Project Proposal: Model-Based Optimization of Strategy Schedules

Filip Bártek Martin Suda

Czech Technical University in Prague

September 7, 2021

Task definition

Input		Example
Parameterized target algorithm	A	Vampire (an ATP)
Parameter configuration space	$\boldsymbol{\Theta}$	Vampire strategy space
Instance set	Π	FOL problems from TPTP
Cost metric	c	PAR10[1]

Output	Task	Tool
Configuration	Algorithm configuration	SMAC
Portfolio	Portfolio optimization	Hydra
Schedule	Schedule optimization	HOS-ML

[1] Penalized average runtime with a penalty factor of 10

Sequential Model-based Algorithm Configuration (SMAC)[2]

Input: Target algorithm A; parameter configuration space $\boldsymbol{\Theta}$; instance set Π; cost metric c

Output: Configuration $\boldsymbol{\theta}_{inc} \in \boldsymbol{\Theta}$

$[\boldsymbol{R}, \boldsymbol{\theta}_{inc}] \leftarrow \mathit{Initialize}(\boldsymbol{\Theta}, \Pi)$

$\triangleright\ R = \{([\boldsymbol{\theta}_1, \pi_1], o_1), \ldots, ([\boldsymbol{\theta}_n, \pi_n], o_n)\} \subseteq [\boldsymbol{\Theta} \times \Pi] \times \mathbb{R}$

repeat

$\quad \mathcal{M} \leftarrow \mathit{FitModel}(\boldsymbol{R})$

$\quad \boldsymbol{\Theta}_{new} \leftarrow \mathit{SelectConfigurations}(\mathcal{M}, \boldsymbol{\theta}_{inc}, \boldsymbol{\Theta}, \boldsymbol{R})$

$\quad [\boldsymbol{R}, \boldsymbol{\theta}_{inc}] \leftarrow \mathit{Intensify}(A, \boldsymbol{\Theta}_{new}, \boldsymbol{\theta}_{inc}, \boldsymbol{R}, \Pi, c)$

until total time budget for configuration exhausted

return $\boldsymbol{\theta}_{inc}$

[2] Hutter et al. [2011]

Empirical performance model (EPM) $\mathcal{M}$

Input: Configuration $\boldsymbol{\theta}$

Output: Predictive statistics of $c(\boldsymbol{\theta}, \Pi)$

- Mean $\mu_{\boldsymbol{\theta}}$
- Variance $\sigma^2_{\boldsymbol{\theta}}$

Architecture: Random forest (10 regression trees)

Prediction:

1. Each tree predicts $c(\boldsymbol{\theta}, \pi)$ for configuration $\boldsymbol{\theta}$ and instance π (represented by a feature vector)
2. Aggregation across instances: mean
3. Aggregation across trees: mean and variance

Batch evaluation for multiple configurations and instances is cheap.

Candidate configuration selection ($SelectConfigurations$)

Positive improvement: $I(\boldsymbol{\theta}) = \max\{0, c(\boldsymbol{\theta}_{inc}, \Pi) - c(\boldsymbol{\theta}, \Pi)\}$
Maximize the *expected* positive *improvement* (EI) over $\boldsymbol{\theta}_{inc}$:

- 10 000 random configurations
- Multi-start local search
  - Initial population: 10 configurations with the highest EI in $\boldsymbol{R}$
  - Randomized one-exchange neighborhood

Interleave:

- Configurations with high EI
- Random configurations

Intensification

Input: Target algorithm A; sequence of candidate configurations $\boldsymbol{\Theta}_{new}$; incumbent configuration $\boldsymbol{\theta}_{inc}$; set of target algorithm runs $\boldsymbol{R}$; instance set Π; cost metric c

Output: Set of runs $\boldsymbol{R}$; new incumbent configuration $\boldsymbol{\theta}_{inc}$

for $\boldsymbol{\theta}_{new} \in \boldsymbol{\Theta}_{new}$ **do**
    Evaluate $\boldsymbol{\theta}_{inc}$ on a random instance from $\Pi \setminus \Pi[\boldsymbol{\theta}_{inc}]$
    loop
        Evaluate $\boldsymbol{\theta}_{new}$ on some subset of $\Pi[\boldsymbol{\theta}_{inc}] \setminus \Pi[\boldsymbol{\theta}_{new}]$
        if $c(\boldsymbol{\theta}_{new}, \Pi[\boldsymbol{\theta}_{new}]) > c(\boldsymbol{\theta}_{inc}, \Pi[\boldsymbol{\theta}_{new}])$ **then break**
        end if
        if $\Pi[\boldsymbol{\theta}_{new}] = \Pi[\boldsymbol{\theta}_{inc}]$ **then** $\boldsymbol{\theta}_{inc} \leftarrow \boldsymbol{\theta}_{new}$; **break**
        end if
    end loop
end for
return $[\boldsymbol{R}, \boldsymbol{\theta}_{inc}]$

Hydra[3] with SMAC

Input: Target algorithm A; parameter configuration space $\boldsymbol{\Theta}$; instance set Π; cost metric c; number of iterations K

Output: Portfolio P

$c_1 \leftarrow c$

$\boldsymbol{\theta}_1 \leftarrow SMAC(A, \boldsymbol{\Theta}, \Pi, c_1)$

$P_1 \leftarrow \{\boldsymbol{\theta}_1\}$

for $k \leftarrow 2 \ldots K$ **do**

  Define c_k: $c_k(\boldsymbol{\theta}, \pi) = \min(c(\boldsymbol{\theta}, \pi), \min_{\hat{\boldsymbol{\theta}} \in P_{k-1}} c(\hat{\boldsymbol{\theta}}, \pi))$

  $\boldsymbol{\theta}_k \leftarrow SMAC(A, \boldsymbol{\Theta}, \Pi, c_k)$

  $P_k \leftarrow P_{k-1} \cup \{\boldsymbol{\theta}_k\}$

end for

return P_K

[3] Xu et al. [2010]

Issues with SMAC3

- Bugs
- If the instance set is large, the iterated local search for configuration takes a lot of time

Initial experiment

- Instance set: 1000 FOL problems from TPTP 7.5.0
- Instance features: 32 syntactic features
- Target algorithm: Vampire
    - Parameter configuration space: 113 parameters
- Training computation budget: 4 CPUs, 8 hours

Results

Solver	Configurations	Time limit [s]	Timeouts[4]
Hydra	1	8	518
Hydra	2	4	503
Hydra	4	2	479
Hydra	8	1	465
Vampire[5]	1	8	422

[4] Number of timeouts on 1000 training problems

[5] `vampire --mode casc`

Future

- Optimize a configuration schedule
  - Interleave optimization of configurations and schedule
  - Modify Hydra: Construct schedule iteratively
  - Modify SMAC: Replace the incumbent configuration with an incumbent schedule
- Analyze the EPM, namely the parameter importance
- Better instance features
  - Runtime statistics of probing runs
- More problem domains (SMT, HOL)

Thank you for your attention!

Filip Bártek
filip.bartek@cvut.cz

Acknowledgement: This work is supported by the Czech Science Foundation project no. 20-06390Y (JUNIOR grant), and the Grant Agency of the Czech Technical University in Prague, grant no. SGS20/215/OHK3/3T/37.

Terminology

ATP	SMAC	Symbol
prover	target algorithm	A
strategy	parameter configuration	$\boldsymbol{\theta}$
strategy space	parameter configuration space (PCS)	$\boldsymbol{\Theta}$
problem	instance	π
problem set	instance set	Π
	runtime budget	B
	portfolio	P
schedule		f
	EPM	$\mathcal{M}$

Tasks

- Per-instance algorithm selection (AS)
- Algorithm configuration (AC)
  - One configuration for all instances
  - Per-instance
- Portfolio
- Scheduling

Parameter configuration space example

Vampire call example

vampire −−age_weight_ratio 1:1 −−naming 8

PCS

```
age_weight_ratio:log_ratio real [-10.0, 3.0] [0.0]

naming:special categorical {regular, 0} [regular]
naming integer [2, 64] [8] log
naming | naming:special == regular
```

Experiment setup

Sequential Model-based Algorithm Configuration (SMAC) command line options

```
smac
--acq_opt_challengers 1000
--sls_n_steps_plateau_walk 2
--mode Hydra
--n_optimizers 4
```

Scenario

```
run_obj = runtime
overall_obj = par10
deterministic = true
initial_incumbent = DEFAULT
```

SMAC summary

- Empirical performance model (EPM): Random forest
  - Input: Configuration
  - Output: Aggregate cost (for example PAR10)
- Optimizes the strategy for multiple instances
- Incumbent and candidate are always compared using only the instances on which they have both been run
- Parameter types: categorical, integer, real
- Acquisition function: EI
- Adaptive capping: a candidate is capped at 1.2 times the runtime of the incumbent

Related optimization tools

- ParamILS (Hutter et al. [2009]): algorithm configuration, model-free, only categorical and ordinal parameters
- BliStr (Urban [2015]): combines instance clustering with ParamILS
- HOS-ML (Holden and Korovin [2021]): combines instance clustering with SMAC, proposes schedules
- MaLeS (Kühlwein and Urban [2015]): dynamically schedules strategies with good predicted performance
- CPHydra (Bridge et al. [2012]): given an instance, produces a schedule of solvers

References I

Derek Bridge, Eoin O'Mahony, and Barry O'Sullivan. Case-based reasoning for autonomous constraint solving. In *Autonomous Search*, volume 9783642214349, pages 73–95. Springer-Verlag Berlin Heidelberg, oct 2012. ISBN 9783642214349. doi: 10.1007/978-3-642-21434-9_4. URL https://link.springer.com/chapter/10.1007/978-3-642-21434-9_4.

Edvard K. Holden and Konstantin Korovin. Heterogeneous Heuristic Optimisation and Scheduling for First-Order Theorem Proving. In *CICM 2021*, pages 107–123, Cham, jul 2021. Springer, Cham. doi: 10.1007/978-3-030-81097-9_8. URL https://link.springer.com/chapter/10.1007/978-3-030-81097-9_8.

References II

F. Hutter, H. H. Hoos, K. Leyton-Brown, and T. Stuetzle. ParamILS: An automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36:267–306, oct 2009. ISSN 1076-9757. doi: 10.1613/JAIR.2861. URL https://jair.org/index.php/jair/article/view/10628.

Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6683 LNCS, pages 507–523. Springer, Berlin, Heidelberg, 2011. ISBN 9783642255656. doi: 10.1007/978-3-642-25566-3_40. URL https://link.springer.com/chapter/10.1007/978-3-642-25566-3_40.

References III

Laura Kovács and Andrei Voronkov. First-order theorem proving and Vampire. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, volume 8044 of *Lecture Notes in Computer Science*, pages 1–35. Springer, 2013. ISBN 978-3-642-39798-1. doi: 10.1007/978-3-642-39799-8_1.

Daniel Kühlwein and Josef Urban. MaLeS: A framework for automatic tuning of automated theorem provers. *Journal of Automated Reasoning*, 55(2):91–116, August 2015. ISSN 15730670. doi: 10.1007/s10817-015-9329-1. URL `https://github.com/dkuehlwein/males`.

M. Lindauer, H. Hoos, F. Hutter, and T. Schaub. AutoFolio: An automatically configured algorithm selector. *Journal of Artificial Intelligence Research*, 53:745–778, 2015.

References IV

Marius Lindauer, Katharina Eggensperger, Matthias Feurer, Stefan Falkner, André Biedenkapp, and Frank Hutter. SMAC v3: Algorithm configuration in Python. https://github.com/automl/SMAC3, 2017.

Geoff Sutcliffe. The TPTP problem library and associated infrastructure. *Journal of Automated Reasoning*, 59(4), December 2017. ISSN 0168-7433. doi: 10.1007/s10817-017-9407-7.

Josef Urban. BliStr: The blind strategymaker. In *EPiC Series in Computer Science*, volume 36, pages 312–303. EasyChair, January 2015. doi: 10.29007/8n7m. URL http://www.tptp.org/CASC/J6/Design.html#CompetitionDivisions.

References V

Lin Xu, Holger H. Hoos, and Kevin Leyton-Brown. Hydra: Automatically configuring algorithms for portfolio-based selection. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, page 210–216. AAAI Press, 2010.