Felix Ulrich-Oltean

Supervised by
Peter Nightingale and **James Walker**

PhD student @ University of York

fvuo500@york.ac.uk, twitter.com/FelixVUO, felixvuo.github.io

Supported by grant EP/R513386/1 from the UK Engineering and Physical Sciences Research Council.

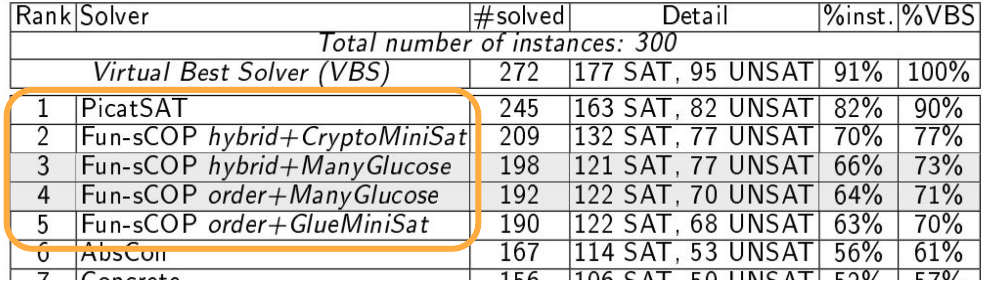
Learning to Choose SAT Encodings for Pseudo-Boolean and Integer Sum Constraints

This Talk

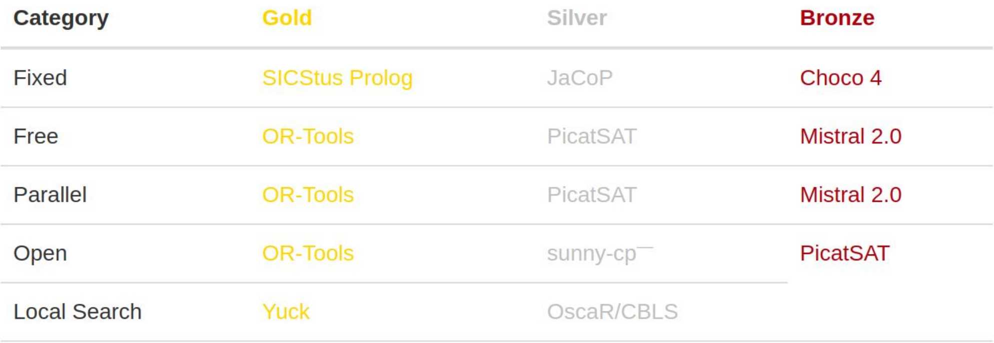
Solving CSPs by encoding to SAT	CSP ➡ SAT
An example SAT encoding	Encoding
Learning encoding choices	Learning
Results and conclusions	Results

The talk is based on our paper ***Learning to Choose SAT Encodings for Pseudo-Boolean and Integer Sum Constraints*** submitted to the Doctoral Programme at CP2021. We thank the reviewers for their helpful comments. We are also presenting aspects of this work at the ModRef2021 workshop.

Why SAT?



Rank	Solver	#solved	Detail	%inst.	%VBS
Total number of instances: 300					
Virtual Best Solver (VBS)		272	177 SAT, 95 UNSAT	91%	100%
1	PicatSAT	245	163 SAT, 82 UNSAT	82%	90%
2	Fun-sCOP *hybrid+CryptoMiniSat*	209	132 SAT, 77 UNSAT	70%	77%
3	Fun-sCOP *hybrid+ManyGlucose*	198	121 SAT, 77 UNSAT	66%	73%
4	Fun-sCOP *order+ManyGlucose*	192	122 SAT, 70 UNSAT	64%	71%
5	Fun-sCOP *order+GlueMiniSat*	190	122 SAT, 68 UNSAT	63%	70%
6	AbsCon	167	114 SAT, 53 UNSAT	56%	61%
7	Concrete	156	106 SAT, 50 UNSAT	52%	57%



Category	Gold	Silver	Bronze
Fixed	SICStus Prolog	JaCoP	Choco 4
Free	OR-Tools	PicatSAT	Mistral 2.0
Parallel	OR-Tools	PicatSAT	Mistral 2.0
Open	OR-Tools	sunny-cp—	PicatSAT
Local Search	Yuck	OscaR/CBLS	

CSP ➡ SAT

Encoding

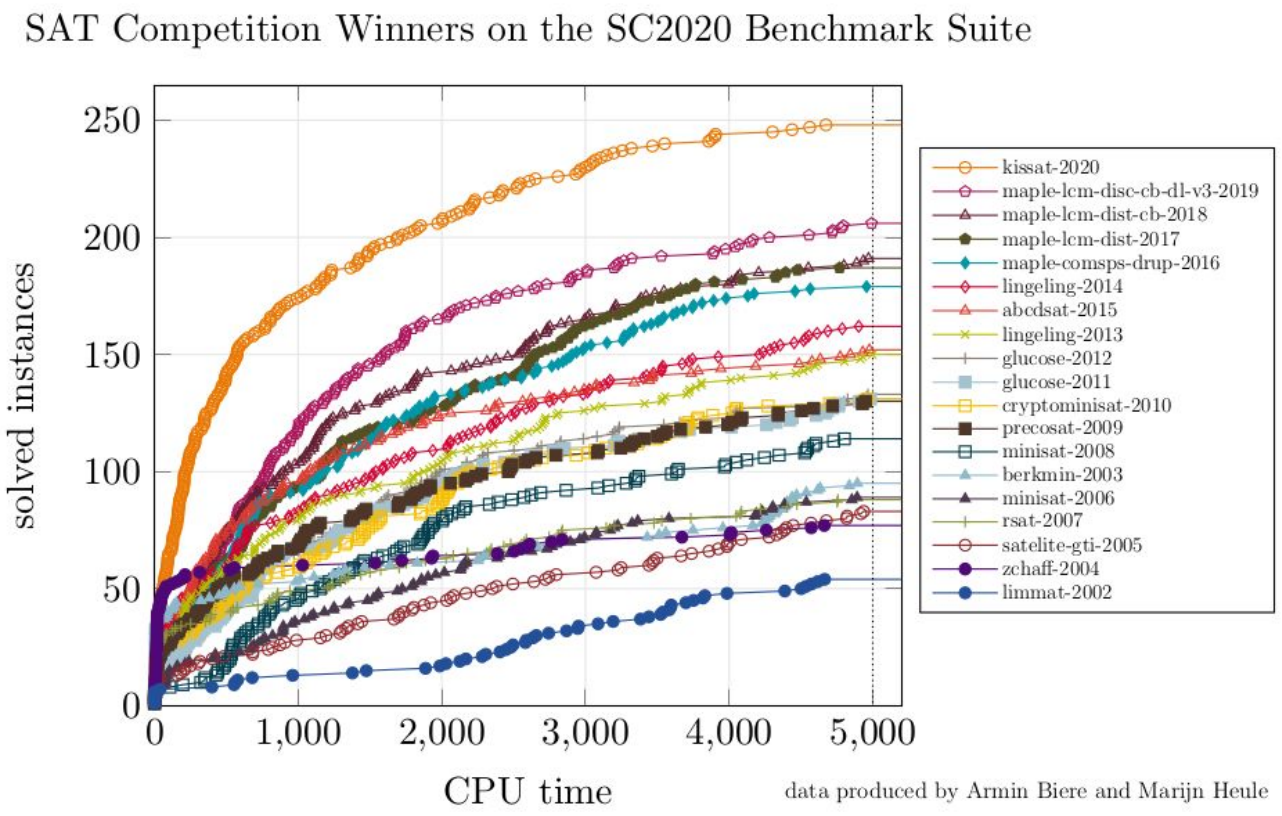
Learning

Results

TOP: Results of the XCSP 2019 Constraint Solver Competition http://www.cril.univ-artois.fr/XCSP19/files/resultsXCSP3-19.pdf, organised by the creators of XCSP3 http://www.xcsp.org/

BOTTOM: Results of the MiniZinc Challenge 2020 https://www.minizinc.org/challenge2020/results2020.html

Why SAT?



SAT Competition Winners on the SC2020 Benchmark Suite

data produced by Armin Biere and Marijn Heule

The SAT Competition 2020 problems run on the winning solvers in previous years, http://fmv.jku.at/kissat/ (thanks to Armin Biere)

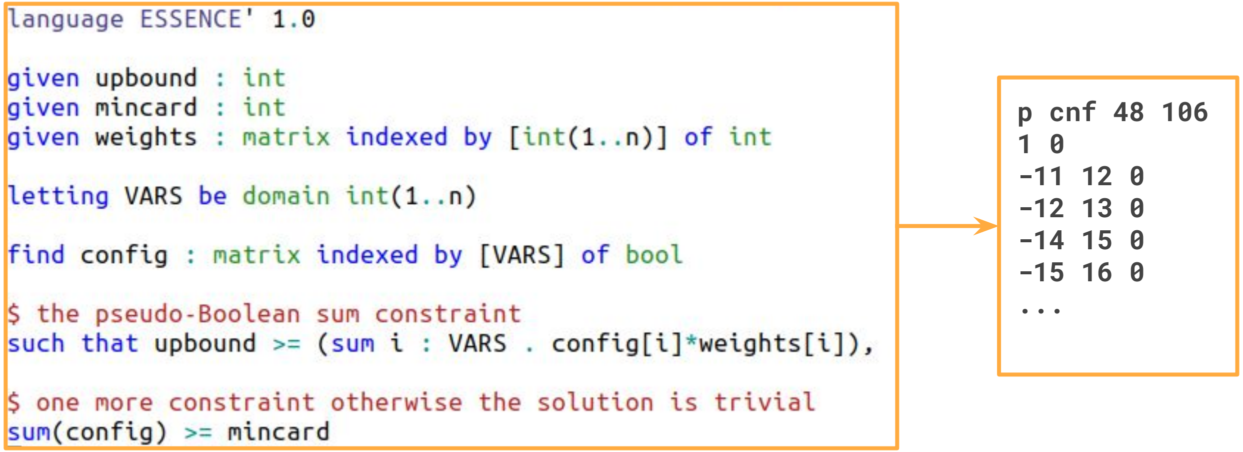
CSP ➡ SAT

Encoding

Learning

Results

Savile Row, Essence Prime, SAT



```
language ESSENCE' 1.0

given upbound : int
given mincard : int
given weights : matrix indexed by [int(1..n)] of int

letting VARS be domain int(1..n)

find config : matrix indexed by [VARS] of bool

$ the pseudo-Boolean sum constraint
such that upbound >= (sum i : VARS . config[i]*weights[i]),

$ one more constraint otherwise the solution is trivial
sum(config) >= mincard
```

```
p cnf 48 106
1 0
-11 12 0
-12 13 0
-14 15 0
-15 16 0
...
```

CSP ➡ SAT

Encoding

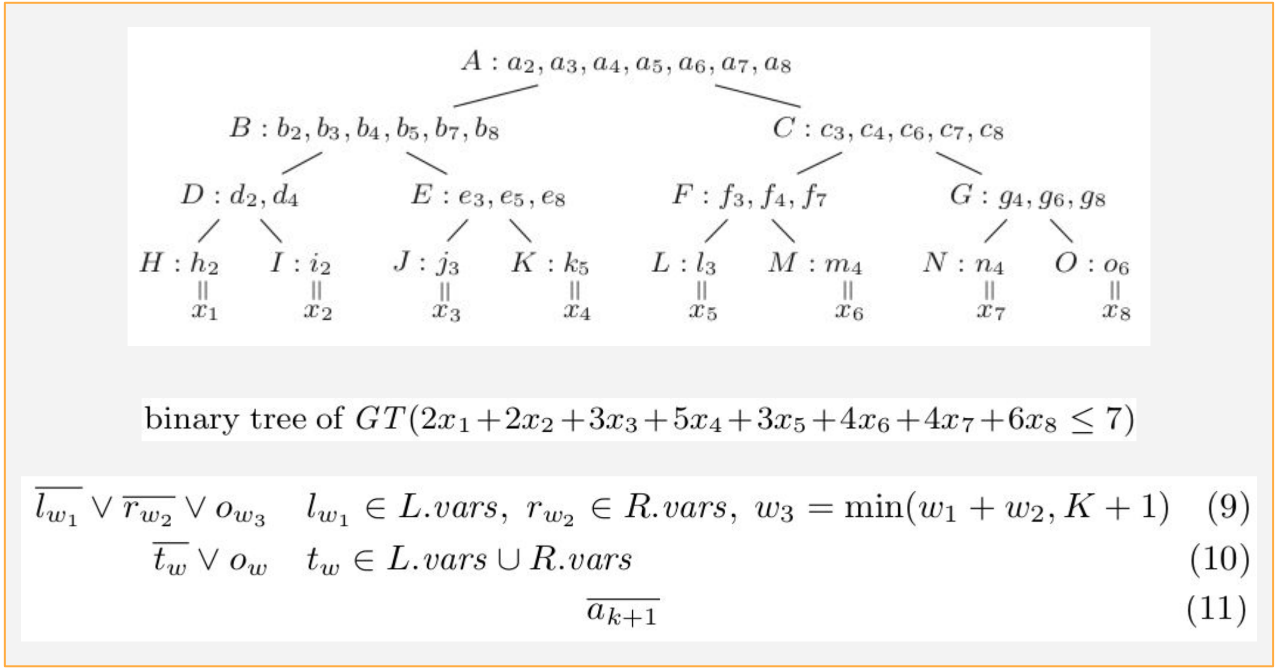
Learning

Results

LEFT: An Essence Prime model for a simple knapsack-like problem with a single pseudo-Boolean sum constraint

RIGHT: The beginning of a Boolean SAT formula in DIMACS format, as produced by Savile Row from the Essence Prime model. The formula essentially begins $(\neg x_{11} \lor x_{12}) \land (\neg x_{12} \lor x_{13}) \land \ldots$

Encoding Example



binary tree of $GT(2x_1+2x_2+3x_3+5x_4+3x_5+4x_6+4x_7+6x_8 \leq 7)$

$$\overline{l_{w_1}} \vee \overline{r_{w_2}} \vee o_{w_3} \qquad l_{w_1} \in L.vars,\ r_{w_2} \in R.vars,\ w_3 = \min(w_1 + w_2, K+1) \tag{9}$$

$$\overline{t_w} \vee o_w \qquad t_w \in L.vars \cup R.vars \tag{10}$$

$$\overline{a_{k+1}} \tag{11}$$

An extract from a SAT encoding description for a pseudo-Boolean sum constraint.

Diagrams and clauses for the “Generalized Totalizer” from *Bofill, Coll, Suy, Villaret: SAT Encodings of Pseudo-Boolean Constraints with At-Most-One Relations*, in CPAIOR 2019 https://doi.org/10.1007/978-3-030-19212-9_8

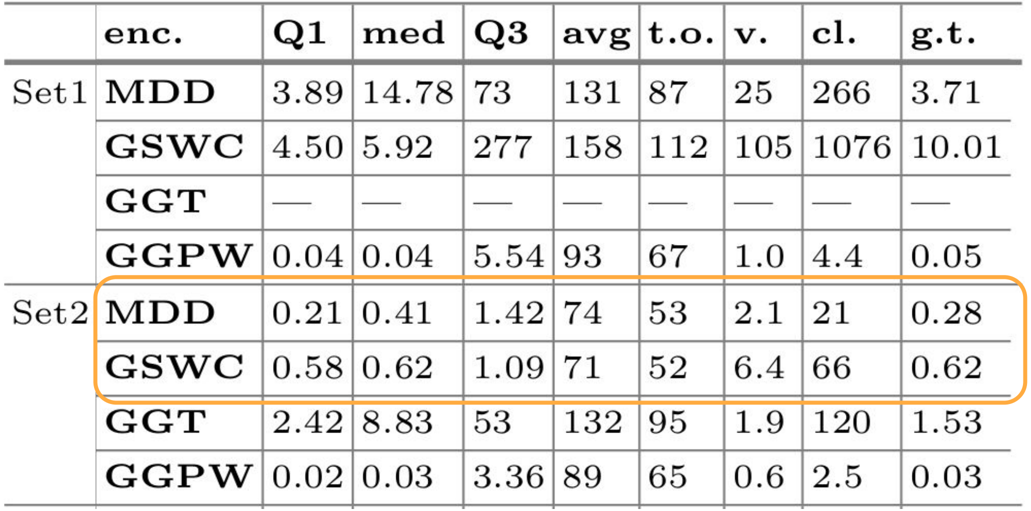
CSP ➡ SAT

Encoding

Learning

Results

Encoding a Constraint



	enc.	Q1	med	Q3	avg	t.o.	v.	cl.	g.t.
Set1	**MDD**	3.89	14.78	73	131	87	25	266	3.71
	GSWC	4.50	5.92	277	158	112	105	1076	10.01
	GGT	—	—	—	—	—	—	—	—
	GGPW	0.04	0.04	5.54	93	67	1.0	4.4	0.05
Set2	**MDD**	0.21	0.41	1.42	74	53	2.1	21	0.28
	GSWC	0.58	0.62	1.09	71	52	6.4	66	0.62
	GGT	2.42	8.83	53	132	95	1.9	120	1.53
	GGPW	0.02	0.03	3.36	89	65	0.6	2.5	0.03

CSP ➡ SAT

Encoding

Learning

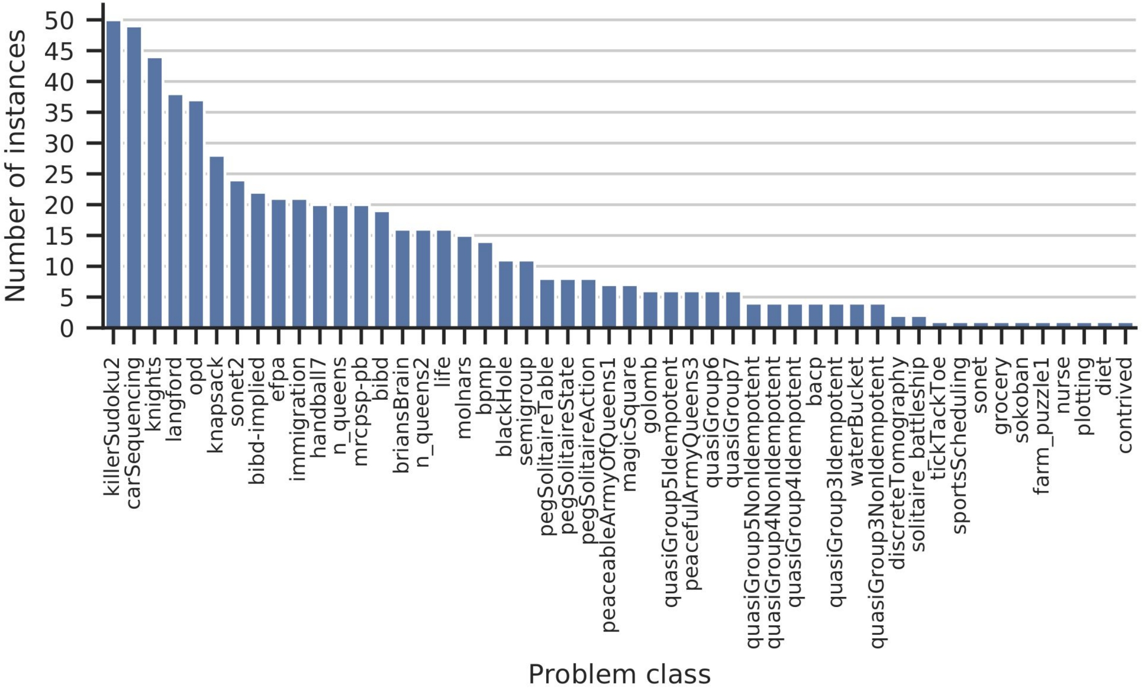
Results

Performance summary from *Bofill, Coll, Suy, Villaret: SAT Encodings of Pseudo-Boolean Constraints with At-Most-One Relations*, in CPAIOR 2019
https://doi.org/10.1007/978-3-030-19212-9_8

Experimental setup

- Savile Row has MDD, GSWC, GGPW, GGT + Tree encodings
- 5 choices for sums x 5 choices for PBs = 25 configurations
- each instance run with each configuration 5 times and the median time taken (to average out SAT solver randomness)
- timeouts set to 1 hour each for Savile Row and the SAT solver (Kissat)

Problem Corpus



CSP ➡ SAT

Encoding

Learning

Results

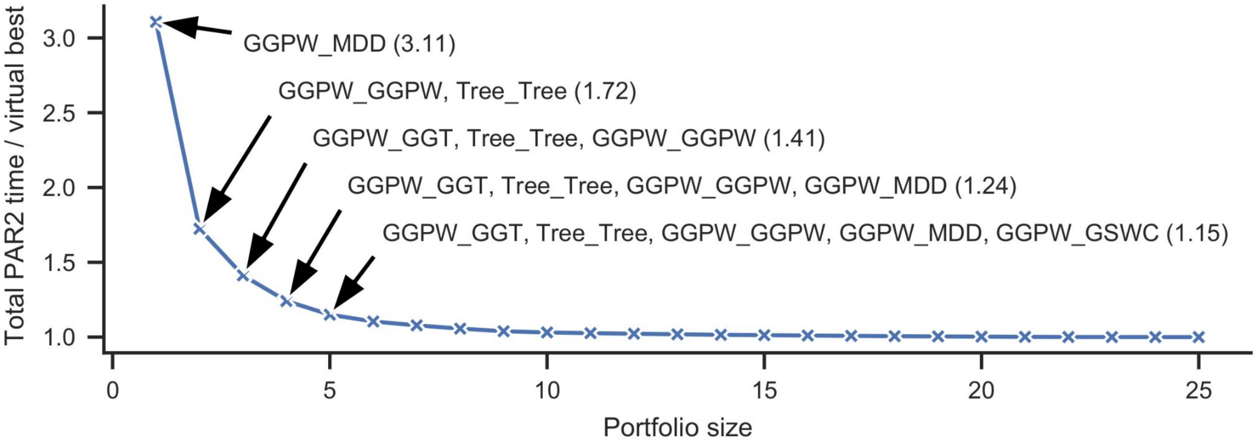
Essence Prime Models mainly from *Davidson, Akgün, Espasa, Nightingale: Effective Encodings of Constraint Programming Models to SMT.*
In CP 2020 https://doi.org/10.1007/978-3-030-58475-7_9

Pairwise Training

- random forests trained to make binary choice for each pair of configurations
- pairwise predictions give a ranking
- top configuration becomes our prediction

Pairwise voting random forests inspired by *Lindauer, Hoos, Hutter, Schaub: AutoFolio: An Automatically Configured Algorithm Selector*. In JAIR 2015
https://doi.org/10.1613/jair.4726

A complementary portfolio



The virtual best PAR2 run-time on our corpus for all portfolio sizes as a multiple of the overall virtual best; the resulting portfolios (of *sum_pb* configurations) are shown for sizes 1 to 5

CSP ➡ SAT

Encoding

Learning

Results

Pairwise voting random forests inspired by *Lindauer, Hoos, Hutter, Schaub: AutoFolio: An Automatically Configured Algorithm Selector*. In JAIR 2015
https://doi.org/10.1613/jair.4726

Instance Features

- **f2f**: from `fzn2feat` tool [1]: 95 generic CSP instance features relating to constraints, variables, and their domains. Extracted by outputting FlatZinc from Savile Row, then running `fzn2feat`
- **f2fsr**: an attempt to extract the same features from Savile Row’s internal model just before encoding to SAT
- **sumpb**: new pb-related features
- **combi**: *f2fsr* and *sumpb* combined

CSP ➡ SAT

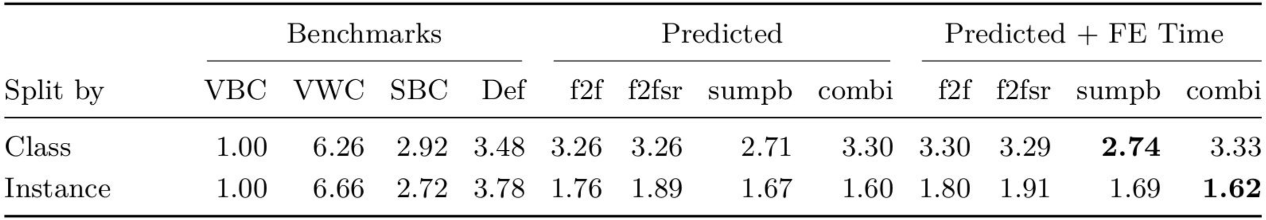
Encoding

Learning

Results

[1] *Amadini, Gabbrielli, Mauro: An enhanced features extractor for a portfolio of constraint solvers*. In SAC ’14 https://doi.org/10.1145/2554850.2555114

Evaluating performance



	Benchmarks				Predicted				Predicted + FE Time			
Split by	VBC	VWC	SBC	Def	f2f	f2fsr	sumpb	combi	f2f	f2fsr	sumpb	combi
Class	1.00	6.26	2.92	3.48	3.26	3.26	2.71	3.30	3.30	3.29	**2.74**	3.33
Instance	1.00	6.66	2.72	3.78	1.76	1.89	1.67	1.60	1.80	1.91	1.69	**1.62**

CSP ➡ SAT

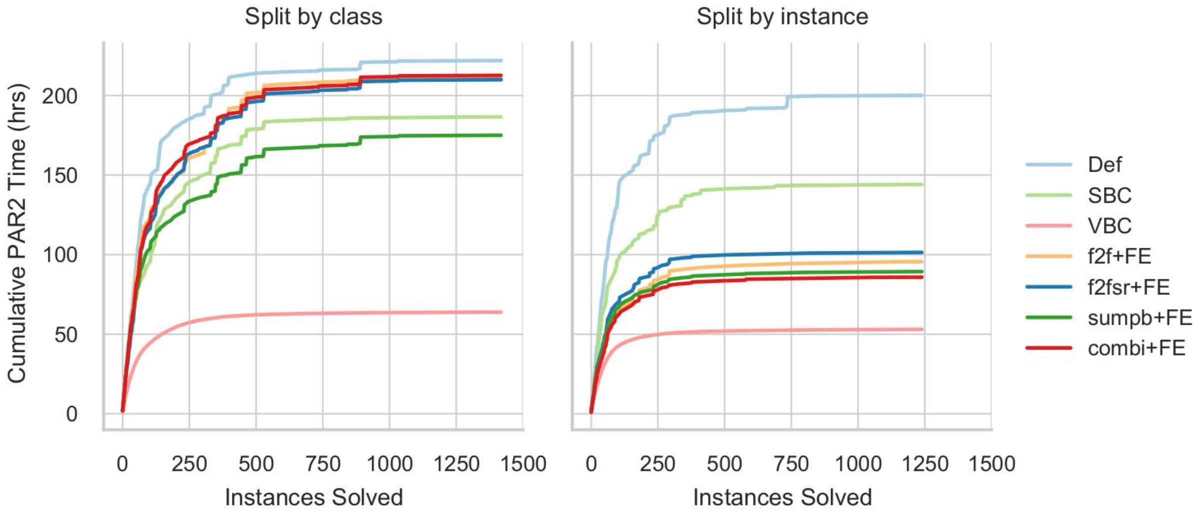
Encoding

Learning

Results

Total PAR2 times over the 10 test sets **as a multiple of the virtual best** configuration time. We show the times for the virtual best (VBC), virtual worst (VWC), single best (SBC), and default (Def) configurations, followed by the timings using predictions made on our four feature sets, without and with feature extraction (FE) time. The best time (including FE) for each row is shown in bold.

Results



Cumulative PAR2 time over the 10 test sets, with instances sorted (by VBC solving time) to place the most difficult instances first on the x-axis.

Findings and Future

For our corpus:

- ML can outperform the single best encoding
- good encoding for PBs more important than for sums
- PB features better at predicting for new problem classes

In the future:

- extend to a broader benchmark of problems
- apply to other encoding choices
- consider at-most-one groups
- analyse and discuss feature importance