24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Stochastic Local Search and Machine Learning: From Theory to Applications and Vice Versa

Ole Jakob Mengshoel[1,2], **Tong Yu**[2], **Ming Zeng**[2]

1 PRELUDE

In AI and ML, several methods rely on stochasticity or randomization: mutation and crossover in evolutionary algorithms; dropout and stochastic gradient descent in deep learning; stochasticity in stochastic local search (SLS) [5]; and randomization in systematic search [3]. SLS algorithms, which we study here, are competitive in solving computationally hard problems such as satisfiability (SAT), sparse signal recovery, scheduling, and most probable explanations in Bayesian networks (BNs) [16, 5, 6, 13, 15]. Essentially, SLS algorithms are greedy optimizers that also make random moves in order to avoid getting trapped in local but non-global optima.

Clearly, SLS is a well-established idea in AI. However, SLS is sometimes underutilized and there are certain potential difficulties when applying SLS. This highlight showcases recent developments that address some of these difficulties—related to ML, formalization with Markov chains, and setting hyperparameters—and briefly points to future research directions with potential.

2 PSEUDO-BOOLEAN FUNCTIONS

We consider the space of bit-strings $\mathbb{B} = \{0,1\}^m$, specifically functions f that map from $\mathbb{B}$ to the real numbers $\mathbb{R}$. Our focus is pseudo-Boolean function (PBF) optimization, where the goal is to optimize (without loss of generality, maximize) the fitness function f:

$$\boldsymbol{b}^* = \arg\max_{\boldsymbol{b}\in\mathbb{B}} f(\boldsymbol{b}) . \quad (1)$$

We keep (1) simple, in order to not complicate notation. However, one can generalize in order to handle multiple optima $\boldsymbol{b}_1^*, \boldsymbol{b}_2^*, \ldots$ and multiple fitness (or objective) functions $f_1, f_2, \ldots$.

While optimization of PBFs and closely related problems is typically computationally complex, there are also many and important applications including:

- Computing a BN's most probable explanation: Find an explanation with maximal posterior probability [10, 13, 12]
- Computing models in propositional logic: Find a satisfying assignment (model) [16, 5]
- Feature selection: Maximize accuracy by selecting the right features for machine learning [8, 4, 11, 20]
- Sparse signal recovery: Find, from a dense measurement (sound, image, video), the best sparse components [14, 15]
- Structure search for deep neural networks or Bayesian networks: Search for network structures that maximizes accuracy on test data

SLS may be used in all of these applications, as illustrated and discussed for some of them below.

3 STOCHASTIC LOCAL SEARCH CONCEPTS

What is SLS? Suppose that we start with the goal of PBF optimization via deterministic local search (or hill-climbing): start somewhere in f's search space $\boldsymbol{b} \in \mathbb{B}$, search for improvement in $\boldsymbol{b}$'s neighborhood, and iterate until a local optimum is found.

To reduce the problem of finding local but non-global optima, SLS adds pseudo-randomness or stochasticity. Specifically, SLS employs search operators that are based on the current state $\boldsymbol{b} \in \mathbb{B}$, such as:[3]

- The greedy operator O_G chooses, using f, a highest-fitness neighbor of $\boldsymbol{b}$: any bit-string $\boldsymbol{b}' \in \mathbb{B}$ that differs from $\boldsymbol{b}$ in one bit.
- The noise operator O_N flips a bit in $\boldsymbol{b}$ uniformly at random.
- The restart operator O_R randomly reinitializes to a new $\boldsymbol{b}' \in \mathbb{B}$.

The results after applying these operators vary [6, 10, 11, 20], but can be as follows. After O_N and O_R, SLS searches from $\boldsymbol{b}'$, while after O_G SLS searches from $\boldsymbol{b}'$ if $f(\boldsymbol{b}') \geq f(\boldsymbol{b})$.

As a concrete SLS case study, we now consider MarkovSLS [11]. In MarkovSLS, probabilistic hyperparameters p_G, p_N, and p_R control the application of their respective search operators O_G, O_N, and O_R. If $p_N = 1$, MarkovSLS performs a random walk using O_N only, and so forth. Since the total probability of p_G, p_N, and p_R adds to one, it is sufficient to specify two of the hyperparameters and let the third be implied. Thus, MarkovSLS is parameterized by the hyperparameters $A = (p_R, p_N)$, where p_R is the probability of applying O_R (restart) and p_N is the probability of applying O_N (noise).

The two MarkovSLS variants, SoftSLS and AdaptiveSLS [11], provide different ways of setting hyperparameters [9]. In SoftSLS, the hyperparameters (p_R, p_N) are tuned offline, enabling analysis using standard homogeneous Markov chains. Experimentally, we investigate the dependency of SoftSLS's performance on its hyperparameters. AdaptiveSLS, in contrast, performs online hyperparameter control [9]. Experimentally, on synthetic and feature selection problems, we compare AdaptiveSLS with other algorithms including SoftSLS analytically optimized via hyperparameter tuning [9]. We find that AdaptiveSLS performs very well while not requiring prior knowledge of the search space. This confirms and builds upon other results suggesting that SLS hyperparameter settings have great impact on search success and efficiency [5, 10, 2, 9, 20].

[1] Norwegian University of Science and Technology, ole.j.mengshoel-@ntnu.no
[2] Carnegie Mellon University, {ole.mengshoel, tong.yu, ming.zeng}-@sv.cmu.edu

[3] There are many other SLS methods, algorithms, and operators, as well as dramatically different approaches to stochastic optimization. Unfortunately, limited space does not allow them to be covered here.

4 STOCHASTIC LOCAL SEARCH TOPICS

Many innovations and results have moved SLS forward [6]. We now briefly discuss recent key topics at the interface of ML and SLS.

Stochastic Local Search and Machine Learning. Feature engineering, including feature selection, is essential in ML applications [4]. Assuming m features, feature selection easily maps into the PBF setting as follows: Each feature subset is represented by a bitstring $\mathbf{b} = (b_1, \ldots, b_m) \in \mathbb{B}$ and has a fitness $f(\mathbf{b})$. If a particular bit $b_i = 0$, we do not include the i-th feature when learning a model by means of wrapper feature selection [4]. In contrast, if $b_i = 1$, we do include the i-th feature. In ML, $f(\boldsymbol{b})$ is validation accuracy, using the feature set $\boldsymbol{b}$, for example for a classifier (kNN, decision tree, naïve Bayes, ...) and a particular dataset. SLS finds a feature set $\boldsymbol{b}'$, and outputs $f(\boldsymbol{b}')$ [11, 20]. The goal is: $\boldsymbol{b}' = \boldsymbol{b}^*$.

In the area of sparse signal recovery, which is closely related to ML, our recent stochastic CoSaMP (StoCoSaMP) algorithm [15] substantially improves the performance of a seminal greedy pursuit algorithm, CoSaMP [14]. This was done by integrating SLS methods and CoSaMP. After tuning [9] of p_N, using for example $p_N = 0.5$, StoCoSaMP increases accuracy while reducing computational cost relative to CoSaMP across a broad range of problem instances [15].

Stochastic Local Search Formalization. SLS algorithms typically have multiple hyperparameters that determine their performance. Unfortunately, the understanding of how these hyperparameters interact in different contexts, such as when problem difficulty, computing platform, or algorithm use case vary, has in our opinion been lagging compared to the strong empirical results often obtained with SLS. Current research seeks to improve the theoretical foundation of SLS [10, 2, 11] while also developing SLS algorithms with competitive performance on problems of practical interest.

We have therefore formulated and analyzed several parametric Markov chain models of SLS [10, 11]. In such Markov chains, SLS hyperparameters show up as parameters that can be varied and even optimized [10, 2, 11]. Using these parametric Markov chains, we compute expected hitting times, show that they are rational functions (ratios of polynomials) for individual problem instances as well as for their mixtures, and use them to aid in SLS performance optimization. Interestingly, such expected hitting time curves are analytical counterparts to noise response curves reported in the experimental literature [10]. Hand in hand with this improved theoretical understanding of SLS, we have studied the computational problems of MPE in BNs [10, 12, 13] and feature selection in ML [11, 20].

Stochastic Local Search Optimization. A straightforward way to optimize SLS is to adopt recent hyperparameter tuning methods [7, 17, 18]. However, we experienced that while very general and mathematically elegant, such methods are not always scalable and can be slow [20]. But, as discussed above, the SLS optimization problem can be reduced to minimizing a rational function. While the use of exact rational functions in practical PBF optimization is unrealistic, we benefit from insights obtained via these results.

Specifically, low-order polynomial approximations can be surprisingly accurate [10]. Thus, we set out to perform polynomial approximations in a computationally efficient way [20], using multi-armed bandits (MABs) integrated with our SLS algorithm. In particular, we developed an SLS MAB, in which each MAB arm A corresponds to a hyperparameter pair $A = (p_R, p_N)$ of SLS and SLS provides rewards to the MAB. To efficiently leverage the polynomial approximation, we propose a parametric setting and update the parametric model using Thompson sampling [19, 1]. We use a polynomial function to approximate the reward given $A = (p_R, p_N)$, and find that degree-4 polynomials perform very well while being extremely computationally efficient [20].

5 POSTLUDE

Reflecting on our experience with CoSaMP [14], we hypothesize that there are other areas of computing in which greedy algorithms are very popular, but by introducing SLS methods (ala StoCoSaMP [15]) results can be substantially improved. Is that true? Second, the computational cost of ML can currently be high, and the use of SLS for ML, for example for feature selection by means of wrapper methods [4, 11, 20], is no exception. Maybe the Markov chain-based analysis discussed here can be the basis for SLS methods that better handle massive and complex datasets while reducing computational cost dramatically? Third, the theory and formal methods communities have also studied parametric Markov chains, but interactions with the AI and ML communities have been limited. Perhaps this will change, now that trustworthy AI is coming to the forefront?

REFERENCES

[1] O. Chapelle and L. Li, 'An empirical evaluation of Thompson sampling', in *NIPS*, pp. 2249–2257, (2011).
[2] S. Ermon, C. Gomes, A. Sabharwal, and B. Selman, 'Designing fast absorbing Markov chains', in *AAAI*, pp. 849–855, (2014).
[3] C. P. Gomes, B. Selman, and H. Kautz, 'Boosting combinatorial search through randomization', in *AAAI*, pp. 431–437, (1998).
[4] I. Guyon and A. Elisseeff, 'An introduction to variable and feature selection', *JMLR*, **3**, 1157–1182, (2003).
[5] H. H. Hoos, 'An adaptive noise mechanism for WalkSAT', in *AAAI-2002*, pp. 655–660, (2002).
[6] H. H. Hoos and T. Stützle, *Stochastic Local Search: Foundations and Applications*, Morgan Kaufmann, San Francisco, 2005.
[7] F. Hutter, H. H. Hoos, and K. Leyton-Brown, 'Sequential model-based optimization for general algorithm configuration', in *LION*, (2011).
[8] R. Kohavi and G. H. John, 'Wrappers for feature subset selection', *AIJ*, **97**(1), 273–324, (1997).
[9] G. Krafotias, M. Hoogendoorn, and A. E. Eiben, 'Parameter control in evolutionary algorithms: Trends and challenges', *IEEE Transactions on Evolutionary Computation*, **19**(2), 167–187, (2015).
[10] O. J. Mengshoel, 'Understanding the role of noise in stochastic local search: Analysis and experiments', *AIJ*, **172**(8), 955–990, (2008).
[11] O. J. Mengshoel, Y. Ahres, and T. Yu, 'Markov chain analysis of noise and restart in stochastic local search', in *IJCAI*, pp. 639–646, (2016).
[12] O. J. Mengshoel, D. Roth, and D. C. Wilkins, 'Portfolios in stochastic local search: Efficiently computing most probable explanations in Bayesian networks', *JAR*, **46**(2), 103–160, (2011).
[13] O. J. Mengshoel, D. C. Wilkins, and D. Roth, 'Initialization and restart in stochastic local search: Computing a most probable explanation in Bayesian networks', *IEEE TKDE*, **23**(2), 235–247, (2011).
[14] D. Needell and J. A. Tropp, 'CoSaMP: Iterative signal recovery from incomplete and inaccurate samples', *Applied and Computational Harmonic Analysis*, **26**(3), 301–321, (2009).
[15] D. K. Pal and O. J. Mengshoel, 'Stochastic CoSaMP: Randomizing greedy pursuit for sparse signal recovery.', in *ECML-PKDD*, pp. 761–776, (2016).
[16] B. Selman, H. Levesque, and D. Mitchell, 'A new method for solving hard satisfiability problems', in *AAAI*, pp. 440–446, (1992).
[17] J. Snoek, H. Larochelle, and R. P. Adams, 'Practical Bayesian optimization of machine learning algorithms', in *NIPS*, pp. 2951–2959, (2012).
[18] N. Srinivas, A. Krause, M. Seeger, and S. M. Kakade, 'Gaussian process optimization in the bandit setting: No regret and experimental design', in *ICML*, pp. 1015–1022, (2010).
[19] W. R. Thompson, 'On the likelihood that one unknown probability exceeds another in view of the evidence of two samples', *Biometrika*, **25**(3-4), 285–294, (1933).
[20] T. Yu, B. Kveton, and O. J. Mengshoel, 'Thompson sampling for optimizing stochastic local search', in *ECML-PKDD*, pp. 493–510, (2017).