Selecting SAT Encodings for Pseudo-Boolean and Linear Integer Constraints

Felix Ulrich-Oltean (**felix.ulrich-oltean@york.ac.uk**), Peter Nightingale, James Walker

University of York

50 YEARS OF COMPUTER SCIENCE AT YORK

www.york.ac.uk/computerscience

WHY?

- How we **represent** a problem is important.
- Some constraints have many SAT encodings available, but it's **hard to choose** the best one for a CSP.
- **Machine learning** can help us make a good choice.

WHAT?

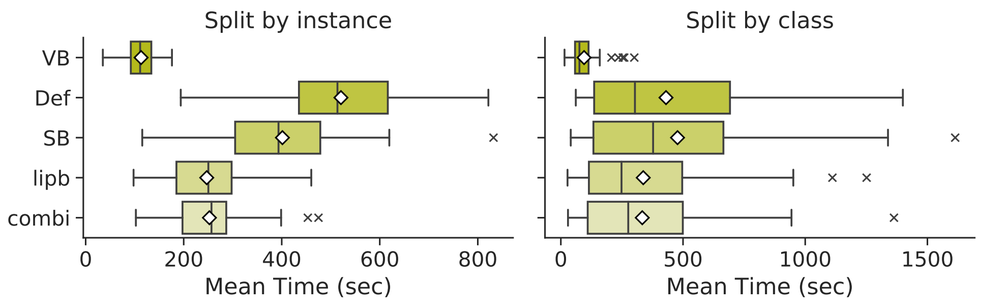
- We implement and evaluate an **ML approach** to predicting good SAT encodings of Pseudo-Boolean and Linear Integer constraints.
- We introduce new **constraint-specific features**.
- We calculate and discuss **feature importance**.

HOW?

- Prepare a **varied corpus** of instances from many different problem classes.
- Create a **reduced-size portfolio** of encoding configurations across the two constraint types.
- Train random forest classifiers on **pairs** of configurations, then vote at prediction time.

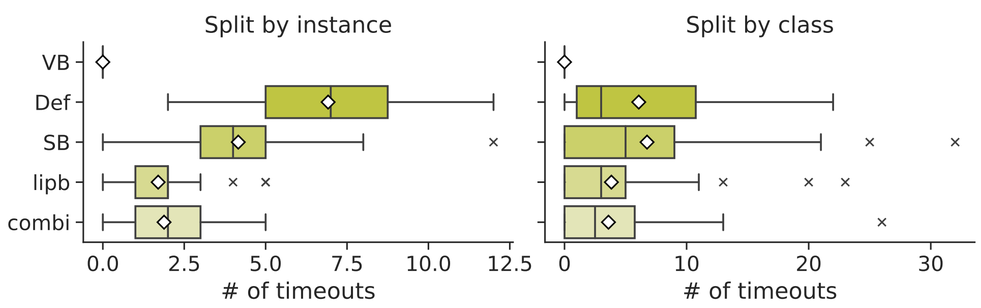
SO?

- We **narrowed the gap** between Single Best and Virtual Best by up to 38% for **unseen problem classes** and 54% for seen classes.



Distribution of run times across 50 cycles of split, train, predict

- Our new features made selection **more robust**, with fewer timeouts.



Distribution of timeouts across 50 cycles of split, train, predict

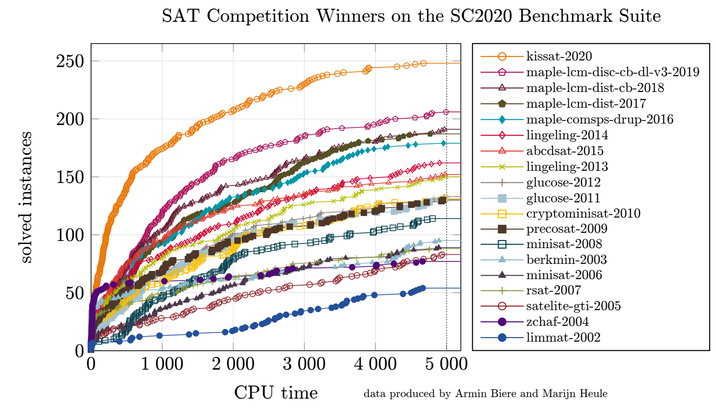
- We greatly outperformed **AutoFolio** [2], a sophisticated Algorithm Selection and Configuration tool.

Reference			Prediction	
VB	SB	Def	AutoFolio	Our ML
1.00	21.91	18.99	20.01	**13.75**

PAR10 mean predicted times relative to Virtual Best, using libp features

Why Solve CSPs with SAT?

Most recent CSP solving competitions have been won by solvers which encode to SAT (see the MiniZinc Challenge and the XCSP competition). We can also continue to benefit from advances in SAT solving. The plot (right) from fmv.jku.at/kissat shows that SAT solvers are able to solve more problems year-on-year.



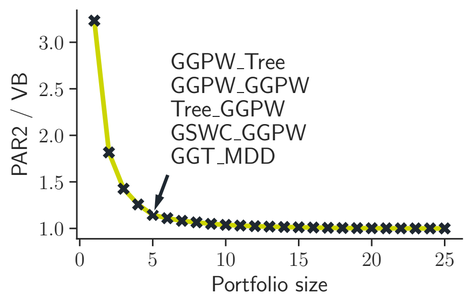
What is a SAT Encoding?

A SAT encoding is a scheme to produce a Boolean (SAT) formula which represent a CSP, or part of a CSP. Encodings use different structures to represent the underlying constraint, setting the SAT variables accordingly to ensure that the SAT clauses are satisfied if and only if the original CSP constraint is satisfied.

We focus on pseudo-Boolean and linear integer constrains, using the *GGT*, *GGPW*, *GSWC* and *MDD* encodings [1], as well as Savile Row's default *Tree* encoding [3].

How is Pairwise Training Made Feasible?

With 25 configuration options (5 PB $\times$ 5 LI), we would need to train $\binom{25}{2} = 300$ classifiers. This quickly grows infeasible with more constraint types, and more choices of encoding. We grow a portfolio of configurations by adding in choices which reduce the VB runtime the most. The plot below shows that with 5 configurations, we can get within 14% of the performance achievable with the full set of encodings.

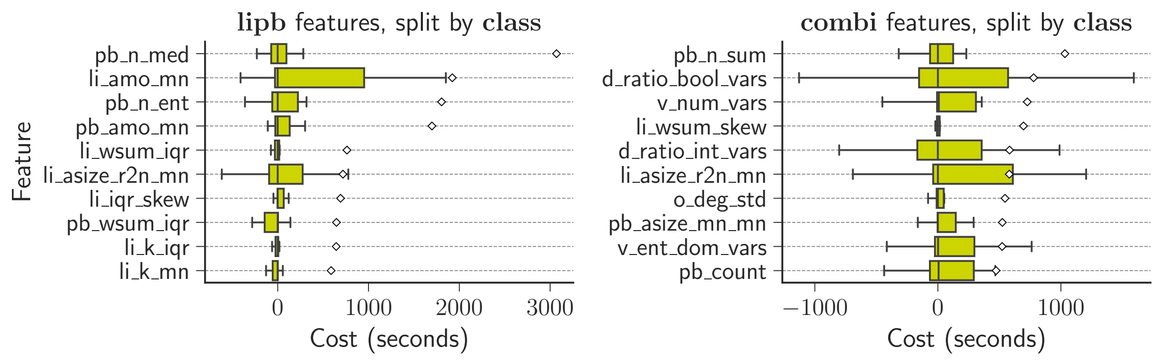


PAR2 runtime as proportion of Virtual Best for different sized portfolios

Which Features Matter?

Generic instance features enable good predictions which outperform the single best choice of encodings from the training set. But the introduction of features **specific** to PB/LI constraints gives better results, with a lower mean runtime and fewer timeouts.

The plot below shows the 10 most important features in the specialised *lipb* featureset and the *combi* featureset which additionally contains the generic features. We see that both generic and specialised features (prefixed `li_` and `pb_`) are used by the ML classifier in equal measure.



Permutation feature importance over 50 split-train-predic cycles

References

[1] Miquel Bofill, Jordi Coll, Josep Suy, and Mateu Villaret. SAT encodings of pseudo-boolean constraints with at-most-one relations. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 112–128. Springer, 2019.

[2] Marius Lindauer, Holger H. Hoos, Frank Hutter, and Torsten Schaub. AutoFolio: An Automatically Configured Algorithm Selector. *Journal of Artificial Intelligence Research*, 53:745–778, August 2015.

[3] Peter Nightingale, Özgür Akgün, Ian P. Gent, Christopher Jefferson, Ian Miguel, and Patrick Spracklen. Automatically improving constraint models in Savile Row. *Artificial Intelligence*, 251:35–61, October 2017.

Find Out More

Go to https://felixvuo.github.io or scan the QR code for:

- the full paper
- the data for the experiments
- getting in touch

We used the Savile Row modelling assistant. More details and software at savilerow.cs.st-andrews.ac.uk.