EPiC Series in Computing

Volume 71, 2020, Pages 45–51

Vampire 2018 and Vampire 2019.
The 5th and 6th Vampire Workshops

Bayesian Optimisation for Heuristic Configuration in Automated Theorem Proving

Agnieszka Słowik, Chaitanya Mangla, Mateja Jamnik, Sean B. Holden, and Lawrence C. Paulson

University of Cambridge, Department of Computer Science and Technology
Cambridge, UK
agnieszka.slowik@cl.cam.ac.uk

Abstract

Modern theorem provers such as Vampire utilise premise selection algorithms to control the proof search explosion. Premise selection heuristics often employ an array of continuous and discrete parameters. The quality of recommended premises varies depending on the parameter assignment. In this work, we introduce a principled probabilistic framework for optimisation of a premise selection algorithm. We present results using Sumo Inference Engine (SInE) and the Archive of Formal Proofs (AFP) as a case study. Our approach can be used to optimise heuristics on large theories in minimum number of steps.

1 Introduction

Theorem provers use heuristics at various points in their operation, such as in search control and premise selection. These heuristics often have parameters that greatly influence the practical performance of a prover. Existing approaches to selecting such parameters require human supervision, rules of thumb or extensive testing [4]. Such testing is often conducted on large theory sets, and is thus computationally expensive. Every assignment to the collection of parameters forms a point in parameter space, and as the parameters grow in number and range, an exhaustive search for optimal parameters becomes infeasible. An alternative is to sparsely navigate the space of parameters in search of the optimal point using probabilistic search.

A simple probabilistic approach to parameter selection is to use a variation of *ϵ-greedy search* [13]. Given a metric that determines the value of each parameter combination within the parameter range, the algorithm starts at a random point and proceeds with exploring the local neighbourhood. The best point found becomes the starting point for the next iteration. As a result, the agent is approaching a local optimum. With probability equal to ϵ, the agent randomly draws a new point for exploration. In theory, the ϵ-greedy search is an exhaustive search given unlimited time, and therefore in the limit it should give us the global optimum. In practice, it makes inefficient use of resources since the points it tests tend to be densely clustered. The key issue here is that the knowledge gained by testing any point is discarded.

The knowledge gained from every tested point in the parameter space can be used to reduce the uncertainty about the shape of the objective function. We can use our understanding of the objective function within this uncertainty to make an informed decision about the next point to test. Every test further reduces our uncertainty. Furthermore, we often have a priori knowledge of certain aspects of the objective function. For example, we may have an idea of how close we expect neighbouring points to be. If we construct a probabilistic model of the objective function with this a priori knowledge, we can then use the knowledge gained from testing arbitrary points to increase our certainty, thereby improving our prediction of the next best point to test.

Bayesian Optimisation [9] is a principled method for this purpose and *Gaussian Processes* (GPs) [11] provide a means for probabilistic modelling of functions using prior knowledge and machine learning. We can thus search for the parameter assignment that optimises the heuristic (such as an established premise selection algorithm) without the cost of extensive testing.

We conduct a case study using a state-of-the-art heuristic SInE [4], a premise selection algorithm used in Vampire. Our framework based on GPs takes 4.3 minutes under expectation to find the optimal set of parameters in an AFP article. The premises recommended by the optimised SInE are sufficient to prove 85.3% of the conjectures using Sledgehammer [1].

2 Background

2.1 Related work

The Sequential Model-based Algorithm Configuration framework (SMAC) [5] is a similar approach to efficient optimisation of algorithm parameters, in particular SAT solvers [7]. SMAC employs *Expected Improvement* criterion while our framework uses *Upper Confidence Bound* (UCB), as explained in Section 2.3. The parameter κ available in UCB allows a fine tuning of the exploration-exploitation trade-off in parameter search.

2.2 Premise selection and SInE

Here, we focused on the task of *premise selection* which can be defined as follows:

Definition 1. *Given a set of premises $\mathbb{P}$, an ATP system $\mathcal{A}$ and a new conjecture C, select the premises from $\mathbb{P}$ that will most likely lead to a proof of C by $\mathcal{A}$.*

The size of the modern mathematical corpora creates a "needle in a haystack" problem for automated theorem provers. Only a small subset of available premises is relevant to any given conjecture. For instance, Open CYC [8] contains over 3 million axioms while each of the problems has a proof involving up to 20 premises.

SInE is a simple heuristic-based premise selection algorithm introduced to optimise reasoning in large theories. The algorithm aims to estimate the importance of function and predicate symbols based on their frequencies in the conjecture and in the premises at hand. The least frequent symbols indicate a *trigger relation* between the goal statement and a premise. This basic variant of the trigger relation is defined as follows:

Let $occ(s)$ be the number of premises in which the symbol s occurs and S the set of all symbols in a premise p. We define the least general symbol s' as the symbol for which:

$$\forall s \in S : occ(s') \leq occ(s)$$

and use it as a trigger for p. If the symbol s' appears in the conjecture, the algorithm selects the premise p. This basic heuristic suffers from low robustness since small changes in the number of frequencies can lead to a loss of important premises. The heuristic was thus extended in the following way:

$$trigger(s') \iff \{occ(s') \leq g\} \vee \{\forall s \in S : occ(s') \leq t \cdot occ(s)\}$$

where the parameters $t \geq 1$, $g \geq 1$ are referred to as the *tolerance threshold* and *generality threshold*, respectively.

Finally, premises triggered by the goal may contain symbols that lead to other relevant premises. This introduces another parameter $k \geq 0$ referred to as *depth*. It leads to the inductive construction of triggering symbols and premises:

1. All symbols s of the goal are 0-step triggered.
2. If s is k-step triggered and it triggers a premise p, then p is $k+1$-step triggered.
3. If p is k-step triggered and s occurs in p, then s is k-step triggered.

The algorithm is therefore parameterised by one continuous parameter t and two discrete parameters g and k. These parameters were shown to greatly influence the performance of the algorithm. The disparities between the optimal parameter assignment depending on the problem set can be significant. For instance, premise selection on SUMO and CYC problems from the TPTP library benefits from setting the depth parameter k to infinity, while using a problem set from Mizar results in selection of the sufficient number of premises with the depth limit set to one [4]. Bayesian optimisation framework provides an automatic suggestion of the optimal parameter assignment based on the problem set.

2.3 Bayesian optimisation

Bayesian optimisation methods construct a probabilistic model of the objective function f and use it to determine informative sample locations. Under some prior on f, the points in the parameters space are repeatedly evaluated based on the posterior mean and variance predictions.

In our setting, the unknown objective function represents the usefulness of SInE given a parameter assignment. We assume this function was sampled from a Gaussian process. As we evaluate the performance of SInE given the point in the parameter space, the Bayesian optimisation framework improves the posterior distribution for the objective function as the agent becomes more certain of the regions worth exploring. In our implementation we choose the point in the parameter space to be evaluated in the next iteration based on the posterior distribution and upper confidence bound of a Gaussian process which is one of the standard methods referred to as the *Gaussian Process-Upper Confidence Bound (GP-UCB)* algorithm [12].

A Gaussian Process is a distribution over functions specified by a mean μ and a covariance function κ:

$$f(x) \sim GP(\mu(x), \kappa(x, x')).$$

Common choices of covariance function include the finite dimensional linear, squared exponential and Matérn kernels [2]. We used the Matérn kernel which can be seen as a generalisation of the Gaussian radial basis function.

We define the *upper confidence bound* in the maximisation problem as:

$$UCB(x) = \mu(x) + \kappa\sigma(x)$$

where $\kappa \geq 0$. The first part of the update rule favours the points x which are likely to give a high reward in terms of the objective function. The second part prefers the points where the function f is uncertain, thereby negotiating the trade-off between exploitation and exploration. The amount of exploration is controlled by the constant κ.

3 Case study

3.1 Dataset

The Archive of Formal Proofs (AFP) is a collection of proofs formalised in Isabelle [10]. We used a parsed version of the dataset that meets the input requirements of MaSh [6], the machine learning premise selector currently implemented in Isabelle. Here, we report the results on 10 articles containing various theories of sizes ranging from around 100 to around 1500 conjectures. Each conjecture was paired with a history of key premises necessary to prove the conjecture with Sledgehammer [1]. These logs provide the ground truth for our experiments – we assume that a successful premise selection includes all of the premises required by Sledgehammer.

3.2 Evaluation metrics

In premise selection, it is acceptable to provide more premises than necessary to prove a conjecture in order to minimise the risk of missing a key lemma. However, the main purpose of filtering is to lower the cost of considering irrelevant lemmas, and so an efficient algorithm should minimise the number of unnecessary recommendations.

To let this trade-off guide the optimisation process, we propose a metric based on *precision* and *recall*. In our setting, recall is represented by the ratio of the relevant premises recommended by the selection algorithm to the total number of premises needed for the proof. We experimented with several approaches to expressing precision in our setting and found that using a ratio of relevant premises recommended by SInE to an expression that increases exponentially with the size of the recommended set:

1. leads to an objective function that is strongly correlated with the main goal of recommending all of the premises needed to prove a conjecture;

2. favours a small number of redundant recommendations over the risk of missing one of the key premises, but penalises large sets of recommendations;

3. reduces sparsity that arises from directly optimising the number of theorems proved, which improves the optimisation process.

This metric is computed individually for each conjecture as follows:

Assuming we have a set of n conjectures, let $i = 1, 2, \ldots, n$ be the index of a conjecture to be proved. Let P_i be a non-empty set of lemmas required to prove the conjecture i and $\tilde{P}_i$ a set of lemmas recommended by a premise selection algorithm. Here, P_i is the set of premises used by Sledgehammer to prove the conjecture i, and $\tilde{P}_i$ refers to the set of premises recommended by SInE. If $|\tilde{P}_i| \neq 0$:

AFP article	Nr of goals	Proofs found [%]	Time [s]	Optimal parameters
Polynomials	135	87%	57s	t: 16.3, g: 58, k: 131
AbstractHoareLogics	793	63%	249s	t: 17.6, g: 57, k: 130
Completeness	475	89%	151s	t: 18.9, g: 63, k: 134
FinFun	263	95%	73s	t: 19.6, g: 57, k: 132
HeardOf	716	93%	331s	t: 19.5, g: 57, k: 131
InductiveConfidentiality	1425	82%	451s	t: 19.6, g: 58, k: 130
RefineMonadic	1509	95%	522s	t: 14.7, g: 64, k: 123
MiniML	345	84%	104s	t: 19.1, g: 58, k: 131
RecursionTheory	656	85%	205s	t: 19, g: 57, k: 130
SortEncodings	776	80%	437s	t: 14, g: 64, k: 123

Table 1: Premise selection for AFP. We report the time required to find the global optimum for each of the articles. Next, we compare the premises recommended by SInE controlled by the optimal set of parameters to the set of premises used by Sledgehammer in the actual proof search. We assume that all of the premises used in the proof search are crucial. Consequently, the values in the "Proofs found" column can be considered as the lower bound.

$$S_i = \frac{|\tilde{P}_i \cap P_i|}{|P_i|} + \frac{|\tilde{P}_i \cap P_i|}{2^{|\tilde{P}_i|}}.$$

For $|\tilde{P}_i| = 0$ we set the score S_i to zero since the conjecture could not be proved. After computing the value of S_i for each conjecture, we define the objective function across the whole dataset as follows:

$$S = \sum_{i=1}^{n} S_i.$$

At the testing stage we evaluate the algorithm based on the number of conjectures that would be proved in practice by Sledgehammer using the premises recommended by SInE. We assume that all of the premises used by Sledgehammer are necessary to prove the conjecture whereas in practice the prover might be able to find an alternative solution that requires a different set of premises. Consequently, this testing metric will tend to underestimate the number of conjectures proved using the SInE recommendations.

3.3 Results and analysis

GPs assume continuous input variables. Discrete-valued parameters require additional approximations. We followed a common approach which is to use a surrogate continuous variable for each discrete parameter, and round its value to the closest integer before evaluating the objective.

The results of our case study using ten AFP articles (see Table 1) suggest that the framework is efficient in finding the optimal parameter combination across different theories. This allows us to explore a wider range of parameters and produce an offline heuristic recommendation to a theorem prover.

4 Future Work

An interesting consecutive study involves comparing SMAC to Bayesian optimisation in terms of time and accuracy, in the context of optimising the heuristics used in theorem proving. Another possible direction would be to experiment with recent approaches to handling discrete-valued parameters in Bayesian optimisation with GPs [3].

5 Conclusion

Motivated by the challenges in algorithm configuration in theorem proving, we proposed a principled approach to finding a global optimum in a minimum number of steps. The framework can be used to improve the existing heuristics, as well as to draw an accurate comparison between the rule-based heuristics and new methods for premise selection.

References

[1] Sascha Böhme and Tobias Nipkow. Sledgehammer: Judgement day. In *Proceedings of the 5th International Conference on Automated Reasoning*, pages 107–121, Berlin, Heidelberg, 2010. Springer-Verlag.

[2] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, 11 2014.

[3] Eduardo C. Garrido-Merchán and Daniel Hernández-Lobato. Dealing with integer-valued variables in bayesian optimization with gaussian processes. 06 2017.

[4] Kryštof Hoder and Andrei Voronkov. Sine qua non for large theory reasoning. In Nikolaj Bjørner and Viorica Sofronie-Stokkermans, editors, *Automated Deduction – CADE-23*, pages 299–314, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[5] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the 5th International Conference on Learning and Intelligent Optimization*, LION'05, pages 507–523, Berlin, Heidelberg, 2011. Springer-Verlag.

[6] Daniel Kühlwein, Jasmin Christian Blanchette, Cezary Kaliszyk, and Josef Urban. Mash: Machine learning for sledgehammer. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving*, pages 35–50, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[7] Marius Lindauer and Frank Hutter. Warmstarting of model-based algorithm configuration. *CoRR*, abs/1709.04636, 2017.

[8] C. Matuszek, J. Cabral, and M. Wirbrock. An introduction to the syntax and content of Cyc. In B. Chitta, editor, *Formalizing and compiling background knowledge and its applications to knowledge representation and question answering. Papers from AAAI spring symposium*, page 44. AAAI Press, Menlo Park, CA, 2006.

[9] Jonas Močkus. On bayesian methods for seeking the extremum. In G. I. Marchuk, editor, *Optimization Techniques IFIP Technical Conference Novosibirsk, July 1–7, 1974*, pages 400–404, Berlin, Heidelberg, 1975. Springer Berlin Heidelberg.

[10] Tobias Nipkow, Markus Wenzel, and Lawrence C. Paulson. *Isabelle/HOL: A Proof Assistant for Higher-order Logic*. Springer-Verlag, Berlin, Heidelberg, 2002.

[11] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[12] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 1015–1022, USA, 2010. Omnipress.

[13] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.