A PROOFS

Lemma 1. *The algorithm to uniformly draw k samples in S, pick the best and return is a $(1, 1)$-oracle.*

Proof. Consider the following simple $(1, 1)$-oracle for single-objective optimization: after sampling k samples, we rank them according to their function values, and split them into two $k/2$ smaller subsets $\tilde{S}_{\text{good}}$ and $\tilde{S}_{\text{bad}}$. Other points are randomly assigned to either of the two subsets. Then if $\mathbf{x}^*$ happens to be among the k collected samples (which happens with probability $k/|S|$), definitely we have $\mathbf{x}^* \in S_{\text{good}}$. Therefore, we have:

$$P\left(\mathbf{x}^* \in S_{\text{good}} | \mathbf{x}^* \in S\right) \geq \frac{k}{|S|} \geq 1 - \exp\left(-\frac{k}{|S|}\right) \tag{7}$$

which is an oracle with $\alpha = \eta = 1$. The last inequality is due to $e^x \geq 1 + x$ (and thus $e^{-x} \geq 1 - x$). □

Lemma 2. *Define $g(\lambda) : \mathbb{R}^+ \mapsto \mathbb{R}^+$ as:*

$$g(\lambda) : \lambda \mapsto \sum_{t=1}^{T} w_t \log\left(1 + \frac{1}{\lambda w_t}\right) \tag{8}$$

The following maximization problem

$$\max_{\{z_t\}} \sum_{t=1}^{T} \log\left(1 - e^{-z_t}\right) \quad \text{s.t.} \sum_{t=1}^{T} w_t z_t = K \tag{9}$$

has optimal solutions

$$z_t^* = \log\left(1 + \frac{1}{\lambda w_t}\right), \quad 1 \leq t \leq T \tag{10}$$

where λ is determined by $g(\lambda) = K$. With optimal $\{z_t^*\}$, the objective reaches $-\sum_t \log(1 + \lambda w_t)$.

Proof. Its Lagrange is:

$$J\left(\{z_t\}\right) = \sum_{t=1}^{T} \log\left(1 - e^{-z_t}\right) - \lambda\left(\sum_{t=1}^{T} w_t z_t - K\right) \tag{11}$$

Taking derivative w.r.t. z_t and we have:

$$\begin{aligned}
\frac{\partial J}{\partial z_t} &= \frac{e^{-z_t}}{1 - e^{-z_t}} - \lambda w_t = 0. \\
\frac{1}{1 - e^{-z_t}} - 1 - \lambda w_t &= 0 \\
\frac{1}{1 - e^{-z_t}} &= 1 + \lambda w_t \\
1 - e^{-z_t} &= \frac{1}{1 + \lambda w_t} \\
e^{-z_t} = 1 - \frac{1}{1 + \lambda w_t} &= \frac{\lambda w_t}{1 + \lambda w_t} \\
z_t = -\log\frac{\lambda w_t}{1 + \lambda w_t} = \log\frac{1 + \lambda w_t}{\lambda w_t} &= \log\left(1 + \frac{1}{\lambda w_t}\right)
\end{aligned} \tag{12}$$

□

Lemma 3. *Both $g(\lambda)$ and $g^{-1}(y)$ are monotonously decreasing. Furthermore, let $\bar{w} := T\left(\sum_{t=1}^{T} w_t^{-1}\right)^{-1}$ be the Harmonic mean of $\{w_t\}$ and $w_{\max} := \max_{t=1}^{T} w_t$, we have:*

$$\frac{\bar{w}^{-1}}{\exp(\bar{w}^{-1} y/T) - 1} \leq g^{-1}(y) \leq \frac{w_{\max}^{-1}}{\exp(w_{\max}^{-1} y/T) - 1} \leq \frac{T}{y}. \tag{13}$$

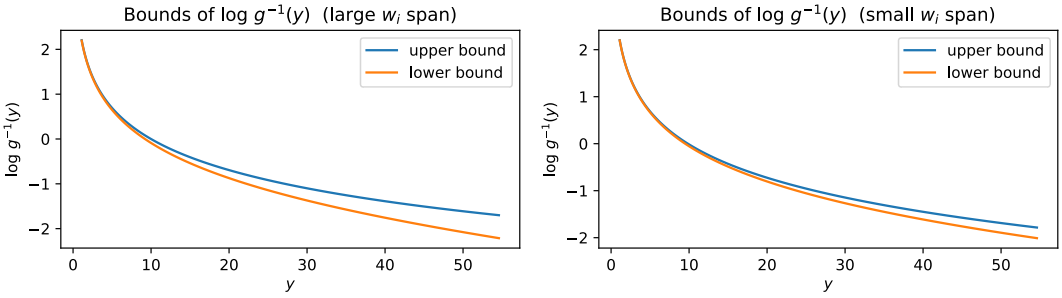


Figure 8: Upper and lower bounds of $g^{-1}(y)$ with different $\{w_i\}$. Left: $w_i = 2^{\text{linspace}(-0.1,10)}$. Right: $w_i = 2^{\text{linspace}(2,5)}$. Small $\{w_i\}$ span leads to better bounds.

Proof. It is easy to see when λ increases, each term in $g(\lambda)$ decreases and thus $g(\lambda)$ is a decreasing function of λ. Therefore, its inverse mapping $g^{-1}(y)$ is also decreasing.

Let $\mu(y) := 1/g^{-1}(y) > 0$. Then we have:

$$y = \sum_{t=1}^{T} w_t \log\left(1 + \frac{\mu(y)}{w_t}\right) \tag{14}$$

It is clear that when $y = 0$, $\mu(y) = 0$. Taking derivative with respect to y in both side, we have:

$$1 = \mu'(y) \sum_{t=1}^{T} \frac{1}{1 + \frac{\mu(y)}{w_t}} \tag{15}$$

where $\mu'(y) = \frac{\mathrm{d}\mu(y)}{\mathrm{d}y}$ is the derivative of $\mu(y)$. Using the property of Harmonic mean, we have:

$$\mu'(y) = \left(\sum_{t=1}^{T} \frac{1}{1 + \frac{\mu(y)}{w_t}}\right)^{-1} \leq \frac{\sum_{t=1}^{T} 1 + \frac{\mu(y)}{w_t}}{T^2} = \frac{1}{T}\left(1 + \frac{\mu(y)}{\bar{w}}\right) \tag{16}$$

This gives:

$$\frac{\mu'(y)}{1 + \mu(y)/\bar{w}} \leq \frac{1}{T} \tag{17}$$

Integrate on both side starting from $y = 0$, we have:

$$\bar{w} \log(1 + \mu(y)/\bar{w})\Big|_0^y \leq \frac{y}{T}\Big|_0^y \tag{18}$$

Using $\mu(0) = 0$ we thus have:

$$\bar{w} \log(1 + \mu(y)/\bar{w}) \leq \frac{y}{T} \tag{19}$$

This leads to $\mu(y) \leq \bar{w}\left[\exp(y\bar{w}^{-1}/T) - 1\right]$. With $g^{-1}(y) = 1/\mu(y)$, we arrive the final lower bound for $g^{-1}(y)$.

For an alternative upper bound of $g^{-1}(y)$, we just notice that (here $w_{\max} := \max_t w_t$):

$$\mu'(y) = \left(\sum_{t=1}^{T} \frac{1}{1 + \frac{\mu(y)}{w_t}}\right)^{-1} \geq \left(\frac{T}{1 + \frac{\mu(y)}{w_{\max}}}\right)^{-1} = \frac{1}{T}\left(1 + \frac{\mu(y)}{w_{\max}}\right) \tag{20}$$

Using the same technique as above, we have $\mu(y) \geq w_{\max}\left[\exp(yw_{\max}^{-1}/T) - 1\right]$ and the upper bound of $g^{-1}(y)$ follows.

Finally, note that $e^x \geq 1 + x$, we have

$$\frac{w_{\max}^{-1}}{\exp(w_{\max}^{-1} y/T) - 1} \leq \frac{w_{\max}^{-1}}{w_{\max}^{-1} y/T} = \frac{T}{y} \tag{21}$$

□

Theorem 1. *Following optimal sequence, the algorithm yields a reward r^* lower bounded by the following:*

$$r^* \geq r_{\mathrm{b}} \exp\left[\left(\log 2 - \frac{\eta N^\alpha \phi(\alpha, T)}{K}\right) T\right] \tag{22}$$

where $r_{\mathrm{b}} := N^{-1}$ and $\phi(\alpha, T) := (1 - 2^{-\alpha T})/(1 - 2^{-\alpha})$.

Proof. First note that $|S_T| \leq |S_0|/2^T$ and thus $\frac{1}{|S_T|} \geq 2^T/N$. So we just need to bound $P(\mathbf{x}^* \in S_T)$, which can be written as:

$$P(\mathbf{x}^* \in S_T) = \prod_{t=1}^{T} P(\mathbf{x}^* \in S_t | \mathbf{x}^* \in S_{t-1}) \geq \prod_{t=1}^{T} \left(1 - \exp\left(-\frac{k_t}{\eta |S_{t-1}|^\alpha}\right)\right) \tag{23}$$

Therefore we have

$$\log P(\mathbf{x}^* \in S_T) \geq \sum_{t=1}^{T} \log\left(1 - \exp\left(-\frac{k_t}{\eta |S_{t-1}|^\alpha}\right)\right) \tag{24}$$

We want to find the action sequence $\{k_t\}$ so that $\log P(\mathbf{x}^* \in S_T)$ is maximized. Let $w_t := \eta |S_{t-1}|^\alpha$ and $z_t := k_t / w_t$, applying Lemma 2, and we know that

$$\max_{\{k_t\}} \log P(\mathbf{x}^* \in S_T) \geq -\sum_{t=1}^{T} \log(1 + \lambda w_t) \tag{25}$$

where the Lagrangian multiplier λ satisfies the equation $g(\lambda) = K$.

Now we have:

$$\sum_{t=1}^{T} \log(1 + \lambda w_t) \overset{①}{\leq} \sum_{t=1}^{T} \log\left(1 + \frac{T}{K} w_t\right) \tag{26}$$

$$\overset{②}{\leq} \sum_{t=1}^{T} \log\left(1 + \frac{T}{K} \eta (N/2^{t-1})^\alpha\right) \tag{27}$$

$$\overset{③}{\leq} \frac{\eta T N^\alpha}{K} \sum_{t=1}^{T} \frac{1}{2^{\alpha(t-1)}} \tag{28}$$

$$= \phi(\alpha, T) \frac{\eta T N^\alpha}{K} \tag{29}$$

Here ① is due to Lemma 3 which tells that $\lambda = g^{-1}(K) \leq T/K$, ② is due to $w_t := \eta |S_{t-1}|^\alpha$ and $|S_{t-1}| \leq N/2^{t-1}$, and ③ due to $\log(1 + x) \leq x$.

Putting all of them together, we know that

$$r^* \geq \max_{\{k_t\}} \frac{1}{|S_T|} P(\mathbf{x}^* \in S_T) \geq \frac{2^T}{N} \exp\left(-\phi(\alpha, T) \frac{\eta T N^\alpha}{K}\right) \tag{30}$$

□

Optimal action sequence $\{k_t^*\}$. From the proof, we could also write down the optimal action sequence that achieves the best reward: $k_t^* = w_t \log\left(1 + \frac{1}{\lambda w_t}\right)$, where $w_t := \eta |S_{t-1}|^\alpha$. Using Lemma 3, we could compute the upper and lower bound estimation of $\lambda = g^{-1}(K)$. Here $\bar{w} := T\left(\sum_{t=1}^{T} w_t^{-1}\right)^{-1}$ be the Harmonic mean of $\{w_t\}$ and $w_{\max} := \max_{t=1}^{T} w_t$:

$$\frac{\bar{w}^{-1}}{\exp(\bar{w}^{-1} K/T) - 1} \leq \lambda \leq \frac{w_{\max}^{-1}}{\exp(w_{\max}^{-1} K/T) - 1} \tag{31}$$

With λ, we could compute approximate $\{k_t^*\}$. Here we make a rough estimation of $\{k_t^*\}$ if we terminate the algorithm when $|S_T|$ is still fairly large. This case corresponds to the setting $T =$

$\beta \log_2 N$ where $\beta < 1$ and all $w_t \sim N^\alpha$. With $K = \Theta(N^\alpha)$ as in semi-parametric case, $\bar{w}^{-1}K = \Theta(1)$, $\exp(\bar{w}^{-1}K/T) - 1 \approx \bar{w}^{-1}K/T$ and $\lambda w_t \sim \log_2 N \gg 1$. Since $\log(1+x) \approx x$ for small x, we have $k_t^* \approx w_t \frac{1}{\lambda w_t} = 1/\lambda$, which is independent of t. Therefore, a constant amount of sampling at each stage is approximately optimal.

Observation 1. *If all f_j are isotropic, $f_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{c}_j\|_2^2$, then $\Omega_P = \text{ConvexHull}(\mathbf{c}_1, \ldots, \mathbf{c}_M)$.*

Proof. Consider $J(\mathbf{x}; \mu) := \sum_{j=1}^M \mu_j f_j(\mathbf{x})$ where the weights $\mu_j \geq 0$ satisfies $\sum_j \mu_j = 1$. For brevity, we write the constraint as $\Delta := \{\mu : \mu_j \geq 0, \sum_j \mu_j = 1\}$.

Now consider the Pareto Set $\Omega_P := \{\mathbf{x} : \exists \mu \in \Delta : \nabla_{\mathbf{x}} J(\mathbf{x}; \mu) = 0\}$. We have the following:

$$\nabla_{\mathbf{x}} J(\mathbf{x}; \mu) = 0 \tag{32}$$

$$\iff \sum_j \mu_j \nabla_{\mathbf{x}} f_j(\mathbf{x}) = 0 \tag{33}$$

$$\iff \sum_j \mu_j (\mathbf{x} - \mathbf{c}_j) = 0 \tag{34}$$

$$\iff \mathbf{x} = \frac{\sum_j \mu_j \mathbf{c}_j}{\sum_j \mu_j} = \sum_j \mu_j \mathbf{c}_j \tag{35}$$

The last step is due to the fact that $\sum_j \mu_j = 1$. Therefore, for any $\mathbf{x} \in \Omega_P$, $\mathbf{x}$ is a convex combination of $\{\mathbf{c}_1, \ldots, \mathbf{c}_M\}$ and thus $\mathbf{x} \in \text{ConvexHull}(\mathbf{c}_1, \ldots, \mathbf{c}_M)$. Conversely, for any $\mathbf{x} \in \text{ConvexHull}(\mathbf{c}_1, \ldots, \mathbf{c}_M)$, we know $\nabla_{\mathbf{x}} J(\mathbf{x}; \mu) = 0$ and thus $\mathbf{x} \in \Omega_P$. □

Observation 2. *If $M = 2$ and $f_j(\mathbf{x}) = (\mathbf{x} - \mathbf{c}_j)^\top H_j (\mathbf{x} - \mathbf{c}_j)$ where H_j are positive definite symmetric matrices, then there exists $\mathbf{w}_1 := H_2(\mathbf{c}_2 - \mathbf{c}_1)$ and $\mathbf{w}_2 := H_1(\mathbf{c}_1 - \mathbf{c}_2)$, so that for any $\mathbf{x} \in \Omega_P$, $\mathbf{w}_1^\top (\mathbf{x} - \mathbf{c}_1) \geq 0$ and $\mathbf{w}_2^\top (\mathbf{x} - \mathbf{c}_2) \geq 0$.*

Proof. Following Observation 1, similarly we have for all $x \in \Omega_P$, $\sum_j \mu_j H_j (\mathbf{x} - \mathbf{c}_j) = 0$, which gives:

$$\mathbf{x} = \left(\sum_j \mu_j H_j \right)^{-1} \sum_j \mu_j H_j \mathbf{c}_j \tag{36}$$

Note that this is an expression of the Pareto Set Ω_P.

Let $A_j := (\sum_j \mu_j H_j)^{-1} \mu_j H_j$. Then $\sum_j A_j = I$. Note that while $\sum_j \mu_j H_j$ and $(\sum_j \mu_j H_j)^{-1}$ are positive definite matrix. A_j may not be.

Let $M := \sum_i \mu_i H_i$. Since $\mu \in \Delta$, M is a PD matrix. Note that we have

$$\sum_j \mu_j H_j \mathbf{c}_j = \sum_j \mu_j H_j \mathbf{c}_j - \sum_j \mu_j H_j \mathbf{c}_k + \sum_j \mu_j H_j \mathbf{c}_k \tag{37}$$

$$= \sum_{j \neq k} \mu_j H_j (\mathbf{c}_j - \mathbf{c}_k) + M \mathbf{c}_k \tag{38}$$

Using Eqn. 36, we know that $\mathbf{x} = M^{-1} \sum_j \mu_j H_j \mathbf{c}_j = \mathbf{c}_k + M^{-1} \sum_{j \neq k} \mu_j H_j (\mathbf{c}_j - \mathbf{c}_k)$.

For $M = 2$, we have $\mathbf{x} = \mathbf{c}_2 + M^{-1} \mu_1 H_1 (\mathbf{c}_1 - \mathbf{c}_2)$. So we have

$$(\mathbf{c}_1 - \mathbf{c}_2)^\top H_1 \mathbf{x} = (\mathbf{c}_1 - \mathbf{c}_2)^\top H_1 \mathbf{c}_2 + (\mathbf{c}_1 - \mathbf{c}_2)^\top H_1 M^{-1} H_1 (\mathbf{c}_1 - \mathbf{c}_2) \tag{39}$$

$$\geq (\mathbf{c}_1 - \mathbf{c}_2)^\top H_1 \mathbf{c}_2 \tag{40}$$

This is because $(\mathbf{c}_1 - \mathbf{c}_2) H_1 M^{-1} H_1 (\mathbf{c}_1 - \mathbf{c}_2) \geq 0$ since $H_1 M^{-1} H_1$ is a PSD matrix. Therefore, let $\mathbf{w}_2 := H_1(\mathbf{c}_1 - \mathbf{c}_2)$ and we have $\mathbf{w}_2^\top (\mathbf{x} - \mathbf{c}_2) \geq 0$, which is independent of $\mu \in \Delta$. This means it holds for any $\mathbf{x} \in \Omega_P$.

Let $\mathbf{w}_1 = H_2(\mathbf{c}_2 - \mathbf{c}_1)$, then similarly we have $\mathbf{w}_1^\top (\mathbf{x} - \mathbf{c}_1) \geq 0$ for all $\mathbf{x} \in \Omega_P$. □

B QUALITY INDICATORS COMPARISON

Table 1: The review of different scalarizing methods.

Quality Indicator	Convergence	Uniformity	Spread	No reference set required
HyperVolume	√	√	√	√
GD	√			
IGD	√	√	√	
MS			√	
S		√		
ONVGR	√			

Generational Distance(GD) (Van Veldhuizen & Lamont, 1998) measures the distance the pareto frontier of approximation samples and true pareto frontier, which requires prior knowledge of true pareto frontier and only convergence is considered. IGD (Bosman & Thierens, 2003) is improved version of GD. IGD calculates the distance the points on true pareto frontier to the closest point on pareto frontier of current samples. Inverted Generational Distance(IGD) satisfies all three evaluation metrics of QI but requires true pareto frontier which is hardly to get in real-world problem. Maximum Spread(MS) (Zitzler et al., 2000) computes the distance between the farthest two points of samples to evaluate the spread. Spacing(S) (Bandyopadhyay et al., 2004) measures how close the distribution of pareto frontier of samples is to uniform distribution. Overall Non-dominated Vector Generation and Ratio(ONVGR) is the ratio of number of samples in true pareto-frontier. The table 1 demonstrates the good characteristics of each quality indicators.

C END-TO-END LAMOO PSEUDOCODE

Below we list the pseudocode for the end-to-end workflow of LaMOO in Algorithm 2. Specfically, it includes search space partition in **Function Split**. Node(promising region) selection in **Function Select**, and new samples generation in **Function Sample**.

Algorithm 2 LaMOO Pseudocode.

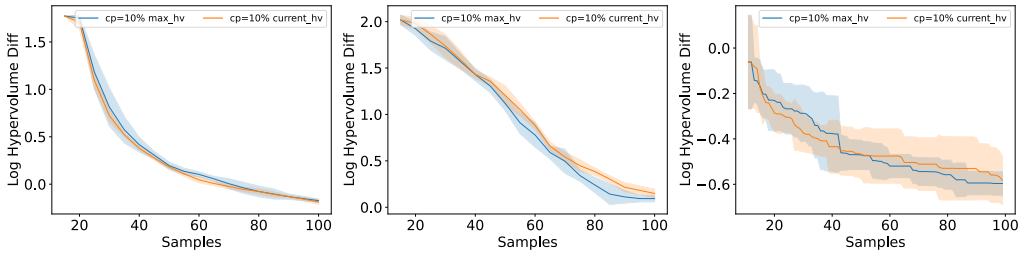
Inputs: Initial D_0 from uniform sampling, sample budget T.
for $t = 0, \ldots, T$ **do**
  Set $\mathcal{L} \leftarrow \{\Omega_{\text{root}}\}$ (collections of regions to be split).
  $\mathcal{V}, v, n \leftarrow$ **Split**($\mathcal{L}$, D_t)
  $k \leftarrow$ **Select**($\mathcal{C}_p$, D_t)
  $D_{t+1} \leftarrow$ **Sample**(k)
end for

Function Split($\mathcal{L}$, D_t)
  while $\mathcal{L} \neq \emptyset$ **do**
    $\Omega_j \leftarrow \text{pop\_first\_element}(\mathcal{L})$, $D_{t,j} \leftarrow D_t \cap \Omega_j$, $n_{t,j} \leftarrow |D_{t,j}|$.
    Compute dominance number $o_{t,j}$ of $D_{t,j}$ using Eqn. 5 and train SVM model $h(\cdot)$.
    If $(D_{t,j}, o_{t,j})$ is splittable by SVM, **then** $\mathcal{L} \leftarrow \mathcal{L} \cup \text{Partition}(\Omega_j, h(\cdot))$.
  end while

Function Select(C_p, D_t)
  for $k = \text{root}$, k is not leaf node **do**
    $D_{t,k} \leftarrow D_t \cap \Omega_k$, $v_{t,k} \leftarrow \text{HyperVolume}(D_{t,k})$, $n_{t,k} \leftarrow |D_{t,k}|$.
    $k \leftarrow \arg \max_{c \in \text{children}(k)} \text{UCB}_{t,c}$, where $\text{UCB}_{t,c} := v_{t,c} + 2C_p\sqrt{\frac{2\log(n_{t,k}}{n_{t,c}}}$
  end for
  return k

Function Sample(k)
  $D_{t+1} \leftarrow D_t \cup D_{\text{new}}$, where D_{new} is drawn from Ω_k based on qEHVI or CMA-ES.
  return $D_t \cup D_{new}$

D EXPLORATION FACTOR(C_p) SETUP WITH UNKNOWN MAXIMUM HYPERVOLUME

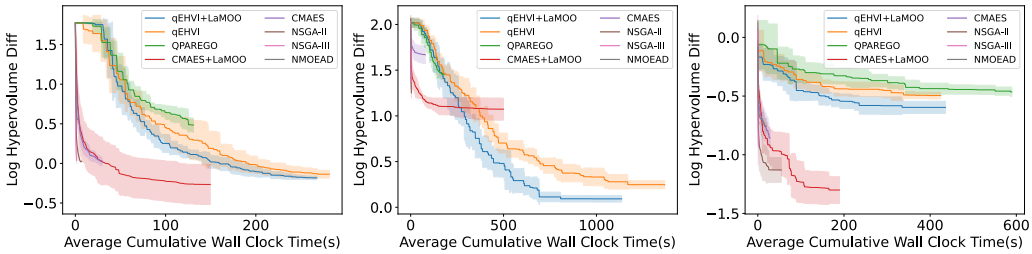


(a) BraninCurrin (b) VehicleSafety (c) Nasbench201

Figure 9: Sampling with static C_p(10% of HV_{max}) and dynamic C_p((10% of $HV_{current}$))

As we mentioned in the paper, a "rule of thumb" is to set the C_p to be roughly 10% of the maximum hypervolume HVmax. If HVmax is unknown, C_p can be dynamically set to 10% of the hypervolume of current samples in each search iteration. The figures below demonstrate the difference between 10% HVmax and 10% current hypervolume in three problems(Branin-Currin, VehicleSafety, and Nasbench201 from left to right). The final performances by using 10% HVmax and 10% current hypervolume are similar.

E WALL CLOCK TIME IN DIFFERENT PROBLEMS



(a) BraninCurrin (b) VehicleSafety (c) Nasbench201

Figure 10: Wall clock time in different problems

Fig. 10 shows the wall clock time of different search algorithms in BraninCurrin(Belakaria et al., 2019), VehicleSafefy (Liao et al., 2008) and Nasbench201 (Dong & Yang, 2020).

F DETAILS OF BENCHMARK PROBLEMS

F.1 PROBLEM DESCRIPTION

BraninCurrin (Belakaria et al., 2019):

$$f^{(1)}(x_1, x_2) = (15x_2 - \frac{5.1 * (15x_1 - 5)^2}{4\pi^2} + \frac{75x_1 - 25}{\pi} - 5)^2 + (10 - \frac{10}{8\pi})) * \cos(15x_1 - 5))$$

$$f^{(2)}(x_1, x_2) = \left[1 - \exp\left(-\frac{1}{(2x_2)}\right)\right] \frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}$$

where $x_1, x_2 \in [0, 1]$.

VehicleSafefy (Liao et al., 2008):

$$
\begin{aligned}
f_1(\boldsymbol{x}) &= 1640.2823 + 2.3573285x_1 + 2.3220035x_2 + 4.5688768x_3 + 7.7213633x_4 + 4.4559504x_5 \\
f_2(\boldsymbol{x}) &= 6.5856 + 1.15x_1 - 1.0427x_2 + 0.9738x_3 + 0.8364x_4 - 0.3695x_1x_4 + 0.0861x_1x_5 \\
&\quad + 0.3628x_2x_4 + 0.1106x_1^2 - 0.3437x_3^2 + 0.1764x_4^2 \\
f_3(\boldsymbol{x}) &= -0.0551 + 0.0181x_1 + 0.1024x_2 + 0.0421x_3 - 0.0073x_1x_2 + 0.024x_2x_3 - 0.0118x_2x_4 \\
&\quad - 0.0204x_3x_4 - 0.008x_3x_5 - 0.0241x_2^2 + 0.0109x_4^2
\end{aligned}
$$

where $\boldsymbol{x} \in [1, 3]^5$.

Nasbench201 (Dong & Yang, 2020):

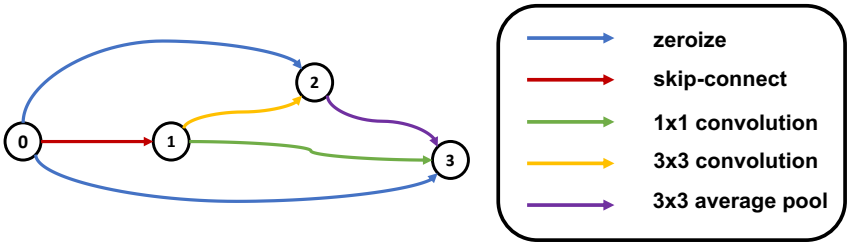


Figure 11: A general architecture of Nasbench201

In Nasbench201, the architectures are made by stacking the cells together. The difference among architectures in Nasbench201 is the design of the cells, see fig 11. Specifically, each cell contains 4 nodes, and there is a particular operation connecting to two nodes including zeroize, skip-connect, 1x1 convolution, 3x3 convolution, and 3x3 average pooling. Therefore, there are C_4^2 = 6 edges in a cell and 5^6 =15625 unique architectures in Nasbench201. According to this background, Each architecture can be encoded into a 6-dimensional vector with 5 discrete numbers (i.e., 0, 1, 2, 3, 4 that corresponds to zeroize, skip-connect, 1x1 convolution, 3x3 convolution, and 3x3 average pooling).

$$
\begin{aligned}
f_1(\boldsymbol{x}) &= Accuracy(\boldsymbol{x}) \\
f_2(\boldsymbol{x}) &= \#FLOPs(\boldsymbol{x})
\end{aligned}
$$

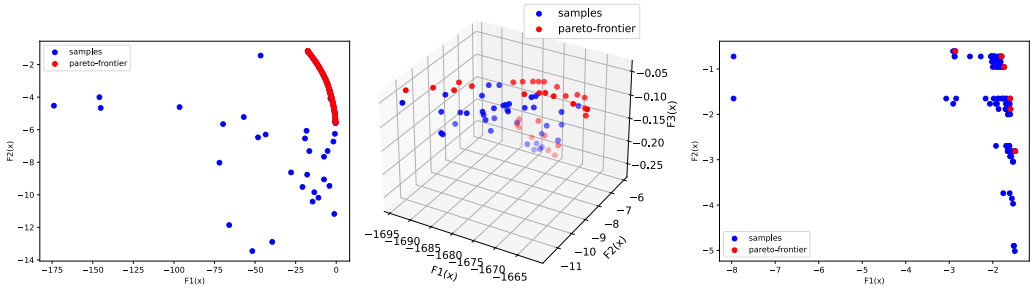
where $\boldsymbol{x} \in \{0, 1, 2, 3, 4\}^6$.

DTLZ2 (Deb et al., 2002b):

$$
\begin{aligned}
f_1(\boldsymbol{x}) &= (1 + g(\boldsymbol{x}_M)) \cos\left(\frac{\pi}{2}x_1\right) \cdots \cos\left(\frac{\pi}{2}x_{M-2}\right) \cos\left(\frac{\pi}{2}x_{M-1}\right) \\
f_2(\boldsymbol{x}) &= (1 + g(\boldsymbol{x}_M)) \cos\left(\frac{\pi}{2}x_1\right) \cdots \cos\left(\frac{\pi}{2}x_{M-2}\right) \sin\left(\frac{\pi}{2}x_{M-1}\right) \\
f_3(\boldsymbol{x}) &= (1 + g(\boldsymbol{x}_M)) \cos\left(\frac{\pi}{2}x_1\right) \cdots \sin\left(\frac{\pi}{2}x_{M-2}\right) \\
&\vdots \\
f_M(\boldsymbol{x}) &= (1 + g(\boldsymbol{x}_M)) \sin\left(\frac{\pi}{2}x_1\right)
\end{aligned}
$$

where $g(\boldsymbol{x}) = \sum_{x_i \in \boldsymbol{x}_M} (x_i - 0.5)^2$, $\boldsymbol{x} \in [0, 1]^d$, and $\boldsymbol{x}_M$ represents the last $d - M + 1$ elements of $\boldsymbol{x}$.

F.2 VISUALIZATION OF PARETO-FRONTIER FOR BENCHMARK PROBLEMS



(a) BraninCurrin (b) VehicleSafety (c) Nasbench201

Figure 12: Visualization of Pareto-frontier in BraninCurrin, VehicleSafety as well as Nasbench201.

F.3 REFERENCE POINTS

†: M represents the number of objectives.

Problem	Reference Point
BraninCurrin	(18.0, 6.0)
VehicleSafety	(1864.72022, 11.81993945, 0.2903999384)
Nasbench201	(-3.0, -6.0)
DTLZ2	(1.1, .., 1.1) $\in \mathbb{R}^{\mathbb{M}^\dagger}$
Molucule Discovery	(0.0, ..., 0.0) $\in \mathbb{R}^{\mathbb{M}^\dagger}$

Table 2: The reference points for all problems in this work.

Table 2 elaborates the reference points in the problems throughout the paper.

F.4 MAXIMUM HYPERVOLUME OF EACH PROBLEM

Problem	Maximum Hypervolume
BraninCurrin	59.36011874867746
VehicleSafety	246.81607081187002
Nasbench201	8.06987476348877
DTLZ2(2 objectives)	1.4460165933151778
DTLZ2(10 objectives)	2.5912520655298095
Molucule Discovery	N/A

Table 3: The maximum hypervolume for all problems in this work.

Table 3 elaborates the observed maximal hypervolume in the problems throughout the paper. We used these value to calculate the log hypervolume difference in fig 3 and fig 4.

G EXPERIMENTS SETUP

Experiment details: For small-scale problems(i.e. Branin-Currin, VehicleSafety, and Nasbench201) and DTLZ2 with 2 and 10 objectives. We randomly generate 10 samples as the initialization. For multi-objective molecule discovery, the number of initial samples is 150. In each iteration, we update 5 batched samples(q value) for all search algorithms.

Hyperparameters of LAMOO: For all problems, we leverage polynomials as the kernel type of SVM and the degree of the polynomial kernel function is set to 4. The minimum samples in the leaf of MCTS is 10. The cp is roughly set to 10% of maximum of hypervolume(i.e. Branin-Currin -> 5, VehicleSafety -> 20, Nasbench201 -> 6, DTLZ2(2 objectives) -> 0.1, DTLZ2(10 objectives) -> 0.25, molecule discovery(2 objectives) -> 0.03, molecule discovery(3 objectives) -> 0.2, molecule discovery(4 objectives) -> 0.06).

Hyperparameters of qEHVI and qParEGO: The number of q is set to 5. The acquisition function is optimized with L-BFGS-B (with a maximum of 200 iterations). In each iteration, 256 raw samples used for the initialization heuristic are generated to be selected by the acquisition function. As the same claim in Daulton et al. (2020), each generated sample is modeled with an independent Gaussian process with a Matern 5/2 ARD kernel.

H VERIFICATION OF LAMOO ON MANY-OBJECTIVE PROBLEMS

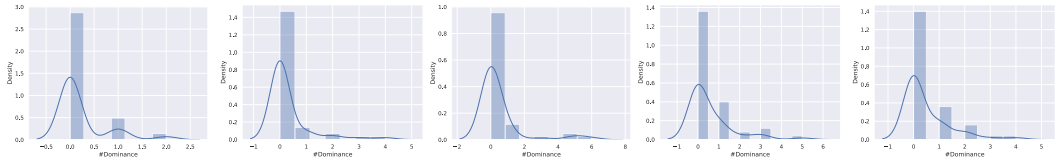


Figure 13: Dominance number distribution with 50 random samples on DTLZ2(10 objectives)

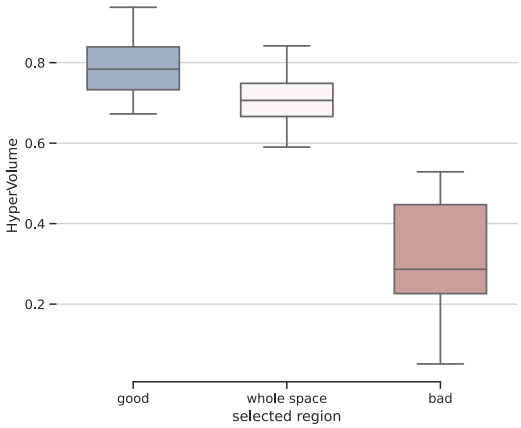


Figure 14: The range of hypervolume for 50 samples randomly generated from different regions in DTLZ2(10 objectives). We generate 25 times of 50 samples in total.

While it is theoretically hard to label the samples into good and bad based on their dominance number in many-objective problems due the the lack of dominance pressure(All samples are non-dominated with each other). If number of objective is not too large(i.e. ≤ 10), the samples can be still split by dominance number. Given the problem(DTLZ2 with 10 objectives) shown in fig 4, we randomly generate 50 samples in the search space and draw the dominance distribution of them(see fig 13). We did this experiment 5 times.

We then partition the search space by a SVM classifier based on the labeled samples into "good" and "bad", and randomly generate 50 samples in "good region", "bad region", and whole space,

respectively. We did this process 5 times with different initial samples. Fig. 14 shows the range of hypervolume of the samples generated from good regions, the whole space, and bad regions. From the figure, we can see that the hypervolume of samples generated from good regions are significantly higher than others.

I COMPUTATIONAL COMPLEXITY ANALYSIS OF LAMOO

Here is a detailed breakdown of the computational complexity of our algorithm (Alg. 1)

Line.6: Compute dominance: $O(D{N_{node}}^2)$ where N_{node} is the number of samples in the node and D is the number of dimensions.

Line 7: Time complexity of SVM : $O(N^2)$ where N_{node} is number of searched samples in total.

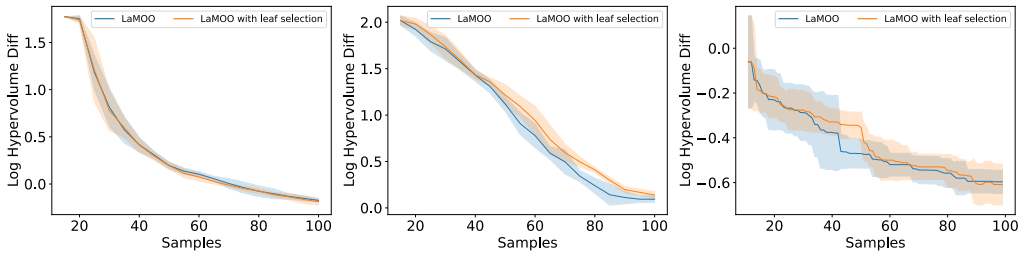
Line 10: Hypervolume: $O(N^{\frac{D}{2}} + N \log N)(D > 3)$ (Beume & Rudolph, 2006) or $O(N \log N)(D \leq 3)$ (Beume et al., 2009), where N is number of searched samples in total and D is the number of dimensions.

HV computation is to be the dominating factor in case of $D > 3$, otherwise, it should be the computation of SVM.

J VARIATION OF LAMOO WITH A CHEAPER OVERHEAD

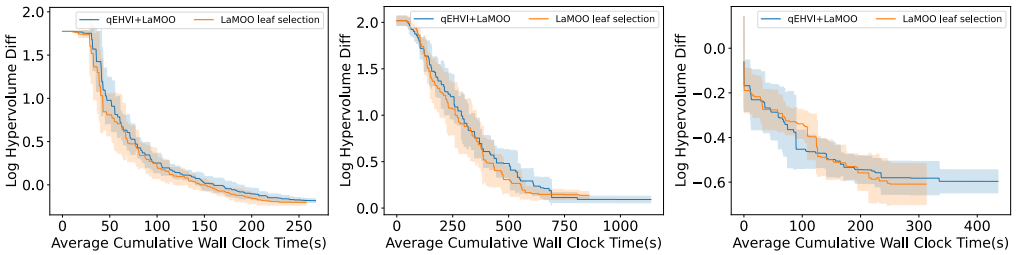
Algorithm 3 LaMOO Pseudocode with leaf based selection.

Inputs: Initial D_0 from uniform sampling, sample budget T.
for $t = 0, \ldots, T$ **do**
  Set $\mathcal{L} \leftarrow \{\Omega_{\text{root}}\}$ (collections of regions to be split).
  while $\mathcal{L} \neq \emptyset$ **do**
    $\Omega_j \leftarrow \text{pop\_first\_element}(\mathcal{L})$, $D_{t,j} \leftarrow D_t \cap \Omega_j$, $n_{t,j} \leftarrow |D_{t,j}|$.
    Compute dominance number $o_{t,j}$ of $D_{t,j}$ using Eqn. 5 and train SVM model $h(\cdot)$.
    If $(D_{t,j}, o_{t,j})$ is splittable by SVM, **then** $\mathcal{L} \leftarrow \mathcal{L} \cup \text{Partition}(\Omega_j, h(\cdot))$.
  end while
  for $k = \text{root}$, k is not leaf node **do**
    $D_{t,k} \leftarrow D_t \cap \Omega_k$, $n_{t,k} \leftarrow |D_{t,k}|$.
  end for
  for l is leaf node **do**
    $v_{t,l} \leftarrow \text{HyperVolume}(D_{t,l})$
  end for
  $k \leftarrow \arg \max_{l \in \text{leaf nodes}} \text{UCB}_{t,l}$, where $\text{UCB}_{t,l} := v_{t,l} + 2C_p\sqrt{\frac{2\log(n_{t,l})}{n_{t,p}}}$, where p is the parent of l.
  $D_{t+1} \leftarrow D_t \cup D_{\text{new}}$, where D_{new} is drawn from Ω_k based on qEHVI or CMA-ES.
end for



(a) BraninCurrin (b) VehicleSafety (c) Nasbench201

Figure 15: Search progress with sample



(a) BraninCurrin (b) VehicleSafety (c) Nasbench201

Figure 16: Search progress with time

Instead of traversing down the search tree to trace the current most promising search path, this variation of LaMOO directly select the leaf node with the highest *UCB value*. Algorithm. 3 illustrates the detail of this variation. Therefore, this variation avoids calculating the hypervolume in the non-leaf nodes of the tree, where hypervolume calculation is the main computational cost of LaMOO especially in many-objective problems. Figure. 15 and figure. 16 validate the variation that is able to reach similar performance of searched samples but saves lots of time. We leave the validation of others problems in the future works.