Experimental Reports for *Sparkle*

Sparkle

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 for the training instances submitted to *Sparkle*, as well as for running the *Sparkle* portfolio selector to solve the test instances in the instance set **PTN2**.

2 Experimental Preliminaries

This section presents the experimental preliminaries, including the list of solvers, the list of feature extractors, the list of instance sets, information about the experimental setup and information about how to construct a portfolio-based algorithm selector in *Sparkle*.

2.1 Solvers

There are 3 solver(s) included in *Sparkle*, as listed below.

1. **CSCCSat**
2. **Lingeling**
3. **MiniSAT**

2.2 Feature Extractors

There are 1 feature extractor(s) included in *Sparkle*, as listed below.

1. **SAT-features-competition2012_revised_without_SatELite_sparkle**

2.3 Training Instance Set(s)

There are 1 instance set(s) included in *Sparkle*, as listed below.

1. **PTN**, number of instances: 12

2.4 Test Instance Set

- Testing set: **PTN2**, consisting of 11 instances

2.5 Experimental Setup for Training Phase

The experimental setup for the training phase is described below.

Feature computation: *Sparkle* uses all the feature extractors presented above to compute a feature vector for each training instance. Every feature extractor computes a feature vector for each training instance. The final feature vector is then the combination of all computed feature vectors. The cutoff time for feature vector computation on each training instance is set to 60 seconds. This time is shared equally between the feature extractors.

Performance computation: *Sparkle* runs each solver one time on each training instance. The cutoff time for each performance computation run is set to 10 seconds.

2.6 Constructing the Portfolio-Based Algorithm Selector

Sparkle saves the results of feature extraction and performance computation described above. This data is then utilized by *Sparkle* to run *AutoFolio* [4] to automatically construct a portfolio-based algorithm selector for *Sparkle*.

3 Experimental Results on the Training Set(s)

In this section, the PAR10 results for the current portfolio selector in *Sparkle* on solving the training instance set(s) listed in Section 2.3 is reported.

3.1 PAR10 Ranking List

Below, the solvers are ranked based on the penalised average runtime (PAR10).

1. **Solvers/CSCCSat**, PAR10: 83.38584091666667
2. **Solvers/Lingeling**, PAR10: 100.0
3. **Solvers/MiniSAT**, PAR10: 100.0

Next, the PAR10 values for the Virtual Best Solver *VBS*, i.e., the perfect portfolio selector, and the actual portfolio selector in *Sparkle* are given.

- ***VBS***, PAR10: 83.38584091666667
- **Actual Portfolio Selector in *Sparkle***, PAR10: 83.38584091666667

3.2 Marginal Contribution Ranking List

Sparkle uses the concept of marginal contribution [5] to measure each solver's contribution to the ***VBS*** and to the **actual portfolio selector in *Sparkle***. In this report, the approach described in the literature [1] is used to compute each solver's marginal contribution.

This resulted in the following ranking of solvers for the ***VBS***:

1. **Solvers/CSCCSat**, marginal contribution: 2.0536562576177286
2. **Solvers/Lingeling**, marginal contribution: 0.0
3. **Solvers/MiniSAT**, marginal contribution: 0.0

And the following ranking for the **actual portfolio selector in *Sparkle***:

1. **Solvers/CSCCSat**, marginal contribution: 2.0536562576177286
2. **Solvers/Lingeling**, marginal contribution: 0.0
3. **Solvers/MiniSAT**, marginal contribution: 0.0

3.3 Scatter Plot Analysis

Figure 1 shows the empirical comparison between the actual portfolio selector in *Sparkle* and the single best solver (*SBS*). Figure 2 shows the empirical comparison between the actual portfolio selector in *Sparkle* and the virtual best solver (*VBS*).

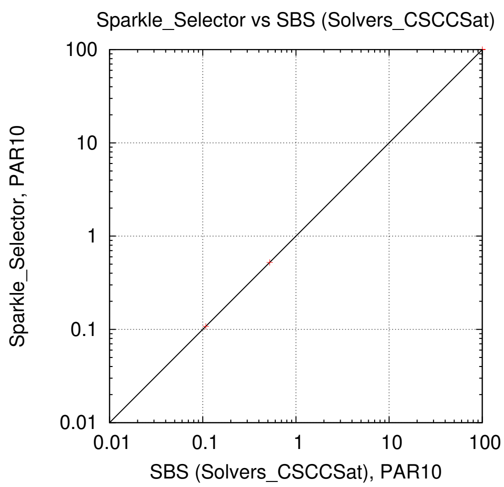

Figure 1: Empirical comparison between the actual portfolio selector in *Sparkle* and the *SBS*.

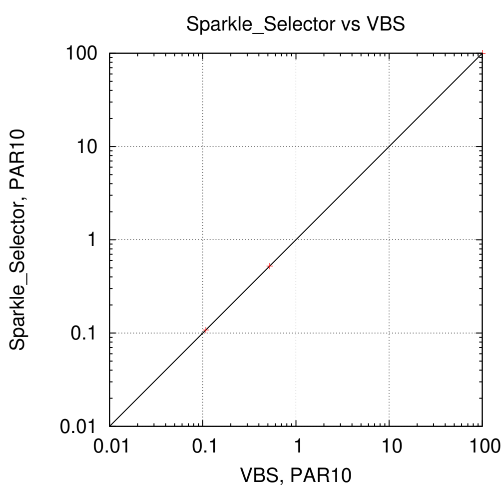

Figure 2: Empirical comparison between the actual portfolio selector in *Sparkle* and the *VBS*.

4 Experimental Results on the Test Set

In this section, the PAR10 results for the current portfolio selector in *Sparkle* on solving the test instance set **PTN2** is reported.

- **Actual Portfolio Selector in *Sparkle***, PAR10: 73.23723045454545

References

[1] Alexandre Fréchette, Lars Kotthoff, Tomasz P. Michalak, Talal Rahwan, Holger H. Hoos, and Kevin Leyton-Brown. Using the shapley value to analyze algorithm portfolios. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 3397–3403. AAAI Press, 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] Marius Thomas Lindauer, Holger H. Hoos, Frank Hutter, and Torsten Schaub. AutoFolio: An automatically configured algorithm selector. *Journal of Artificial Intelligence Research*, 53:745–778, 2015.

[5] Lin Xu, Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. Evaluating component solver contributions to portfolio-based algorithm selectors. In Alessandro Cimatti and Roberto Sebastiani, editors, *Theory and Applications of Satisfiability Testing - SAT 2012 - 15th International Conference, Trento, Italy, June 17-20, 2012. Proceedings*, volume 7317 of *Lecture Notes in Computer Science*, pages 228–241. Springer, 2012.