Configuration Report for the Solver PbO-CCSAT-Generic on the Training Instance Set PTN and Testing Instance Set PTN2 in *Sparkle*

Automatically generated by *Sparkle* (version: 1.0.0)

4th February 2021

1 Introduction

Sparkle [3] is a multi-agent problem-solving platform based on Programming by Optimisation (PbO) [2], and would provide a number of effective algorithm optimisation techniques (such as automated algorithm configuration, portfolio-based algorithm selection, etc) to accelerate the existing solvers.

This experimental report is automatically generated by *Sparkle*. This report presents experimental results on the scenario of configuring the solver PbO-CCSAT-Generic on the training instance set PTN and evaluating it on the testing instance set PTN2.

2 Information about the Instance Set(s)

- Training set: PTN, consisting of 12 instances
- Testing set: PTN2, consisting of 11 instances

3 Information about the Configuration Protocol

The configurator used in *Sparkle* is SMAC (*Sequential Model-based Algorithm Configuration*) [4], and the version of SMAC used in *Sparkle* is 2.10.03.

During the configuration process, *Sparkle* performs 25 independent SMAC runs for configuring the solver PbO-CCSAT-Generic on the trainging instance set PTN; the configuration objective is RUNTIME; the whole configuration time budget is 300 seconds; the cutoff time for each run is 10 seconds.

Each independent run of SMAC would result in one optimised configuration. As a result, *Sparkle* would obtain 25 optimised configurations. Each of these was then evaluated on the entire training set, with one solver run per instance and a cutoff time of 10 seconds, and the configuration with the lowest PAR10 value was selected as the result of the configuration process.

4 Information about the Optimised Configuration

After the configuration process mentioned above, *Sparkle* obtained the optimised configuration. The details of the optimised configuration are described as below.

-gamma_hscore2 '351' -init_solution '1' -p_swt '0.20423712003341465' -perform_aspiration '1' -perform_clause_weight '1' -perform_double_cc '0' -perform_first_div '0' -perform_pac '1' -prob_pac '0.005730374136488115' -q_swt '0.6807207179674418' -sel_clause_div '1' -sel_clause_weight_scheme '1' -sel_var_break_tie_greedy '4' -sel_var_div '2' -threshold_swt '32'

5 Comparison between Configured Version and Default Version on the Training Instance Set

In order to investigate the performance on the training instance set, *Sparkle* would run the configured version of PbO-CCSAT-Generic and the default version of PbO-CCSAT-Generic on the training instance set. During this phase, each version was performed one run per instance with a cutoff time of 10 seconds. The results are reported as follows.

- **PbO-CCSAT-Generic (configured)**, PAR10: 17.954468707243603
- **PbO-CCSAT-Generic (default)**, PAR10: 91.7627785205841

The empirical comparison between the PbO-CCSAT-Generic (configured) and PbO-CCSAT-Generic (default) on the training set of PTN is presented in Figure 1.

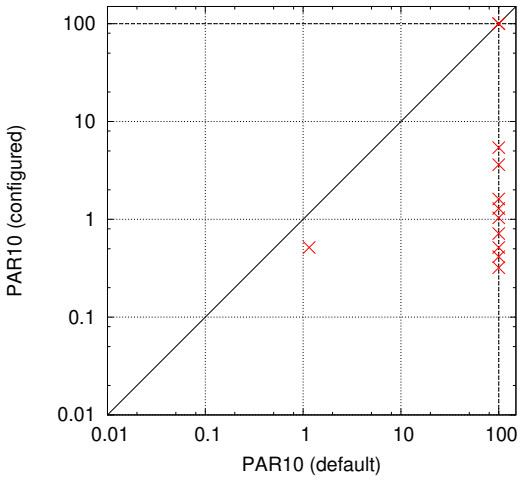

Figure 1: Empirical comparison between the PbO-CCSAT-Generic (configured) and PbO-CCSAT-Generic (default) on the training set of PTN.

Table 1 shows on how many instances the PbO-CCSAT-Generic (configured) and PbO-CCSAT-Generic (default) timed out (did not solve the instance within the cutoff time of 10 seconds) on the training set of PTN, as well as on how many instances both timed out.

6 Comparison between Configured Version and Default Version on the Testing Instance Set

After specifying the optimised configuration, *Sparkle* would run the configured version of PbO-CCSAT-Generic and the default version of PbO-CCSAT-Generic on the testing instance set. During this phase,

configured	default	overlap
2	11	2

Table 1: Number of time-outs for PbO-CCSAT-Generic (configured), PbO-CCSAT-Generic (default), and for how many instances both timed out on the training set of PTN.

each version was performed one run per instance with a cutoff time of 10 seconds. The results are reported as follows.

- **PbO-CCSAT-Generic (configured)**, PAR10: 11.336302540519021
- **PbO-CCSAT-Generic (default)**, PAR10: 91.48524641990662

The empirical comparison between the PbO-CCSAT-Generic (configured) and PbO-CCSAT-Generic (default) on the testing set of PTN2 is presented in Figure 2.

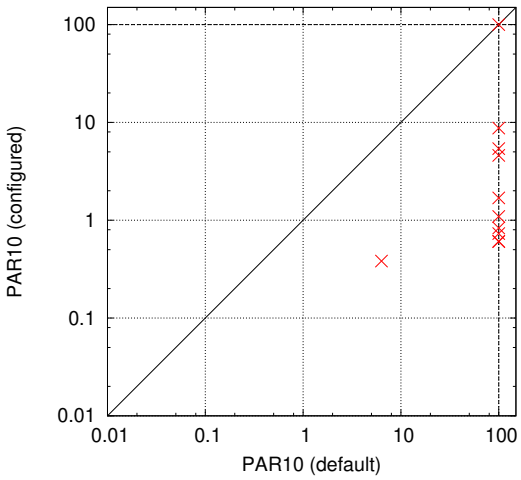

Figure 2: Empirical comparison between the PbO-CCSAT-Generic (configured) and PbO-CCSAT-Generic (default) on the testing set of PTN2.

Table 2 shows on how many instances the PbO-CCSAT-Generic (configured) and PbO-CCSAT-Generic (default) timed out (did not solve the instance within the cutoff time of 10 seconds) on the testing set of PTN2, as well as on how many instances both timed out.

configured	default	overlap
1	10	1

Table 2: Number of time-outs for PbO-CCSAT-Generic (configured), PbO-CCSAT-Generic (default), and for how many instances both timed out on the testing set of PTN2.

7 Parameter importance via Ablation

Ablation analysis [1] is performed from the PbO-CCSAT-Generic (default) to PbO-CCSAT-Generic (configured) to see which parameter changes between them contribute most to the improved perfor-

mance. The ablation path uses the training set PTN and validation is perform on the test set PTN2. The set of parameters that differ in the two configurations will form the ablation path. Starting from the default configuration, the path is computed by performing a sequence of rounds. In a round, each available parameter is flipped in the configuration and is validated on its performance. The flipped parameter with the best performance in that round, is added to the configuration and the next round starts with the remaining parameters. This repeats until all parameters are flipped, which is the best found configuration. The analysis resulted in the ablation path presented in Table 3.

Table 3: Ablation path from PbO-CCSAT-Generic (default) to PbO-CCSAT-Generic (configured) where parameters with higher importance are ranked higher.

Round	Flipped parameter	Source value	Target value	Validation result
0	-source-	N/A	N/A	91.33783
1	sel_var_div	3	2	38.75292
2	perform_pac	0	1	46.78325
3	q_swt	0.0	0.6807207179674418	28.73727
4	p_swt	0.3	0.20423712003341465	28.43357
5	threshold_swt	300	32	10.37413
6	perform_double_cc	1	0	10.42670
7	prob_pac	5.800000000000001E-4	0.005730374136488115	1.68256
8	sel_var_break_tie_greedy, gamma_hscore2	2, 1000	4, 351	19.45887
9	-target-	N/A	N/A	19.45937

References

[1] Chris Fawcett and Holger H. Hoos. Analysing differences between algorithm configurations through ablation. *J. Heuristics*, 22(4):431–458, 2016.

[2] Holger H. Hoos. Programming by optimization. *Communications of the ACM*, 55(2):70–80, 2012.

[3] Holger H. Hoos. Sparkle: A pbo-based multi-agent problem-solving platform. Technical report, Department of Computer Science, University of British Columbia, 2015.

[4] 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 5)*, pages 507–523, 2011.