Scalable Combinatorial Bayesian Optimization with Tractable Statistical models

Abstract

We study the problem of optimizing expensive blackbox functions over combinatorial spaces (e.g., sets, sequences, trees, and graphs). BOCS [2] is a state-of-the-art Bayesian optimization method for tractable statistical models, which performs semi-definite programming based acquisition function optimization (AFO) to select the next structure for evaluation. Unfortunately, BOCS scales poorly for large number of binary and/or categorical variables. Based on recent advances in submodular relaxation [22] for solving Binary Quadratic Programs, we study an approach referred as *Parametrized Submodular Relaxation (PSR)* towards the goal of improving the scalability and accuracy of solving AFO problems for BOCS model. Experiments on diverse benchmark problems including real-world applications in communications engineering and electronic design automation show significant improvements with PSR for BOCS model.

1 Introduction

Many real-world science and engineering applications involve optimizing combinatorial spaces (e.g., sets, sequences, trees, and graphs) using expensive black-box evaluations. For example, in hardware design optimization, we need to appropriately place the processing elements and communication links for achieving high performance guided by expensive computational simulations to emulate the real hardware. Bayesian optimization (BO) [29] is a popular framework for solving expensive black-box optimization problem. BO framework consists of three key elements: *1) Statistical model* (e.g., Gaussian Process) learned from past function evaluations; *2) Acquisition function (AF)* (e.g., expected improvement) to score the potential utility of evaluating an input based on the statistical model; and *3) Acquisition function optimization (AFO)* to select the best candidate input for evaluation. In each BO iteration, the selected input is evaluated and the statistical model is updated using the aggregate training data. Most of the prior work on BO is focused on optimizing continuous spaces. There are two key challenges to extend BO framework to combinatorial spaces. First, defining a surrogate statistical model over combinatorial objects. Second, search through the combinatorial space to select the next structure for evaluation given such a statistical model.

Prior work on combinatorial BO has addressed these two challenges as follows. SMAC [20, 21] is a canonical baseline that employs complex statistical model in the form of random forest and executes a *hand-designed* local search procedure for optimizing the acquisition function. A recent work referred as COMBO [25] proposed a novel combinatorial graph representation for discrete spaces which allows using Gaussian process with diffusion kernels. Reduction to continuous BO [15] employs an encoder-decoder architecture to learn continuous representation of combinatorial objects from data and performs BO in this latent space. Unfortunately, this approach requires a *large* dataset of combinatorial objects, for learning the latent space, which is impossible to acquire for many real-world applications. BOCS [2] is another method that employs a tractable statistical model defined over binary variables and Thompson sampling as the acquisition function. These two choices within BOCS leads to a semi-definite programming (SDP) based solution for solving AFO problems. Unfortunately, BOCS approach scales poorly for large number of binary variables and for categorical variables due to one-hot encoding representation.

Workshop on machine learning for engineering modeling, simulation and design @ NeurIPS 2020

Our work is inspired by the success of submodular relaxation based inference methods in the structured prediction literature [23, 33, 13, 16]. In this paper, we employ the submodular relaxation based Binary Quadratic optimization approach proposed in [22] to improve the computational-efficiency and accuracy of solving AFO problems for BOCS model. We refer to this approach as *Parametrized Submodular Relaxation (PSR)* algorithm. First, we reformulate the AFO problem as submodular relaxation with some parameters. This relaxed problem can be solved efficiently using minimum graph cut algorithms [23, 5]. The accuracy of this relaxed problem critically depends on the unknown parameters. Therefore, we solve an outer optimization problem to find the values of unknown parameters with close approximation to the true objective. *To the best of our knowledge, this is the first application of submodular relaxation to solve combinatorial BO problems*. Experimental results on real-world benchmarks show the efficacy of PSR to improve the state-of-the-art on combinatorial BO with tractable statistical models in terms of both computational-efficiency and accuracy.

Contributions. The main contributions of this paper are: **1)** By leveraging the recent advances in submodular relaxation, we study the parametrized submodular relaxation approach to improve the scalability and accuracy of solving AFO problems for BOCS, the state-of-the-method for tractable statistical models. **2)** We perform comprehensive experiments on real-world benchmarks to show computational-efficiency and accuracy improvements over existing BOCS method.

2 Problem Setup and Challenges

We are given a combinatorial space of structures $\mathcal{X}$ (e.g., sets, sequences, trees, graphs). Without loss of generality, let each combinatorial structure $\mathbf{x} \in \mathcal{X}$ be represented using n discrete variables $x_1, x_2, \cdots, x_n$, where each variable x_i takes k candidate values from the set $\mathcal{CV}(x_i)$. For binary variables, k equals 2 and for categorical variables, k is greater than 2. We assume the availability of an unknown objective function $\mathcal{F} : \mathcal{X} \mapsto \Re$ to evaluate each combinatorial object $\mathbf{x} \in \mathcal{X}$. Each evaluation is expensive and results in outcome $y = \mathcal{F}(\mathbf{x})$. For example, in hardware design optimization, $\mathbf{x}$ is a graph corresponding to the placement of processing elements and communication links, and $\mathcal{F}(\mathbf{x})$ corresponds to an expensive computational simulation. The overall goal is to minimize the number of objective function evaluations to uncover a structure $\mathbf{x} \in \mathcal{X}$ that approximately optimizes $\mathcal{F}$. We consider minimizing the objective $\mathcal{F}$ for the sake of technical exposition and consistent notation.

We now briefly explain the BOCS method [2] that we intend to improve on. Full discussion of related work is provided in the Appendix A.1.

BOCS Approach. BOCS instantiates the three key elements of BO framework as follows. *1) Surrogate statistical model:* A linear Bayesian model defined over binary variables is employed as the surrogate model. The model is described as:

$$f_\alpha(\mathbf{x} \in \mathcal{X}) = \alpha_0 + \sum_j \alpha_j x_j + \sum_{i,j>i} \alpha_{ij} x_i x_j \tag{2.1}$$

where $\mathcal{X} = \{0,1\}^n$ and $\mathbf{x} \in \mathcal{X}$ is a binary vector and α variables are drawn from a sparsity-inducing horseshoe prior [6]. It was experimentally found that the above second-order model provides an excellent trade-off between expressiveness and accuracy. The α variables quantify the uncertainty of the model. *2) Acquisition function:* Thompson sampling [28] is employed as the acquisition function because of its proven theoretical and empirical properties in the context of BO. *3) Acquisition function optimization:* In each BO iteration, we select a candidate structure $\mathbf{x} \in \mathcal{X}$ for evaluation that minimizes the acquisition function. In the case of BOCS method, the acquisition function optimization (AFO) problem becomes: $\arg\min_{\mathbf{x}\in\mathcal{X}} f_\alpha(\mathbf{x}) + \lambda P(\mathbf{x})$, where $\lambda P(\mathbf{x})$ being a regularization term commonly seen in multiple applications. BOCS employs a semi-definite programming (SDP) based relaxation approach to solve the above AFO problem.

Scalability Challenges of BOCS. There are multiple challenges associated with SDP approach used for solving AFO problems in BOCS formulation. First, the time complexity of a standard SDP solver grows at the rate of $O(n^6)$ [34, 22], which is prohibitive for large dimensions. Second, the approximation error for SDP based solution is known to be at most $O(\log n)$ [2, 8], which clearly grows as the dimensions increase, resulting in the loss of accuracy as well. These scaling issues arise when the number of binary variables are large especially since categorical variables are represented by one-hot encoding in BOCS. Our goal in this paper is to provide an algorithmic approach to improve the computational-efficiency and accuracy of solving AFO problems for BOCS method.

3 Parametrized Submodular Relaxation

3.1 High-level Overview of PSR Algorithm

The overall idea of using submodular relaxations for optimizing BQP problems is extensively employed in computer vision [17]. However, we employ the concrete instantiation of this general framework as proposed in the context of prescriptive price optimization [22]. *To the best of our knowledge, this is the first application of submodular relaxation concepts to solve combinatorial BO problems.* Recall that AFO problem for BOCS method is:

$$\arg\min_{\mathbf{x}\in\mathcal{X}} f_\alpha(\mathbf{x}) + \lambda P(\mathbf{x}) \tag{3.1}$$

where $\lambda P(\mathbf{x})$ is a regularization term commonly seen in multiple applications. For example, by choosing $P(\mathbf{x}) = \|\mathbf{x}\|_1$, the optimization problem in 3.1 becomes a Binary Quadratic Program (BQP) as given below:

$$\arg\min_{\mathbf{x}\in\mathcal{X}} \quad \alpha_0 + \sum_j (\alpha_j + \lambda) x_j + \sum_{i,j>i} \alpha_{ij} x_i x_j \tag{3.2}$$

$$\arg\min_{\mathbf{x}\in\mathcal{X}} \quad \mathbf{x}^T A \mathbf{x} + b^T \mathbf{x} \tag{3.3}$$

In the general case, BQP is NP-hard [8, 14]. We propose using an efficient submodular relaxation with some unknown parameters (matrix Λ) for solving this problem. A key advantage of this relaxation is that it allows us to leverage minimum graph cut algorithms to efficiently solve it. The accuracy of this relaxed problem critically depends on the unknown parameters Λ. Therefore, we can utilize an outer optimization problem to find the values of unknown parameters with close approximation to the true acquisition function. We solve this optimization problem using an iterative algorithm (steps 5-9 in Algorithm 1). We perform two algorithmic steps in each iteration. First, we solve the parametrized submodular relaxation of the AFO problem using a minimum graph cut algorithm (step 7). Second, we update the values of unknown parameters Λ using proximal gradient descent (step 8). Algorithm 1 provides the complete pseudo-code of combinatorial BO using PSR algorithm.

Advantages of PSR algorithm. When compared to the semi-definite programming (SDP) relaxation approach to solve AFO problems in BOCS, PSR algorithm has significant advantages in terms of both computational-efficiency and accuracy of solving AFO problems. First, PSR relies on a small number of calls (five to ten iterations based on our experiments) to a minimum graph cut solver, which has relatively very low time-complexity (e.g. $\mathcal{O}(n^3)$ for preflow-push algorithms or $\mathcal{O}(n^3 \log n)$ for Dinic's algorithm [1]) which is significantly better than $\mathcal{O}(n^6)$ for SDP approach. Second, our experiments show that PSR algorithm significantly improves the accuracy over SDP approach with increased dimensionality.

3.2 Key Algorithmic Steps

The two main algorithmic steps of PSR algorithm are: *submodularization of the objective with unknown parameters* and *finding optimized parameters to improve the accuracy of relaxation*. The binary quadratic objective function ($\mathbf{x}^T A \mathbf{x} + b^T \mathbf{x}$ in 3.3) is called as *submodular* if all the elements of the matrix A are non-positive, i.e., $a_{ij} \in A \leq 0 \quad \forall i, j$. However, in our setting, it is not necessary that the BQP objective in Equation 3.3 follows the submodularity property. As mentioned earlier, we approximate the objective by constructing a submodular relaxation in the same way as [22]. This relaxation is parametrized by a matrix Λ, which is directly related to the quality of approximation.

Consider the objective in 3.3 written as a sum of two terms decomposed over the positive (A^+) and non-positive (A^-) terms of matrix A:

$$\mathbf{x}^T A \mathbf{x} + b^T \mathbf{x} = \mathbf{x}^T A^+ \mathbf{x} + \mathbf{x}^T A^- \mathbf{x} + b^T \mathbf{x} \tag{3.4}$$

where $A^+ + A^- = A$ and A^+ and A^- are defined as follows:

$$A^+ = \begin{cases} a_{ij} & \text{if } a_{ij} > 0 \\ 0 & \text{if } a_{ij} \leq 0 \end{cases} \forall a_{ij} \in A, \quad A^- = \begin{cases} a_{ij} & \text{if } a_{ij} \leq 0 \\ 0 & \text{if } a_{ij} > 0 \end{cases} \forall a_{ij} \in A$$

The second term in Equation 3.4 ($\mathbf{x}^T A^- \mathbf{x} + b^T \mathbf{x}$) is a submodular function. Similar to the strategy employed for prescriptive price optimization [22], we construct a submodular relaxation of the first

Algorithm 1 Combinatorial BO via PSR Algorithm

Input: $\mathcal{X}$ = Discrete space, $\mathcal{F}(\mathbf{x})$ = expensive objective function, statistical model $f_\alpha(\mathbf{x} \in \mathcal{X})$
Output: $(\mathbf{x}_{best}, \mathcal{F}(\mathbf{x}_{best}))$, the best uncovered input x_{best} with its function value

Initialize statistical model f_α with a small number of input-output examples; and $t \leftarrow 0$
repeat
  Sample α from posterior of f_α
  Compute the next input to evaluate via acquisition function optimization:
  $\mathbf{x}_{t+1} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{X}} \; AF(f_\alpha, \mathbf{x})$
    Initialize parameters Λ
    repeat
      Solve parametrized submodular relaxation of the AFO problem using graph cuts
      Update Λ via proximal gradient descent
    until convergence of optimization over Λ
  Evaluate objective function $\mathcal{F}(\mathbf{x})$ at $\mathbf{x}_{t+1}$ to get y_{t+1}
  Aggregate the data: $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t \cup \{(x_{t+1}, y_{t+1})\}$ and update the model
  $t \leftarrow t + 1$
until convergence or maximum iterations
$\mathbf{x}_{best} \leftarrow \arg\min_{\mathbf{x}_t\in\mathcal{D}} \; y_t$
return the best uncovered input $\mathbf{x}_{best}$ and the corresponding function value $\mathcal{F}(\mathbf{x}_{best})$

term by bounding it below by a linear function $h(\mathbf{x})$ such that $h(\mathbf{x}) \leq \mathbf{x}^T A^+ \mathbf{x} \quad \forall \mathbf{x} \in \{0,1\}^n$. It can be easily seen that $h(\mathbf{x}) = \mathbf{x}^T(A^+ \circ \Lambda)\mathbf{1} + \mathbf{1}^T(A^+ \circ \Lambda)\mathbf{x} - \mathbf{1}^T(A^+ \circ \Lambda)\mathbf{1}$ is an affine lower bound to $\mathbf{x}^T A^+ \mathbf{x}$, where $\circ$ represents Hadamard product and Λ is a matrix defined as follows: $\Lambda = [\lambda_{ij}]_{n\times n}$, where $\lambda_{ij} \in [0,1]$ is a parameter satisfying the following inequality:

$$\lambda_{ij}(x_i + x_j - 1) \leq x_i x_j \tag{3.5}$$

Using $h(\mathbf{x})$ as the affine lower bound, our new optimization problem becomes:

$$\min_{\mathbf{x}\in\mathcal{X}} h(\mathbf{x}) + \mathbf{x}^T A^- \mathbf{x} + b^T \mathbf{x} \tag{3.6}$$

We use $h_\Lambda(\mathbf{x})$ for denoting the combination of two linear terms in (3.6) i.e. $h_\Lambda(\mathbf{x}) = h(\mathbf{x}) + b^T\mathbf{x}$, along with signifying the dependence on Λ parameter.

$$\min_{\mathbf{x}\in\mathcal{X}} h_\Lambda(\mathbf{x}) + \mathbf{x}^T A^- \mathbf{x} \tag{3.7}$$

It should be noted again that the objective in (3.7) is a lower bound of the original objective in (3.4). This relaxed submodular objective can be solved exactly by turning it into a minimum graph cut problem and utilizing an efficient minimum graph cut algorithm [5]. For a given Λ, we employ a standard graph construction strategy [23] (described in detail in the Appendix A.2.1) dependent on the α parameters sampled from the surrogate model f_α at each BO iteration. A graph G is constructed with $n+2$ vertices: $\mathcal{V} = \{s, t, v_1, \cdots, v_n\}$ where each non-terminal vertex v_i encode one discrete variable $x_i \in \{0,1\}$.

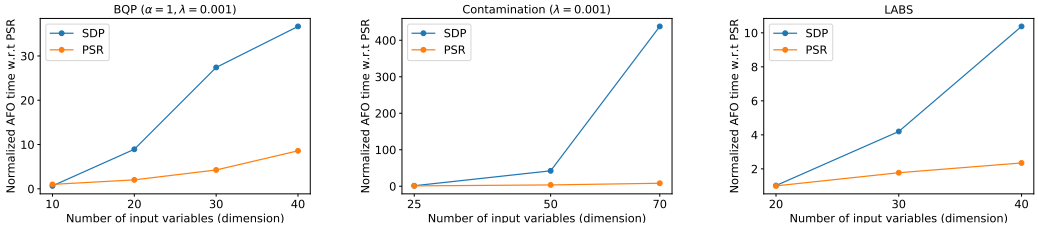
The quality of approximation of the above-mentioned submodular relaxation objective (3.7) critically depends on Λ parameters. [22] constructed an outer optimization problem to improve the accuracy of this relaxation and maximized the objective w.r.t Λ to achieve the best approximation possible. We use the same procedure which is described in the Appendix A.2.2.

4 Experiments and Results

We first describe our experimental setup, and then present results comparing state-of-the-art BOCS method with SDP relaxation and our proposed parameterized submodular relaxation (PSR) algorithm.

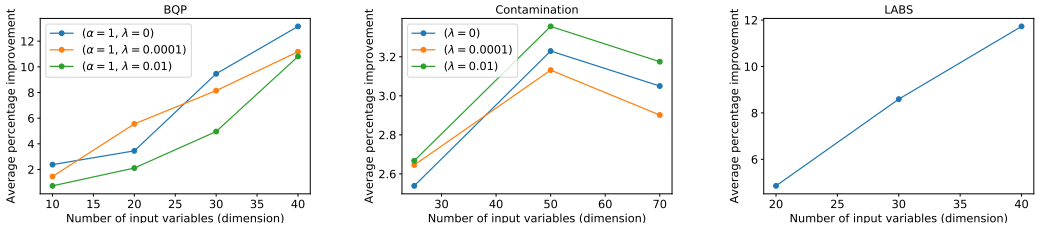
4.1 Experimental Setup

Benchmark domains. We employ five diverse synthetic and real-world benchmarks for our empirical evaluation. Additional experimental details and discussion on Ising benchmark are provided in the Appendix A.3



(a) BQP (b) Contamination (c) LABS

Figure 1: Results comparing PSR algorithm and SDP approach on *average AFO time* normalized w.r.t PSR. The title of each figure refers to the benchmark with corresponding parameter (if any).



(a) BQP (b) Contamination (c) LABS

Figure 2: Results comparing the accuracy of the AF minimizers obtained by PSR and SDP approach measured in terms of *Average percent improvement* in the AF objective over all iterations of the BO procedure.

1. Binary quadratic programming (BQP). The goal in binary quadratic programming (BQP) [2] is to maximize a binary quadratic function with l_1 regularization: $\max_{\mathbf{x}\in\{0,1\}^n}(\mathbf{x}^T Q\mathbf{x} - \lambda\|\mathbf{x}\|_1)$, where Q is a randomly generated matrix defined as Hadamard product of two matrices (M and K); $Q = M \circ K$, where $M \in \mathcal{R}^{n\times n}, M_{ij} = \mathcal{N}(0,1)$, $\mathcal{N}(0,1)$ stands for the standard Gaussian distribution and $K \in \mathcal{R}^{n\times n}, K_{ij} = \exp(-(i-j)^2/\alpha^2)$, α is the correlation length parameter.

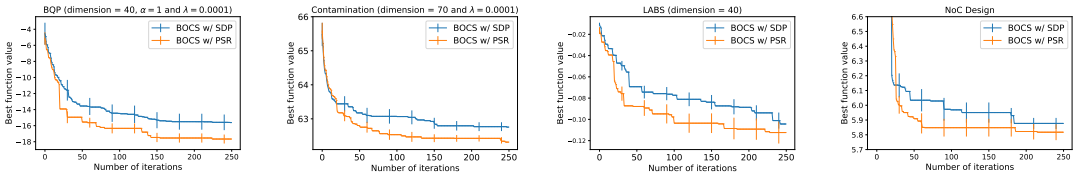
2. Contamination. This problem considers a food supply with n stages, where a binary {0,1} decision (x_i) must be made at each stage to prevent the food from being contaminated with pathogenic micro-organisms [19].

3. Low auto-correlation binary sequences (LABS). The problem is to find a binary {+1,-1} sequence $S = (s_1, s_2, \cdots, s_n)$ of given length n that maximizes *merit factor* defined over a binary sequence. This problem has multiple applications in diverse scientific disciplines including communications engineering where it is used in high-precision interplanetary radar measurements[26, 30].

4. Network optimization in multicore chips. The objective in this domain is to optimize the placement of 17 communication links between 12 cores of a multi-core architecture (*66 binary variables*) to facilitate efficient data transfer. This optimization is guided by expensive simulators that mimics the real hardware. The network optimization problem is part of the rodinia benchmark [9] and uses the gem5-GPU simulator [27].

Evaluation metrics. We demonstrate the advantages of our PSR algorithm for combinatorial BO by comparing it with the state-of-the-art BOCS approach along two fronts.

1) Scalability and accuracy of AF optimization. We compare PSR and SDP approaches for solving acquisition function optimization problems within BOCS method. To evaluate scalability for AFO, we report the average AFO time across all BO iterations normalized w.r.t PSR. Suppose T_{SDP} and T_{PSR} stand for average AFO time for some input dimensionality d. We normalize T_{SDP} and T_{PSR} using the AFO time of PSR for the smallest dimension. To evaluate the accuracy of solving AFO problems, we report the average percentage improvement in the AF objective achieved by PSR when compared to the corresponding AF objective from SDP.



(a) BQP (b) Contamination (c) LABS (d) Network optimization

Figure 3: Results comparing BOCS with PSR (orange line) algorithm and BOCS with SDP (blue line) approach on *best function value achieved* versus number of iterations. The horizontal axis (x-axis) depicts number of iterations while the vertical axis (y-axis) depicts the *best function value*.

2) Overall BO accuracy. We use the best function value achieved after a given number of BO iterations (function evaluations) as a metric to evaluate the two methods: `BOCS w/ SDP` and `BOCS w/ PSR`. Note that BOCS is already shown to significantly improve over SMAC [2]. The method that uncovers high-performing combinatorial structures with less number of function evaluations is considered better. We use the total number of BO iterations similar to BOCS [2].

4.2 Results for Acquisition Function Optimization

Average AFO time. Figure 1 shows the results of PSR and SDP approaches as a function of increasing dimension. Recall that we normalize the average AFO time w.r.t that of PSR for smallest dimension (base case). We can clearly see that the proposed PSR approach requires significantly low computation time when compared to the SDP approach and the gap increases with increasing input dimensions. This supports our claim that PSR algorithm improves the scalability of AFO problems in combinatorial BO setting. It should be noted that AFO problem is solved at each BO iteration. For example, if we run BO for 250 iterations, we need to solve 250 AFO problems. Therefore, the computational-efficiency of PSR is compounded across the entire BO procedure.

Average percentage improvement in AF objective. PSR algorithm also finds better optimized value for AFO problems on each benchmark domain as shown in Figure 2. The vertical axis of the plots in Figure 2 represent the average percentage improvement in AF objective obtained by PSR when compared to that obtained by SDP (higher the better). PSR algorithm always finds a minimizer with lower AF value when compared to SDP's minimizer on all benchmarks. Furthermore, this accuracy gap increases with increasing dimensions reinforcing the ability of PSR to scale to large dimensions while also improving the accuracy.

4.3 Results for Overall BO Accuracy

The main goal in BO is to find best accuracy on the true expensive black-box function $\mathcal{O}$. Ideally, the gains in accuracy for solving AFO problems as shown in previous section should reflect in the overall BO performance using the proposed PSR approach. Indeed, Figure 3 clearly shows that using the BOCS model with PSR algorithm improves the overall accuracy of the BO procedure on all benchmark domains. This is a direct consequence of the improved accuracy achieved by the PSR algorithm in solving AFO problems at each BO iteration. All the reported results are averaged over 10 random runs.

5 Conclusions

This paper studied a principled approach referred as parametrized submodular relaxation (PSR) to improve the scalability and accuracy of the state-of-the-art combinatorial Bayesian optimization algorithm with tractable statistical models called BOCS. The key idea is to reformulate the acquisition function optimization to select the next structure for evaluation as submodular relaxation with some parameters, and perform search over these parameters to improve the accuracy of this relaxed problem. Our experimental results on diverse benchmarks showed that PSR algorithm significantly improved the computational-efficiency and accuracy of BOCS.

References

[1] Ravindra K Ahuja, Thomas L Magnanti, and James B Orlin. Network flows. 1988.

[2] Ricardo Baptista and Matthias Poloczek. Bayesian optimization of combinatorial structures. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 462–471, 10–15 Jul 2018.

[3] Syrine Belakaria, Aryan Deshwal, and Janardhan Rao Doppa. Max-value entropy search for multi-objective Bayesian optimization. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[4] Syrine Belakaria, Aryan Deshwal, Nitthilan Kannappan Jayakodi, and Janardhan Rao Doppa. Uncertainty-aware search framework for multi-objective Bayesian optimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 10044–10052, 2020.

[5] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (9):1124–1137, 2004.

[6] Carlos M Carvalho, Nicholas G Polson, and James G Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

[7] Luis Ceze, Mark D. Hill, and Thomas F. Wenisch. Arch2030: A vision of computer architecture research over the next 15 years. *CoRR*, abs/1612.03182, 2016.

[8] Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending grothendieck's inequality. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 54–60. IEEE, 2004.

[9] Shuai Che, Michael Boyer, Jiayuan Meng, David Tarjan, Jeremy W. Sheaffer, Sang-Ha Lee, and Kevin Skadron. Rodinia: A benchmark suite for heterogeneous computing. In *Proceedings of the 2009 IEEE International Symposium on Workload Characterization (IISWC)*, pages 44–54, 2009.

[10] Aryan Deshwal, Syrine Belakaria, Janardhan Rao Doppa, and Alan Fern. Optimizing discrete spaces via expensive evaluations: A learning to search framework. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 3773–3780, 2020.

[11] Peter I Frazier. A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[12] Peter I Frazier and Jialei Wang. Bayesian optimization for materials design. In *Information Science for Materials Discovery and Design*, pages 45–75. Springer, 2016.

[13] Daniel Freedman and Petros Drineas. Energy minimization via graph cuts: Settling what is possible. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 939–946. IEEE, 2005.

[14] Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. freeman New York, 2002.

[15] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[16] Lena Gorelick, Yuri Boykov, Olga Veksler, Ismail Ben Ayed, and Andrew Delong. Submodularization for binary pairwise energies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1154–1161, 2014.

[17] Lena Gorelick, Yuri Boykov, Olga Veksler, Ismail Ben Ayed, and Andrew Delong. Submodularization for binary pairwise energies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1154–1161, 2014.

[18] Daniel Hernández-Lobato, Jose Hernandez-Lobato, Amar Shah, and Ryan Adams. Predictive entropy search for multi-objective Bayesian optimization. In *International Conference on Machine Learning*, pages 1492–1501, 2016.

[19] Yingjie Hu, JianQiang Hu, Yifan Xu, Fengchun Wang, and Rong Zeng Cao. Contamination control in food supply chain. In *Proceedings of the Winter Simulation Conference*, WSC '10, pages 2678–2681, 2010.

[20] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration (extended version). Technical Report TR-2010-10, University of British Columbia, Department of Computer Science, 2010. Available online: http://www.cs.ubc.ca/˜hutter/papers/10-TR-SMAC.pdf.

[21] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, pages 507–523, 2011.

[22] Shinji Ito and Ryohei Fujimaki. Large-scale price optimization via network flow. In *Advances in Neural Information Processing Systems*, pages 3855–3863, 2016.

[23] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (2):147–159, 2004.

[24] Huan Li and Zhouchen Lin. Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems*, pages 379–387, 2015.

[25] Changyong Oh, Jakub Tomczak, Efstratios Gavves, and Max Welling. Combinatorial bayesian optimization using the graph cartesian product. In *Advances in Neural Information Processing Systems*, pages 2914–2924, 2019.

[26] Tom Packebusch and Stephan Mertens. Low autocorrelation binary sequences. *Journal of Physics A: Mathematical and Theoretical, 49 (2016) 165001*, 2015.

[27] Jason Power, Joel Hestness, Marc Orr, Mark Hill, and David Wood. gem5-gpu: A heterogeneous cpu-gpu simulator. *Computer Architecture Letters*, 13(1), Jan 2014.

[28] Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

[29] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[30] Irwin I Shapiro, Gordon H Pettengill, Michael E Ash, Melvin L Stone, William B Smith, Richard P Ingalls, and Richard A Brockelman. Fourth test of general relativity: preliminary results. *Physical Review Letters*, 20(22):1265, 1968.

[31] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 26th Annual Conference on Advances in Neural Information Processing Systems.*, pages 2960–2968, 2012.

[32] Jialin Song, Yuxin Chen, and Yisong Yue. A general framework for multi-fidelity Bayesian optimization with Gaussian processes. In *AISTATS*, 2019.

[33] Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Pseudo-bound optimization for binary energies. In *European Conference on Computer Vision*, pages 691–707. Springer, 2014.

[34] Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM review*, 38(1):49–95, 1996.

[35] Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient Bayesian optimization. In *International Conference on Machine Learning (ICML)*, 2017.

A Appendix

A.1 Detailed related work

There is very limited work on BO over discrete spaces when compared to continuous space BO, which has seen huge growth over the last few years [11, 12, 31, 35, 32, 18, 3, 4]. SMAC [20, 21] is one canonical baseline which employs random forest as surrogate model and a *hand-designed* local search procedure for optimizing the acquisition function. BOCS [2] employs a parametric statistical models over binary variables, which allows pricipled acquisition function optimization based on semi-definite program solvers. COMBO [25] is a state-of-the-art non-parametric approach that employs Gaussian processes with diffusion kernels defined over discrete spaces as its surrogate model. However, COMBO employs local search with random restarts for acquisition function optimization.

COMBO was shown to achieve better performance than BOCS for complex domains that require modeling higher-order dependencies between discrete variables. However, BOCS achieves good performance whenever the modeling assumptions (e.g., lower-order interactions among variables) are met. Furthermore, the AFO problem in BOCS is a Binary Quadratic Programming (BQP) problem which is well-studied in many fields including computer vision [23] and prescriptive price optimization [22]. In comparison, the acquisition function optimization is much more challenging (results in general non-linear combinatorial optimization problem) for methods such as COMBO and SMAC that employs non-parametric statistical models. A learning to search framework referred as L2S-DISCO [10] was introduced recently to solve the challenges of AFO problems with complex statistical models (e.g., GPs with discrete kernels and random forest). The key insight behind L2S-DISCO is to directly tune the search via learning during the optimization process to select the next structure for evaluation by leveraging the close relationship between AFO problems across BO iterations (i.e., amortized AFO). Since the main focus of this paper is on improving the scalability and accuracy of AFO for the tractable statistical model introduced in BOCS, we describe the details of this approach below.

BOCS Approach. BOCS instantiates the three key elements of BO framework as follows. *1) Surrogate statistical model:* A linear Bayesian model defined over binary variables is employed as the surrogate model. The model is described as:

$$f_\alpha(\mathbf{x} \in \mathcal{X}) = \alpha_0 + \sum_j \alpha_j x_j + \sum_{i,j>i} \alpha_{ij} x_i x_j \tag{A.1}$$

where $\mathcal{X} = \{0,1\}^n$ and $\mathbf{x} \in \mathcal{X}$ is a binary vector and α variables are drawn from a sparsity-inducing horseshoe prior [6]. It was experimentally found that the above second-order model provides an excellent trade-off between expressiveness and accuracy. The α variables quantify the uncertainty of the model. *2) Acquisition function:* Thompson sampling [28] is employed as the acquisition function because of its proven theoretical and empirical properties in the context of BO. *3) Acquisition function optimization:* In each BO iteration, we select a candidate structure $\mathbf{x} \in \mathcal{X}$ for evaluation that minimizes the acquisition function. In the case of BOCS method, the acquisition function optimization (AFO) problem becomes:

$$\arg\min_{\mathbf{x} \in \mathcal{X}} f_\alpha(\mathbf{x}) + \lambda P(\mathbf{x}) \tag{A.2}$$

where $\lambda P(\mathbf{x})$ being a regularization term commonly seen in multiple applications. BOCS employs a semi-definite programming (SDP) based relaxation approach to solve the above AFO problem.

Scalability Challenges of BOCS. There are multiple challenges associated with SDP approach used for solving AFO problems in BOCS formulation. First, the time complexity of a standard SDP solver grows at the rate of $O(n^6)$ [34, 22], which is prohibitive for large dimensions. Second, the approximation error for SDP based solution is known to be at most $O(\log n)$ [2, 8], which clearly grows as the dimensions increase, resulting in the loss of accuracy as well. These scaling issues arise when the number of binary variables are large. Since BOCS represents categorical variables using one-hot encoding, even a small number of categorical variables can lead to a large number of binary variables (e.g., placement of processing elements in hardware design). Our goal in this paper is to provide an algorithmic approach to improve the computational-efficiency and accuracy of solving AFO problems for BOCS method.

A.2 Additional details of PSR algorithm

A.2.1 Graph construction strategy

For a given Λ, we employ a standard graph construction strategy [23] dependent on the α parameters sampled from the surrogate model f_α at each BO iteration to solve the objective described in (3.7). A graph G is constructed with $n+2$ vertices: $\mathcal{V} = \{s, t, v_1, \cdots, v_n\}$ where each non-terminal vertex v_i encode one discrete variable $x_i \in \{0, 1\}$. For each term depending on one variable x_i (each non-zero entry of $h_\Lambda(\mathbf{x})$ in (3.7)), an edge is added in the graph from s to v_i with capacity $h_\Lambda(x_i)$ if $h_\Lambda(x_i)$ is positive or from v_i to t with capacity $-1 \cdot h_\Lambda(x_i)$ if it is negative. Further, each term depending on pair of variables $x_i x_j$ (each non-zero entry of A^- in (3.7)) is represented by two edges i.e. edge v_i to v_j and edge v_j to t with capacity $-1 \cdot A^-_{ij}$.

A.2.2 Optimizing Λ Parameters to Improve Accuracy

The quality of approximation of the submodular relaxation objective (3.7) critically depends on Λ parameters. [22] constructed an outer optimization problem to improve the accuracy of this relaxation and maximized the objective w.r.t Λ to achieve the best approximation possible. We use the same procedure which is described as follows. By including the outer optimization over Λ, the overall problem becomes:

$$\max_{\Lambda \in [0,1]^{n \times n}} \quad (\min_{\mathbf{x} \in D} h_\Lambda(\mathbf{x}) + \mathbf{x}^T A^- \mathbf{x}) \tag{A.3}$$

Equivalently, the outer maximization can be turned into minimization by considering the negative of the outer objective.

$$\min_{\Lambda \in [0,1]^{n \times n}} \quad -1 \cdot (\min_{\mathbf{x} \in D} h_\Lambda(\mathbf{x}) + \mathbf{x}^T A^- \mathbf{x}) \tag{A.4}$$

This optimization problem can be solved efficiently using an iterative algorithm that alternates between solving the inner optimization over $\mathbf{x}$ via graph cut formulation and proximal gradient descent over Λ. If $\mathbf{x}_i$ is the solution of the submodularized inner objective in (A.4) at the i^{th} iteration for a fixed Λ_i, the update equation for Λ is given as follows:

$$\Lambda_{i+1} = (\Lambda_i - \eta_i G_i)^\perp \tag{A.5}$$

where η_i is the step size, G_i is the sub-gradient of the outer objective in (A.4) defined as $G_i = A^+ \circ (\mathbf{1}\mathbf{1}^T - \mathbf{x}_i \mathbf{1}^T - \mathbf{1}\mathbf{x}_i^T)$ and $\perp$ is the projection operator for any matrix P defined as $P^\perp_{ij} = \{0 \text{ if } P_{ij} < 0, 1 \text{ if } P_{ij} > 1, \text{ and } P_{ij} \text{ otherwise}\}$. We employ proximal gradient descent because it scales gracefully, is amenable to recent advances in auto-differentiation tools, and fast convergence [24]. We require few iterations (5-10) of proximal gradient descent and each inner optimization is very fast because of strongly polynomial graph cut algorithms. Indeed, our experiments validate this claim over multiple real-world benchmarks.

A.3 Additional experimental details and results

Algorithmic setup. For the sake of consistency, we convert all benchmark problems to minimization. Note that this can be achieved by minimizing the negative of the original objective if the true goal is to maximize the objective. We built our code on top of the open-source Python implementation of BOCS [1]. We employed the Boykov-Kolmogorov algorithm from graph-tool library [2] for solving minimum graph cut formulation of the relaxed submodular acquisition function objective noting that any minimum cut algorithm can be used to the same effect. We employed two initializations (random and $\mathbf{1}\mathbf{1}^T/2$) for optimizing Λ parameter within PSR noting that both gave similar results. We ran proximal gradient descent procedure for a maximum of 10 iterations on all benchmarks and achieved convergence. All the reported results are averaged over 10 random runs.

A.4 Benchmark description

1. Binary quadratic programming (BQP). The goal in binary quadratic programming (BQP) [2] is to maximize a binary quadratic function with l_1 regularization: $\max_{\mathbf{x} \in \{0,1\}^n} (\mathbf{x}^T Q \mathbf{x} - \lambda \|\mathbf{x}\|_1)$,

[1] https://github.com/baptistar/BOCS

[2] https://graph-tool.skewed.de/

where Q is a randomly generated matrix defined as Hadamard product of two matrices (M and K); $Q = M \circ K$, where $M \in \mathcal{R}^{n\times n}, M_{ij} = \mathcal{N}(0,1)$, $\mathcal{N}(0,1)$ stands for the standard Gaussian distribution and $K \in \mathcal{R}^{n\times n}, K_{ij} = \exp(-(i-j)^2/\alpha^2)$, α is the correlation length parameter.

2. Contamination. This problem considers a food supply with n stages, where a binary {0,1} decision (x_i) must be made at each stage to prevent the food from being contaminated with pathogenic micro-organisms [19]. Each prevention effort at stage i can be made to decrease the contamination by a given random rate Γ_i and incurring a cost c_i. The contamination spreads with a random rate Λ_i if no prevention effort is taken. The overall goal is to ensure that the fraction of contaminated food at each stage i does not exceed an upper limit U_i with probability at least $1-\epsilon$ while minimizing the total cost of all prevention efforts. Following [2], the lagrangian relaxation based problem formulation is given below:

$$\arg\min_{x} \sum_{i=1}^{n} \left[c_i x_i + \frac{\rho}{T} \sum_{k=1}^{T} 1_{\{Z_k > U_i\}} \right] + \lambda \|x\|_1$$

where λ is a regularization coefficient, Z_i is the fraction of contaminated food at stage i, violation penalty coefficient ρ=1, and T=100.

3. Low auto-correlation binary sequences (LABS). The problem is to find a binary {+1,-1} sequence $S = (s_1, s_2, \cdots, s_n)$ of given length n that maximizes *merit factor* defined over a binary sequence as given below:

$$\text{Merit Factor(S)} = \frac{n^2}{E(S)} \text{ where } E(S) = \sum_{k=1}^{n-1} \left(\sum_{i=1}^{n-k} s_i s_{i+k} \right)^2$$

The LABS problem has multiple applications in diverse scientific disciplines including communications engineering where it is used in high-precision interplanetary radar measurements[26, 30].

4. Network optimization in multicore chips. Multi-core architectures are considered very promising for parallel computing [7] in lieu of Moore's law aging quickly. Performance bottleneck due to data movement is a key challenge in multicore research. One promising solution is to optimize the placement of communication links between cores to facilitate efficient data transfer. The objective in this domain is to optimize the placement of 17 communication links between 12 cores of a multi-core architecture (*66 binary variables*) to facilitate efficient data transfer. This optimization is typically guided by expensive simulators that mimics the real hardware. The network optimization problem is part of the rodinia benchmark [9] and uses the gem5-GPU simulator [27]. There is one *constraint* to determine valid structures: existence of a viable path between any pair of cores. We use a large penalty on the objective value whenever this constraint is violated.

Other than the four benchmarks described in the main experimental section, we also evaluated the proposed approach on another important benchmark which is described below.

5. Sparsification of zero-field Ising models (Ising). The distribution of a zero field Ising model $p(z)$ for $z \in \{-1,1\}^n$ is characterized by a symmetric interaction matrix J^p whose support is represented by a graph $G^p = ([n], E^p)$ that satisfies $(i,j) \in E^p$ if and only if $J^p_{ij} \neq 0$ holds [2]. The overall goal in this problem is to find a close approximate distribution $q(z)$ while minimizing the number of edges in E^q. Therefore, the objective function in this case is a regularized KL-divergence between p and q as given below:

$$D_{KL}(p||q_{\mathbf{x}}) = \sum_{(i,j)\in E^p} (J^p_{ij} - J^q_{ij}) E_p[z_i z_j] + log(Z_q/Z_p)$$

where Z_q and Z_p are partition functions corresponding to p and q respectively, and $\mathbf{x} \in \{0,1\}^{E^q}$ is the decision variable representing whether each edge is present in E^q or not. As evident from Figure 4 and 5, our proposed PSR algorithm requires significantly less computation time while improving the quality of acquisition function optimization solution. Moreover, it also finds better overall BO solution on this domain as well (Figure 6).

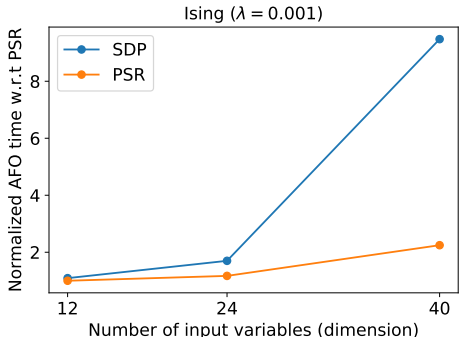


Figure 4: Results comparing PSR algorithm and SDP approach on *average AFO time* normalized w.r.t PSR for the Ising benchmark.

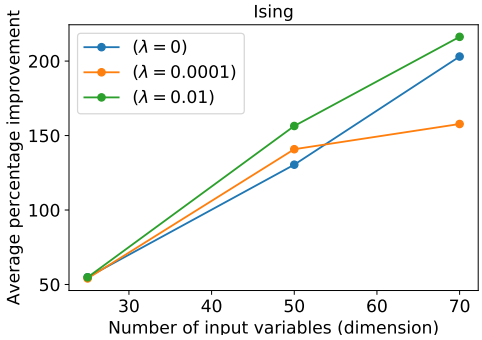


Figure 5: Results comparing the accuracy of the AF minimizers obtained by PSR and SDP approach measured in terms of *Average percent improvement* in the AF objective over all iterations of the BO procedure for the Ising benchmark.

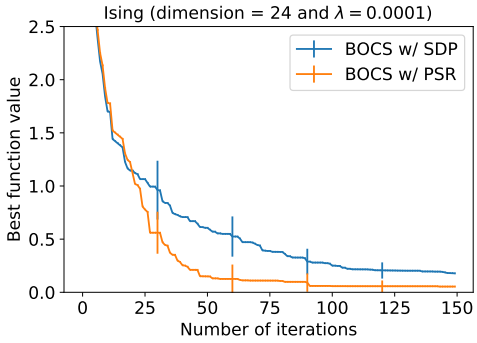


Figure 6: Results comparing BOCS with PSR algorithm and BOCS with SDP approach on *best function value achieved* versus number of iterations for the Ising benchmark. The horizontal axis (x-axis) depicts number of iterations while the vertical axis (y-axis) depicts the *best function value*.