Constrained Discrete Black-Box Optimization using Mixed-Integer Programming

T. P. Papalexopoulos, C. Tjandraatmadja, R. Anderson, J. P. Vielma, D. Belanger

Alexandre Schulz

LIPN

May 16, 2023

Table of Contents

Table of Contents

- solving Black-box optimization problems with **Model based optimiation** (MBO)
- Issues:
  - Ω combinatorial structure, contraints
  - expensive evaluation of f, no gradient information
- Applications:
  - neural architecture search, *Zoph & Le, 2017*
  - program synthesis, *Summers, 1977, Biermann, 1978*
  - small-molecule design, *Elton et al., 2019*
  - protein design, *Yang et al., 2019*

- **Objective**: $x^* = \arg\max_{\Omega} f(x)$
- iteratively refines an approximator $\hat{f} \approx f$
- selects new **query points** by solving an **Inner-loop** optmization problem:

$$x_t = \arg\max_{\Omega_t} a(x)$$

- **Acquisition function**: $a : \Omega \rightarrow \mathbb{R}$
  - derived from point evaluation or from posterior distribution over $\hat{f}$
  - easier to solve
  - "white-box" caracteristics

MBO has two issues :

- solving the **Inner-loop** may be difficult
- Real world application require additional constraints on x

Using **Heuristic Inner-loop solvers** is a solution. Requires domain knowledge.

Authors remark

Crucially, by framing the inner-loop optimization as an MILP, our approach can flexibly incorporate a wide variety of logical, combinatorial, and polyhedral constraints on the domain, which need only be provided in a declarative sense.

- **NN+MILP**: MBO framework for *discrete optimization* with NN surrogates and with **exact** inner-loop guarantees
- Show that NN+MILP matches and surpasses MBO baseline with domain specific evolutionary algorithms
- Experimental benchmarking results : *MINLPLib, NAS-Bench-101 neural architecture*

Using **Heuristic Inner-loop solvers** is a solution. Requires domain knowledge.

Table of Contents

Model Based Optimization

Baseline algorithm

Algorithm 1 MBO

Input: hypothesis class $\mathcal{F}$, budget N, initial dataset $\mathcal{D}_n = \{x_i, f(x_i)\}_{i=1}^n$, optimization domain Ω

for $t = n + 1$ to $t = N$ **do**

  $P(\hat{f}_t) \leftarrow \texttt{fit}(\mathcal{F}, \mathcal{D}_{t-1})$

  $a(x) \leftarrow \texttt{get\_acquisition\_function}(P(\hat{f}_t))$

  $x_t \leftarrow \texttt{inner\_loop\_solver}(a(x), \Omega)$

  $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{x_t, f(x_t)\}$

end for

return $\arg\max_{(x_t, y_t) \in \mathcal{D}_N} y_t$

Figure: MBO baseline algorithm

- 1. perform inference to approximate f
- 2. define $a(x)$ based on $\hat{f}_t(x)$ quantifying the **quality of points to query**
- 3. x_t selected by solving the **inner-loop problem**

Algorithm 2 NN+MILP

Input: hypothesis class $\mathcal{F}$, budget N, initial dataset $\mathcal{D}_n = \{x_i, f(x_i)\}_{i=1}^n$, MILP domain formulation $\mathcal{M}_\Omega$

for $t = n + 1$ to $t = N$ **do**

  $\hat{f}_t \leftarrow \texttt{fit}(\mathcal{F}, \mathcal{D}_{t-1})$ (3.2)

  $\mathcal{M}_t \leftarrow \texttt{build\_milp}(\hat{f}_t,\ \mathcal{M}_\Omega,\ \mathcal{D}_{t-1})$ (3.3)

  $x_t \leftarrow \texttt{optimize}(\mathcal{M}_t)$ (generic MILP solver)

  $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{x_t, f(x_t)\}$

end for

return $\arg\max_{(x_t, y_t) \in \mathcal{D}_N} y_t$

Figure: NN+MILP algorithm

Table of Contents

Goal: find $x^* = \arg\max_{\Omega} f(x)$ where f is an **expensive, noiseless black-box function with n decision variables**.

- N: fixed budget of queries to f
- $\mathcal{X}_t = \{x_i\}_{i=1}^t$: set of sampled points at step t
- $\mathcal{D}_t = \{x_i, y_i = f(x_i)\}_{i=1}^t$: set of sampled points with corresponding reward

At iteration t solving the **acquisition problem** is finding :

$$x_t = \arg\max_{x \in \Omega \setminus \mathcal{X}_{t-1}} \hat{f}_t(x)$$

- $\hat{f}$ chosen as feedforward neural network with piecewise-linear activation functions.
- fully connected layers with ReLU activation
- compatible with convolution and max-pooling

At each iteration $\hat{f}_t$ is trained from scratch with random weight initialization and SGD. $\hat{f}_t$ is trained on $\mathcal{D}_{t-1}$ with L^2 loss.
The acquisition is taken to be $a(x) = \hat{f}_t(x)$.

The inner-loop optimization problem $\mathcal{M}_t$ is :

$$x_t = \arg\max_{x \in \Omega \setminus \mathcal{X}_{t-1}} \hat{f}_t(x)$$

Domain

If not already binary, decision variables are **one-hot encoded**:

$$z_{ij} = \mathbb{I}[x_i = j], \quad i \in [n], j \in \Omega_i$$

subject to the constraints : $\sum_{j \in \Omega_i} z_{ij} = 1, \quad \forall i$

Problem specific constraints can be added as needed.

No-Good Constraints

Leverage the binary nature of z to eliminate $\mathcal{X}_{t-1}$ from $\mathcal{M}_t$.
Consider $\overline{x} \in \Omega$ a point we wish to exclude and $\overline{z}$ its one-hot encoding. Then the constraint

$$\sum_{i,j:\overline{z}_{ij}=0} z_{ij} + \sum_{i,j:\overline{z}_{ij}=1} (1 - z_{ij}) \geq 1$$

ensures that candidate has Hamming distance of at least one to $\overline{z}$.
Note that this formulation will not work for continuous x.

Neural Network

" *The overall MILP objective is the activation corresponding to the regressor's output neuron.*"

Let $y = \max(0, w^T x + b)$ be the output of a single layer with weights w and bias b.

At optimization time w, b are fixed and x, y are the decision variables.

Non-linearity: α binary decision variable indicating the ReLU is activated. We add the constraints

$$0 \leq y \leq M\alpha \tag{1}$$

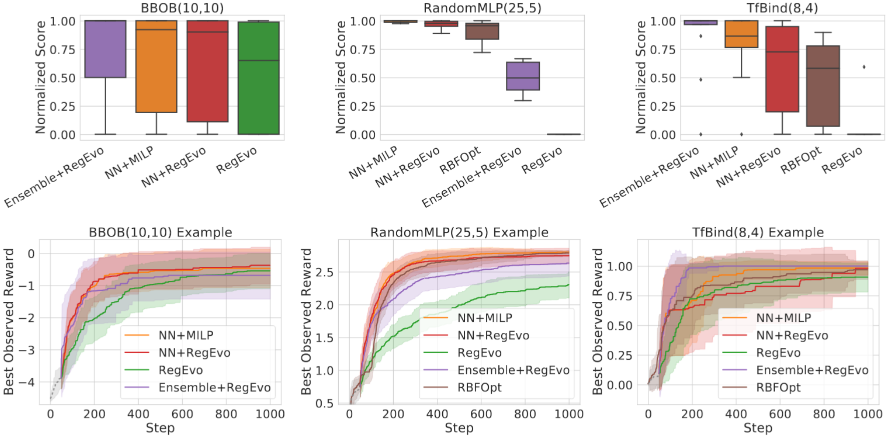
$$w^T x + b \leq y \leq w^T x + b + M(1 - \alpha) \tag{2}$$

where M is a large constant (such as upper bound on range of y).

Table of Contents

- **RandomMLP** The output of a multi-layer perceptron operating on a one-hot encoding of the input.
- **TfBind** Binding strength of a length-8 DNA sequence to a given transcription factor (Barrera et al., 2016).
- **BBOB** Non-linear function from the continuous Black- Box Optimization Benchmarking library (Hansen et al., 2009)
- **Ising** The negative energy of fully-connected binary Ising Model with normally distributed pairwise potentials.

- **RegEvo** Local evolutionary search (Real et al., 2019).
- **NN + RegEvo** An ablation of NN+MILP, with the only difference being the use of **RegEvo** in lieu of MILP for solving the acquisition problem.
- **Ensemble + RegEvo** A re-implementation of the 'MBO' baseline from *Angermueller et al. (2020)*, using an ensemble of linear and random forest regressors as the surrogate.
- **RBFOpt** A competitive mixed-integer black-box optimization solver that uses the 'Radial Basis Function method' as a surrogate model (*Costa Nannicini, 2018*).



BBOB(10,10)
Normalized Score
1.00
0.75
0.50
0.25
0.00
Ensemble+RegEvo
NN+MILP
NN+RegEvo
RegEvo
RandomMLP(25,5)
NN+MILP
NN+RegEvo
RBFOpt
Ensemble+RegEvo
RegEvo
TfBind(8,4)
BBOB(10,10) Example
Best Observed Reward
Step
RandomMLP(25,5) Example
TfBind(8,4) Example

Constrained Discrete Black-Box Optimization using MILP

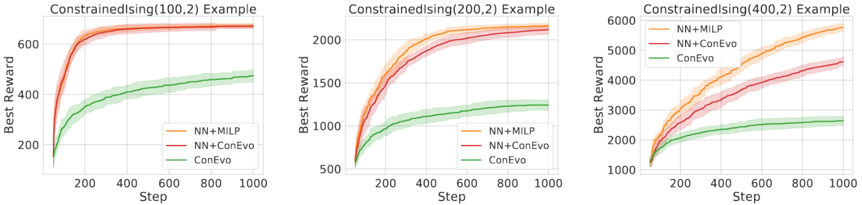


Figure 2: Best observed reward as a function of iteration for an example constrained problem (Section 4.3) for each of n = 100, 200, and 400 (left-to-right). Lines and bands indicate the average and ± 1 sd respectively, over 20 trials for $n = 100$ and 10 trials for the rest. Distribution of normalized final scores and more examples can be found in Appendix E.2

.

Thank You for Your Attention!