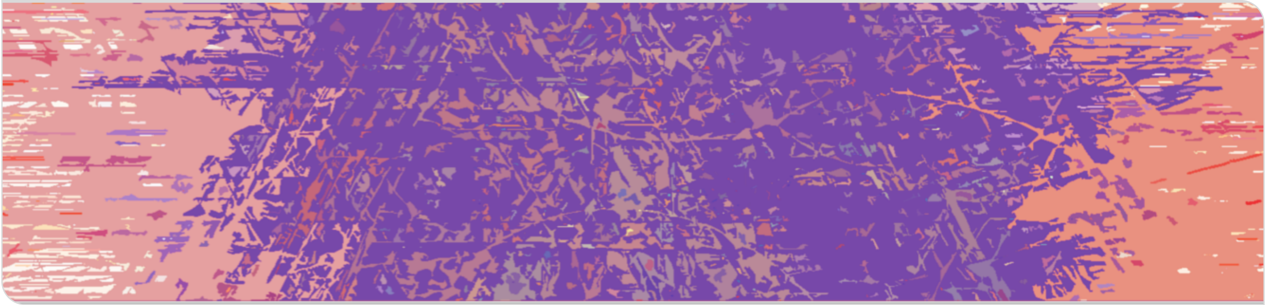
Karlsruhe Institute of Technology

A Comprehensive Study of k-Portfolios of Recent SAT Solvers

SAT 2022 | Haifa, Israel

Jakob Bach, Markus Iser, and Klemens Böhm | August 2, 2022



Motivation

Motivation

- Solvers often exhibit complementarity in benchmarks, e.g., in Main Track of SAT Competitions:
  - 2020: 316 instances solved; 2021: 325 instances solved

Motivation

Karlsruhe Institute of Technology

- Solvers often exhibit complementarity in benchmarks, e.g., in Main Track of SAT Competitions:
  - 2020: 316 instances solved; 2021: 325 instances solved
  - On how many instances did the best individual solvers win?

	Instances won	
Solver #	2020	2021
1	46	25
2	38	22
3	26	20

Motivation

- Solvers often exhibit complementarity in benchmarks, e.g., in Main Track of SAT Competitions:
  - 2020: 316 instances solved; 2021: 325 instances solved
  - On how many instances did the best individual solvers win?

	Instances won	
Solver #	2020	2021
1	46	25
2	38	22
3	26	20

- SAT Competition: 2021: Special Innovation Price for *CaDiCaL_PriPro*
  - Place 10 in Main Track, but part of best two-solver portfolio ...

Motivation

- Solvers often exhibit complementarity in benchmarks, e.g., in Main Track of SAT Competitions:
  - 2020: 316 instances solved; 2021: 325 instances solved
  - On how many instances did the best individual solvers win?

	Instances won	
Solver #	2020	2021
1	46	25
2	38	22
3	26	20

- SAT Competition: 2021: Special Innovation Price for *CaDiCaL_PriPro*
  - Place 10 in Main Track, but part of best two-solver portfolio ...
  - ... together with *lstech_maple* (Place 13 in Main Track)

Problem Definition

Definition (K-Portfolio Problem)

Problem Definition

Karlsruhe Institute of Technology

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \ldots, s_n\}$,

Problem Definition

Karlsruhe Institute of Technology

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \dots, s_n\}$,
- a set of SAT instances $I = \{i_1, \dots, i_l\}$,

Problem Definition

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \ldots, s_n\}$,
- a set of SAT instances $I = \{i_1, \ldots, i_l\}$,
- a scoring function $c : I \times S \to \mathbb{R}$ (here: PAR-2 score),

Problem Definition

Karlsruhe Institute of Technology

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \ldots, s_n\}$,
- a set of SAT instances $I = \{i_1, \ldots, i_l\}$,
- a scoring function $c : I \times S \to \mathbb{R}$ (here: PAR-2 score),
- an instance-specific solver selector $m : I \to S$, and

Problem Definition

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \ldots, s_n\}$,
- a set of SAT instances $I = \{i_1, \ldots, i_l\}$,
- a scoring function $c : I \times S \to \mathbb{R}$ (here: PAR-2 score),
- an instance-specific solver selector $m : I \to S$, and
- a portfolio size $k \in \mathbb{N}$,

Problem Definition

Karlsruhe Institute of Technology

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \ldots, s_n\}$,
- a set of SAT instances $I = \{i_1, \ldots, i_l\}$,
- a scoring function $c : I \times S \to \mathbb{R}$ (here: PAR-2 score),
- an instance-specific solver selector $m : I \to S$, and
- a portfolio size $k \in \mathbb{N}$,

find a solver subset P of size k with minimum average cost: $\underset{P \subseteq S, |P|=k}{\arg\min} \frac{1}{|I|} \cdot \sum_{i \in I} c(i, m(i))$

Problem Definition

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \ldots, s_n\}$,
- a set of SAT instances $I = \{i_1, \ldots, i_l\}$,
- a scoring function $c : I \times S \to \mathbb{R}$ (here: PAR-2 score),
- an instance-specific solver selector $m : I \to S$, and
- a portfolio size $k \in \mathbb{N}$,

find a solver subset P of size k with minimum average cost: $\underset{P \subseteq S, |P| = k}{\arg\min} \frac{1}{|I|} \cdot \sum_{i \in I} c(i, m(i))$

- We analyze two methods for instance-specific solver selection m:
  - Virtual Best Solver (VBS): Oracle always selects best solver

Problem Definition

Definition (K-Portfolio Problem)

Given

- a set of solvers $S = \{s_1, \ldots, s_n\}$,
- a set of SAT instances $I = \{i_1, \ldots, i_l\}$,
- a scoring function $c : I \times S \to \mathbb{R}$ (here: PAR-2 score),
- an instance-specific solver selector $m : I \to S$, and
- a portfolio size $k \in \mathbb{N}$,

find a solver subset P of size k with minimum average cost: $\underset{P \subseteq S, |P|=k}{\arg\min} \frac{1}{|I|} \cdot \sum_{i \in I} c(i, m(i))$

- We analyze two methods for instance-specific solver selection m:
  - Virtual Best Solver (VBS): Oracle always selects best solver
  - Model-based: Prediction model selects solver based on instance features

Related Work

- Analyzing solver complementarity: Xu et al. [21], Fréchette et al. [10]

Related Work

- Analyzing solver complementarity: Xu et al. [21], Fréchette et al. [10]
- Instance-specific solver selection: SATzilla [23, 24], ISAC [13], SNNAP [8], etc.

Related Work

- Analyzing solver complementarity: Xu et al. [21], Fréchette et al. [10]
- Instance-specific solver selection: SATzilla [23, 24], ISAC [13], SNNAP [8], etc.
- Analyzing *k*-portfolios for CSPs: Amadini et al. [1, 2], Dang [9], Ulrich-Oltean et al. [20]

Related Work

- Analyzing solver complementarity: Xu et al. [21], Fréchette et al. [10]
- Instance-specific solver selection: SATzilla [23, 24], ISAC [13], SNNAP [8], etc.
- Analyzing *k*-portfolios for CSPs: Amadini et al. [1, 2], Dang [9], Ulrich-Oltean et al. [20]
- Analyzing *k*-portfolios for anytime algorithms: Nof and Strichman [17]

Solution Approaches – Overview

- K-Portfolio Problem with VBS selector is NP-complete, but also monotone and submodular [17]

Solution Approaches – Overview

Karlsruhe Institute of Technology

- K-Portfolio Problem with VBS selector is NP-complete, but also monotone and submodular [17]

- Examples for exact (i.e., optimal) solutions:
  - Exhaustive search: evaluates $\binom{|S|}{k}$ portfolios

Solution Approaches – Overview

- K-Portfolio Problem with VBS selector is NP-complete, but also monotone and submodular [17]
- Examples for exact (i.e., optimal) solutions:
  - Exhaustive search: evaluates $\binom{|S|}{k}$ portfolios
  - Encoding with Satisfiability Module Theories (SMT) by Nof and Strichman [17]

Solution Approaches – Overview

- K-Portfolio Problem with VBS selector is NP-complete, but also monotone and submodular [17]

- Examples for exact (i.e., optimal) solutions:
  - Exhaustive search: evaluates $\binom{|S|}{k}$ portfolios
  - Encoding with Satisfiability Module Theories (SMT) by Nof and Strichman [17]
  - Encoding as integer linear program developed by us

Solution Approaches – Overview

- K-Portfolio Problem with VBS selector is NP-complete, but also monotone and submodular [17]

- Examples for exact (i.e., optimal) solutions:
  - Exhaustive search: evaluates $\binom{|S|}{k}$ portfolios
  - Encoding with Satisfiability Module Theories (SMT) by Nof and Strichman [17]
  - Encoding as integer linear program developed by us

- Examples for heuristic solutions:
  - Beam search with beam width w: evaluates $O(|S| \cdot w \cdot k)$ portfolios
    - Submodularity bounds quality of greedy search ($w = 1$) relative to optimal solution [16, 17]

Solution Approaches – Overview

Karlsruhe Institute of Technology

- K-Portfolio Problem with VBS selector is NP-complete, but also monotone and submodular [17]

- Examples for exact (i.e., optimal) solutions:
  - Exhaustive search: evaluates $\binom{|S|}{k}$ portfolios
  - Encoding with Satisfiability Module Theories (SMT) by Nof and Strichman [17]
  - Encoding as integer linear program developed by us

- Examples for heuristic solutions:
  - Beam search with beam width w: evaluates $O(|S| \cdot w \cdot k)$ portfolios
    - Submodularity bounds quality of greedy search ($w = 1$) relative to optimal solution [16, 17]
  - K-best [17]: evaluates $O(|S|)$ portfolios

Solution Approaches – Integer Linear Program

$$\forall s \in S: \qquad y_s \in \{0, 1\} \qquad \text{(solver selected or not)}$$

Solution Approaches – Integer Linear Program

$$\text{s.t.} \quad \sum_{s \in S} y_s \leq k \qquad \text{(portfolio size)}$$

$$\forall s \in S: \quad y_s \in \{0, 1\} \qquad \text{(solver selected or not)}$$

Solution Approaches – Integer Linear Program

Karlsruhe Institute of Technology

$$\min_{x,y} \quad \frac{1}{|I|} \cdot \sum_{i \in I} \sum_{s \in S} c(i, s) \cdot x_{i,s}$$

$$\text{s.t.} \quad \sum_{s \in S} y_s \leq k \qquad \text{(portfolio size)}$$

$$\forall i \in I, \forall s \in S: \quad x_{i,s} \in \{0, 1\} \qquad \text{(solver selected for instance or not)}$$

$$\forall s \in S: \quad y_s \in \{0, 1\} \qquad \text{(solver selected or not)}$$

Solution Approaches – Integer Linear Program

$$\min_{x,y} \quad \frac{1}{|I|} \cdot \sum_{i \in I} \sum_{s \in S} c(i,s) \cdot x_{i,s}$$

$$\text{s.t.} \qquad \sum_{s \in S} y_s \leq k \qquad \text{(portfolio size)}$$

$$\forall i \in I: \quad \sum_{s \in S} x_{i,s} = 1 \qquad \text{(one solver per instance)}$$

$$\forall i \in I, \forall s \in S: \quad x_{i,s} \in \{0,1\} \qquad \text{(solver selected for instance or not)}$$

$$\forall s \in S: \quad y_s \in \{0,1\} \qquad \text{(solver selected or not)}$$

Solution Approaches – Integer Linear Program

$$
\begin{aligned}
\min_{x,y} \quad & \frac{1}{|I|} \cdot \sum_{i \in I} \sum_{s \in S} c(i,s) \cdot x_{i,s} & \\
\text{s.t.} \quad & \sum_{s \in S} y_s \leq k & \text{(portfolio size)} \\
\forall i \in I: \quad & \sum_{s \in S} x_{i,s} = 1 & \text{(one solver per instance)} \\
\forall s \in S: \quad & \sum_{i \in I} x_{i,s} \leq |I| \cdot y_s & \text{(only use solvers from portfolio)} \\
\forall i \in I, \forall s \in S: \quad & x_{i,s} \in \{0,1\} & \text{(solver selected for instance or not)} \\
\forall s \in S: \quad & y_s \in \{0,1\} & \text{(solver selected or not)}
\end{aligned}
$$

Experimental Design

- Two datasets (from Main Tracks of recent SAT Competitions):
  1) *SC2020* (316 instances, 48 solvers) [4]
  2) *SC2021* (325 instances, 46 solvers) [5]
  - 138 features from feature extractor of SATzilla 2012 [22, 24]
  - Instance features and solver runtimes retrieved from Global Benchmark Database (GBD) [12]

Experimental Design

- Two datasets (from Main Tracks of recent SAT Competitions):
  1) *SC2020* (316 instances, 48 solvers) [4]
  2) *SC2021* (325 instances, 46 solvers) [5]
  - 138 features from feature extractor of SATzilla 2012 [22, 24]
  - Instance features and solver runtimes retrieved from Global Benchmark Database (GBD) [12]

- Four solution approaches:
  - *Optimal solution* via integer programming [19]
  - *Beam search* with beam width $w \in \{1, 2, 3, \ldots, 10, 20, 30, \ldots, 100\}$
  - *K-best*
  - *Random sampling* with 1000 repetitions

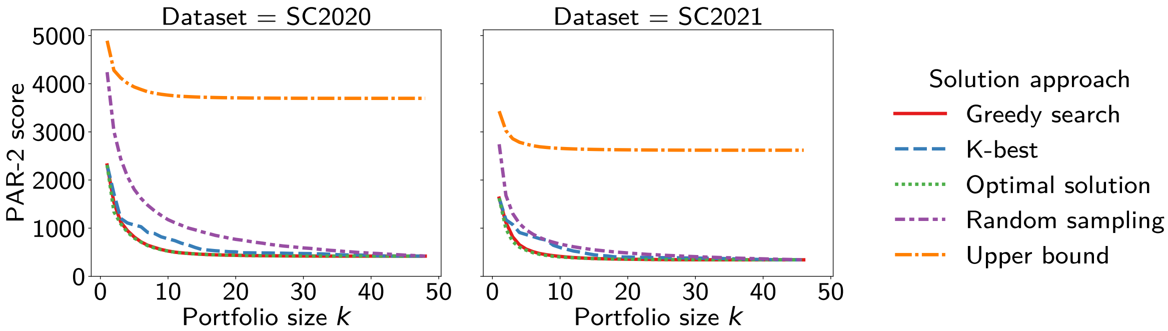
Experimental Design

- Two datasets (from Main Tracks of recent SAT Competitions):
  1) *SC2020* (316 instances, 48 solvers) [4]
  2) *SC2021* (325 instances, 46 solvers) [5]
  - 138 features from feature extractor of SATzilla 2012 [22, 24]
  - Instance features and solver runtimes retrieved from Global Benchmark Database (GBD) [12]

- Four solution approaches:
  - *Optimal solution* via integer programming [19]
  - *Beam search* with beam width $w \in \{1, 2, 3, \ldots, 10, 20, 30, \ldots, 100\}$
  - *K-best*
  - *Random sampling* with 1000 repetitions

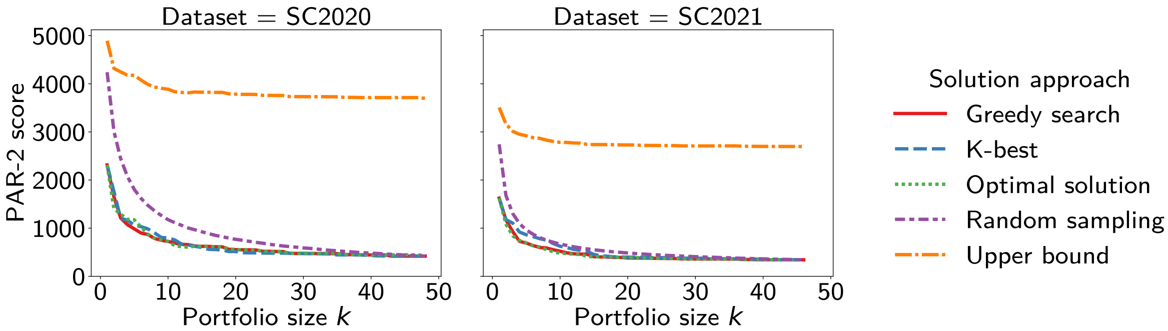
- Two multi-class prediction models: Random forests [6, 18] and XGBoost [7] with 100 trees each

Results – Portfolio Search (VBS on Training Set)



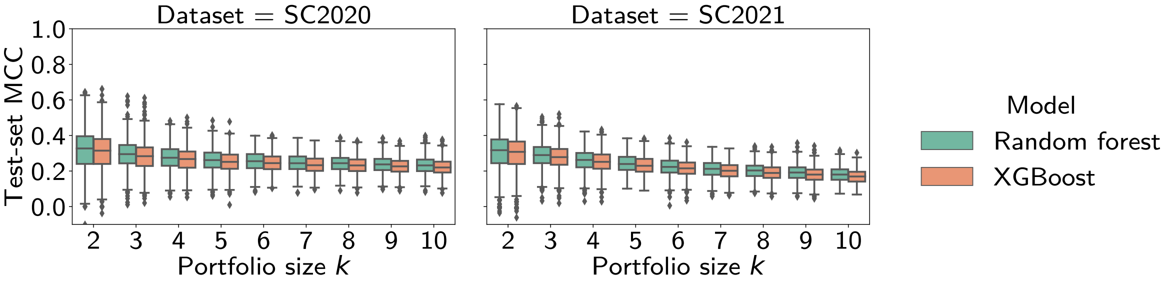
Training-set VBS performance for different datasets, values of k, and portfolio-search approaches.

Results – Portfolio Search (VBS on Test Set)



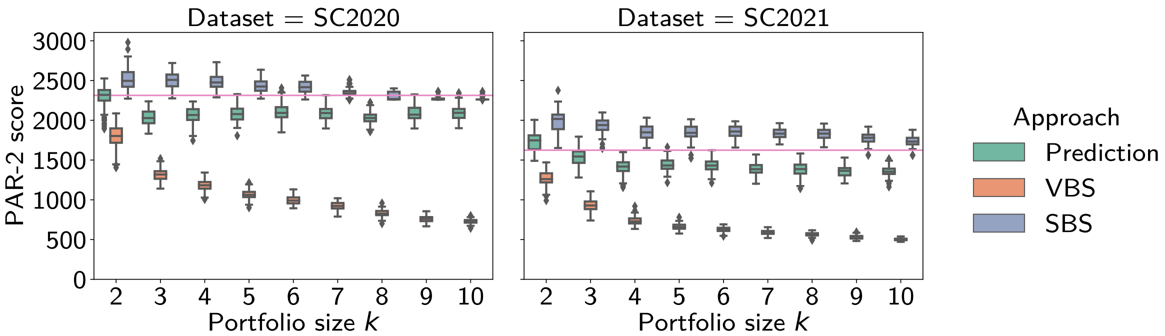
Test-set VBS performance for different datasets, values of k, and portfolio-search approaches.

Results – Recommending Solvers (MCC)



Test-set prediction performance in terms of Matthews correlation coefficient (MCC) [15, 11] for different datasets, values of k, and prediction models. Randomly sampled portfolios.

Results – Recommending Solvers (PAR-2 Score)



Test-set solver performance for different datasets, values of k, and solver-recommendation approaches. Global SBS pictured as horizontal line. Portfolios from *beam search* with $w = 100$. Random forests for predictions.

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers
- Greedy portfolio search close to optimal portfolio

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers
- Greedy portfolio search close to optimal portfolio
- K-best (only considering individual solver performance) worse than greedy and optimal portfolio

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers
- Greedy portfolio search close to optimal portfolio
- K-best (only considering individual solver performance) worse than greedy and optimal portfolio
- Our prediction approach does not benefit from increased portfolio size

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers
- Greedy portfolio search close to optimal portfolio
- K-best (only considering individual solver performance) worse than greedy and optimal portfolio
- Our prediction approach does not benefit from increased portfolio size

- Directions for future work:
  - Improve prediction performance, e.g., by using new features like community-based ones [3, 14]

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers
- Greedy portfolio search close to optimal portfolio
- K-best (only considering individual solver performance) worse than greedy and optimal portfolio
- Our prediction approach does not benefit from increased portfolio size

- Directions for future work:
  - Improve prediction performance, e.g., by using new features like community-based ones [3, 14]
  - Analyze special-purpose solvers and solver configurations

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers
- Greedy portfolio search close to optimal portfolio
- K-best (only considering individual solver performance) worse than greedy and optimal portfolio
- Our prediction approach does not benefit from increased portfolio size

- Directions for future work:
  - Improve prediction performance, e.g., by using new features like community-based ones [3, 14]
  - Analyze special-purpose solvers and solver configurations
  - Tune solvers within portfolios [13]

Summary and Future Work

- Evaluated solver portfolios on data from SAT Competitions 2020 and 2021
- Small portfolios already show potential of high runtime improvement compared to individual solvers
- Greedy portfolio search close to optimal portfolio
- K-best (only considering individual solver performance) worse than greedy and optimal portfolio
- Our prediction approach does not benefit from increased portfolio size

- Directions for future work:
  - Improve prediction performance, e.g., by using new features like community-based ones [3, 14]
  - Analyze special-purpose solvers and solver configurations
  - Tune solvers within portfolios [13]
  - Compare to sophisticated portfolio approaches like SATzilla [23, 24]

References I

[1] Roberto Amadini, Maurizio Gabbrielli, and Jacopo Mauro. "An Empirical Evaluation of Portfolios Approaches for Solving CSPs". In: *Proc. CPAIOR*. Yorktown Heights, NY, USA, 2013, pp. 316–324. DOI: 10.1007/978-3-642-38171-3_21.

[2] Roberto Amadini, Maurizio Gabbrielli, and Jacopo Mauro. "An Extensive Evaluation of Portfolio Approaches for Constraint Satisfaction Problems". In: *Int. J. Interact. Multim. Artif. Intell.* 3.7 (2016), pp. 81–86. DOI: 10.9781/ijimai.2016.3712.

[3] Carlos Ansótegui et al. "Community Structure in Industrial SAT Instances". In: *J. Artif. Intell. Res.* 66 (2019), pp. 443–472. DOI: 10.1613/jair.1.11741.

[4] Tomáš Balyo et al., eds. *Proceedings of SAT Competition 2020: Solver and Benchmark Descriptions*. Department of Computer Science, University of Helsinki, 2020. URL: http://hdl.handle.net/10138/318754.

References II

[5] Tomáš Balyo et al., eds. *Proceedings of SAT Competition 2021: Solver and Benchmark Descriptions*. Department of Computer Science, University of Helsinki, 2021. URL: http://hdl.handle.net/10138/333647.

[6] Leo Breiman. “Random Forests”. In: *Mach. Learn.* 45.1 (2001), pp. 5–32. DOI: 10.1023/A:1010933404324.

[7] Tianqi Chen and Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. In: *Proc. KDD*. San Francisco, CA, USA, 2016, pp. 785–794. DOI: 10.1145/2939672.2939785.

[8] Marco Collautti et al. “SNNAP: Solver-Based Nearest Neighbor for Algorithm Portfolios”. In: *Proc. ECML PKDD*. Prague, Czech Republic, 2013, pp. 435–450. DOI: 10.1007/978-3-642-40994-3_28.

[9] Nguyen Dang. “A portfolio-based analysis method for competition results”. In: *Proc. ModRef*. Haifa, Israel, 2022. DOI: 10.48550/arXiv.2205.15414.

References III

[10] Alexandre Fréchette et al. “Using the Shapley Value to Analyze Algorithm Portfolios”. In: *Proc. AAAI*. Phoenix, AZ, USA, 2016, pp. 3397–3403. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/10440`.

[11] Jan Gorodkin. “Comparing two K-category assignments by a K-category correlation coefficient”. In: *Comput. Biol. Chem.* 28.5–6 (2004), pp. 367–374. DOI: `10.1016/j.compbiolchem.2004.09.006`.

[12] Markus Iser, Luca Springer, and Carsten Sinz. “Collaborative Management of Benchmark Instances and their Attributes”. In: *arXiv preprint arXiv:2009.02995* (2020). URL: `https://arxiv.org/abs/2009.02995`.

[13] Serdar Kadioglu et al. “ISAC – Instance-Specific Algorithm Configuration”. In: *Proc. ECAI*. Lisbon, Portugal, 2010, pp. 751–756. DOI: `10.3233/978-1-60750-606-5-751`.

[14] Chunxiao Li et al. “On the Hierarchical Community Structure of Practical SAT Formulas”. In: *Proc. SAT*. Barcelona, Spain, 2021. DOI: `10.1007/978-3-030-80223-3_25`.

References IV

[15] Brian W. Matthews. "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". In: *Biochim. Biophys. Acta - Protein Struct.* 405.2 (1975), pp. 442–451. DOI: 10.1016/0005-2795(75)90109-9.

[16] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. "An analysis of approximations for maximizing submodular set functions - I". In: *Math. Program.* 14.1 (1978), pp. 265–294. DOI: 10.1007/BF01588971.

[17] Yair Nof and Ofer Strichman. "Real-time solving of computationally hard problems using optimal algorithm portfolios". In: *Ann. Math. Artif. Intell.* (2021), pp. 1–18. DOI: 10.1007/s10472-020-09704-4.

[18] Fabian Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *J. Mach. Learn. Res.* 12.85 (2011), pp. 2825–2830. URL: http://jmlr.org/papers/v12/pedregosa11a.html.

[19] Haroldo G. Santos and Túlio A. M. Toffolo. *Python-MIP*. June 1, 2021. URL: https://python-mip.com/.

References V

[20] Felix Ulrich-Oltean, Peter Nightingale, and James Alfred Walker. "Selecting SAT Encodings for Pseudo-Boolean and Linear Integer Constraints". In: *Proc. CP*. Haifa, Israel, 2022, 38:1–38:17. DOI: 10.4230/LIPIcs.CP.2022.38.

[21] Lin Xu et al. "Evaluating Component Solver Contributions to Portfolio-Based Algorithm Selectors". In: *Proc. SAT*. Trento, Italy, 2012, pp. 228–241. DOI: 10.1007/978-3-642-31612-8_18.

[22] Lin Xu et al. *Features for SAT*. Tech. rep. University of British Columbia, 2012. URL: https://www.cs.ubc.ca/labs/beta/Projects/SATzilla/Report_SAT_features.pdf.

[23] Lin Xu et al. "SATzilla: Portfolio-based Algorithm Selection for SAT". In: *J. Artif. Intell. Res.* 32 (2008), pp. 565–606. DOI: 10.1613/jair.2490.

[24] Lin Xu et al. "SATzilla2012: Improved Algorithm Selection Based on Cost-sensitive Classification Models". In: *Proc. SAT Challenge*. 2012, pp. 57–58. URL: http://hdl.handle.net/10138/34218.