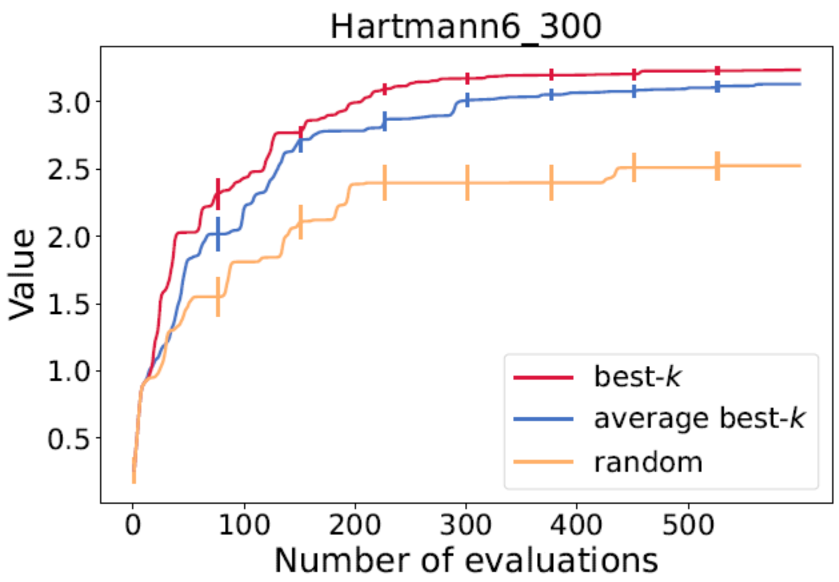
METHOD	LEVY10_100	LEVY10_300	HARTMANN6_300	HARTMANN6_500
DROPOUT	0.100	0.030	0.020	0.012
MCTS-VS	0.429	0.433	0.352	0.350

Recall $\frac{d_t^*}{d}$ is used to compare the quality of variable selection, where d_t^* is the number of valid variables selected at iteration t and d is the number of valid variables

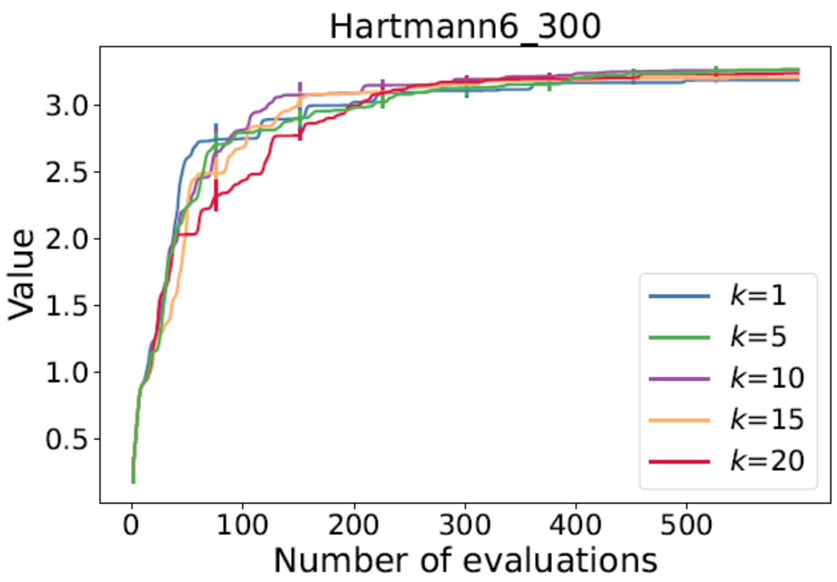
- Dropout: $\frac{d}{D}$ in expectation
- MCTS-VS: run for 600 evaluations on five different random seeds and calculate the average recall

The recall of MCTS-VS is much larger than Dropout

Experiments – RQ5



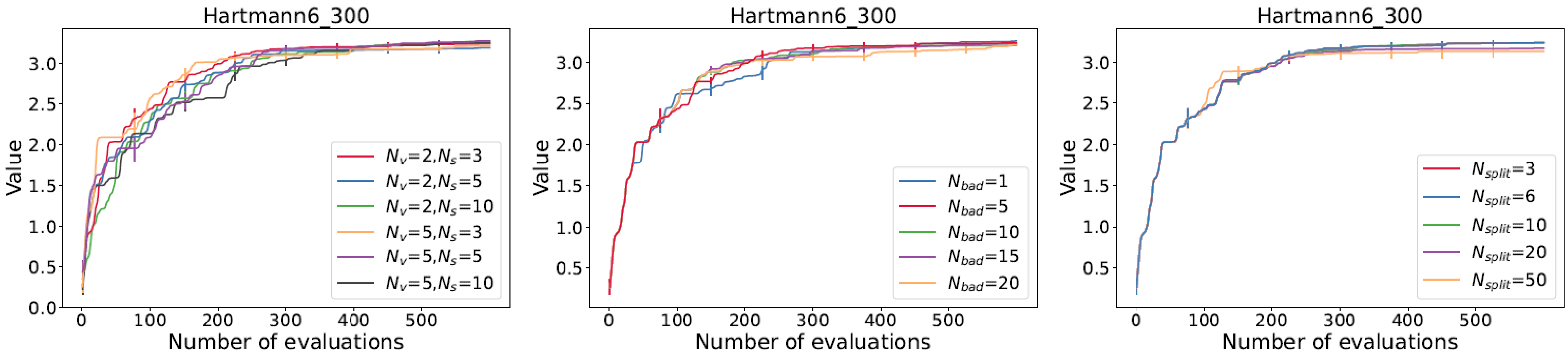
(a) “Fill-in” strategy



(b) Hyper-parameter k of the best-k strategy

The best-k strategy is good, and MCTS-VS is not sensitive to the selection of k

Experiments – RQ5



(a) Number of samples (b) N_{bad} (c) N_{split}

MCTS-VS is not sensitive to other hyper-parameters

Conclusion

MCTS-VS uses MCTS to recursively partition the variables into important and unimportant ones, and only optimizes those important variables

Feature work:

- A more well-designed metric for importance
- A specific theoretical analysis for MCTS-VS

Thank you!