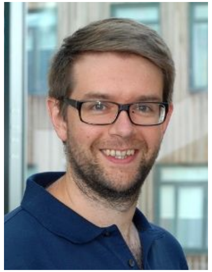
# Felix Ulrich-Oltean

Supervised by
**Peter Nightingale, James Cussens, James Walker**



PhD student @ University of York

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


Working on

# Learning SAT Encodings for Individual Constraints


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

# What’s in store

## Motivation: Why SAT?

## Encoding to SAT, an example

## Preliminary experiments

## Learning the best encodings

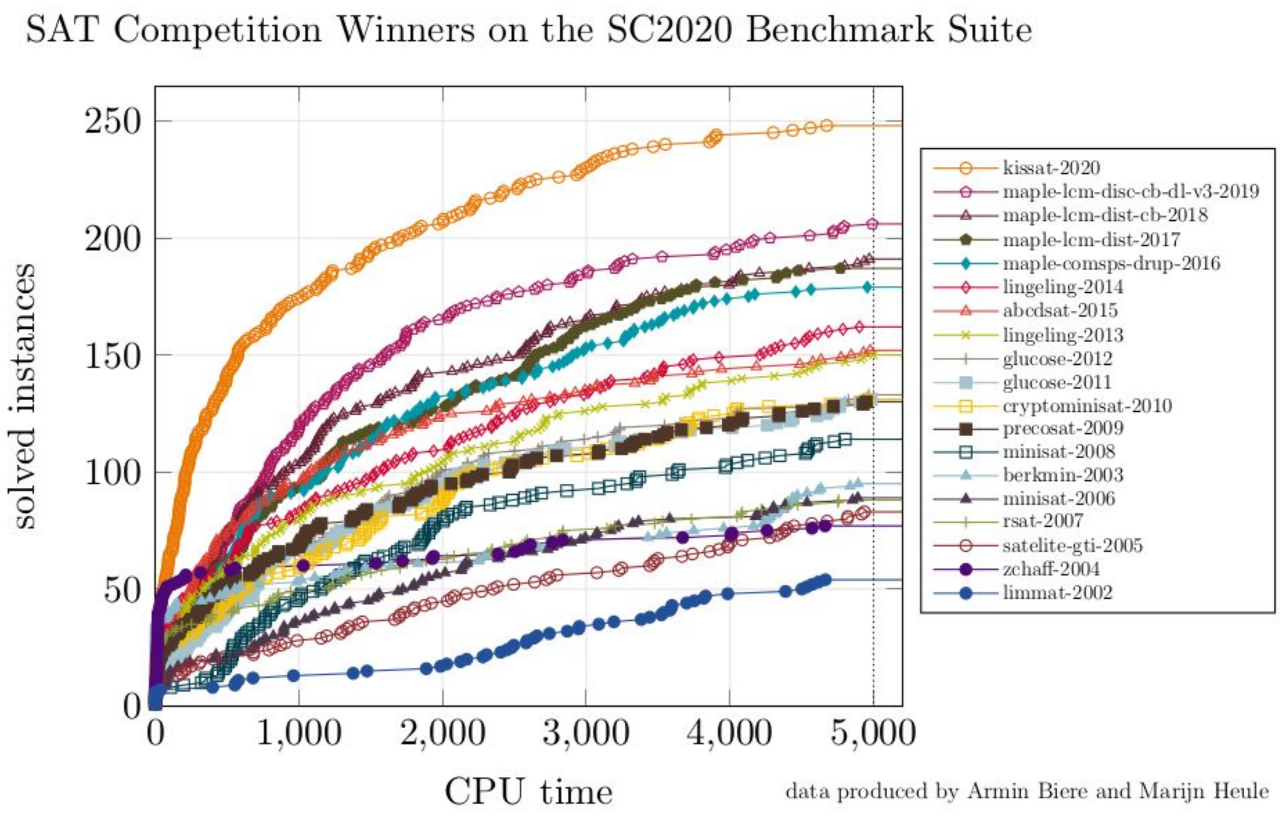
The talk is based on my Research Summary Paper submitted to the Doctoral Programme at CP2020. My thanks go to the peer reviewers for their helpful, actionable comments.

# 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% |
| 8 | choco-solver *parallel* | 153 | 113 SAT, 40 UNSAT | 51% | 56% |
| 9 | choco-solver | 149 | 101 SAT, 48 UNSAT | 50% | 55% |
| 10 | BTD | 135 | 84 SAT, 51 UNSAT | 45% | 50% |
| 11 | cosoco | 126 | 82 SAT, 44 UNSAT | 42% | 46% |
| 12 | cosoco *parallel* | 121 | 86 SAT, 35 UNSAT | 40% | 44% |

From the Presentations of 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/

# Why SAT?




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

# Encoding to SAT: an example

```
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
...
```

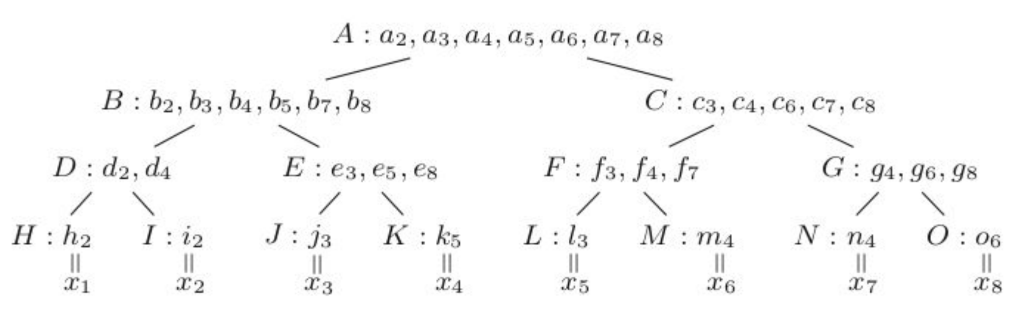


LEFT: An Essence Prime model for a simple knapsack-like problem with a single pseudo-Boolean sum constraint

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 $\vee$ x_12) $\wedge$ ($\neg$x_12 $\vee$ x_13) $\wedge$** ...

# Encoding a Constraint






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

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

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

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

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

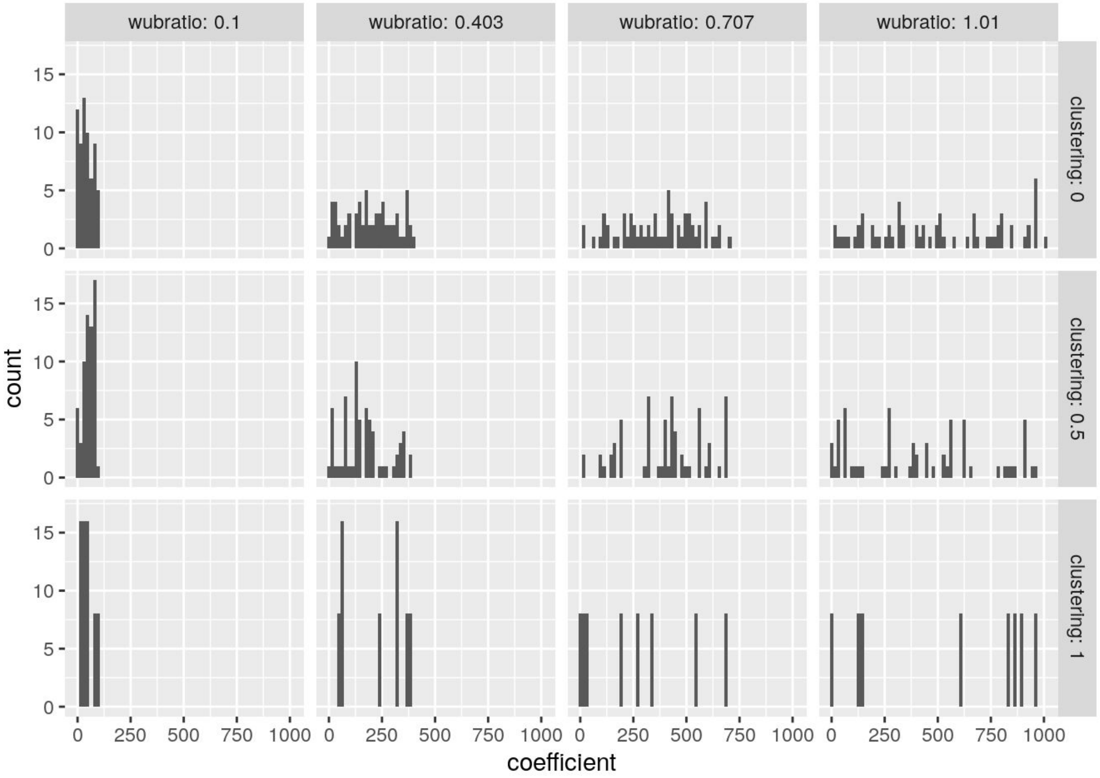
Diagrams and clauses for the “Generalized Totalizer” from *Bofill, Coll, Suy, Villaret: SAT Encodings of Pseudo-Boolean Constraints with At-Most-One Relations*, in CPAIOR 2019 https://doi.org/10.1007/978-3-030-19212-9_8

# Encoding a Constraint

| | enc. | Q1 | med | Q3 | avg | t.o. | v. | cl. | g.t. |
|---|---|---|---|---|---|---|---|---|---|
| Set1 | **BDD** | 14.00 | 17.59 | t.o. | 219 | 158 | 857 | 1714 | 35.6 |
| | **SWC** | 10.51 | 14.12 | t.o. | 199 | 144 | 1100 | 2177 | 17.2 |
| | **GT** | — | — | — | — | — | — | — | — |
| | **GPW** | 0.93 | 0.97 | 23 | 114 | 85 | 5.9 | 77 | 0.8 |
| Set2 | **BDD** | 4.29 | 5.65 | 133 | 141 | 96 | 57 | 115 | 2.0 |
| | **SWC** | 4.10 | 5.41 | 138 | 140 | 95 | 68 | 135 | 1.3 |
| | **GT** | 5.33 | 6.94 | 182 | 154 | 110 | 10 | 1640 | 18.0 |
| | **GPW** | 0.46 | 0.48 | 11 | 108 | 77 | 3.5 | 42 | 0.4 |
| Set3 | **BDD** | 215 | t.o. | t.o. | 423 | 218 | 4.8 | 9.6 | 0.7 |
| | **SWC** | 247 | t.o. | t.o. | 429 | 227 | 6.0 | 12 | 0.6 |
| | **GT** | 240 | t.o. | t.o. | 427 | 223 | 1.3 | 31 | 1.6 |
| | **GPW** | 172 | t.o | t.o | 415 | 229 | 0.8 | 5.1 | 0.3 |

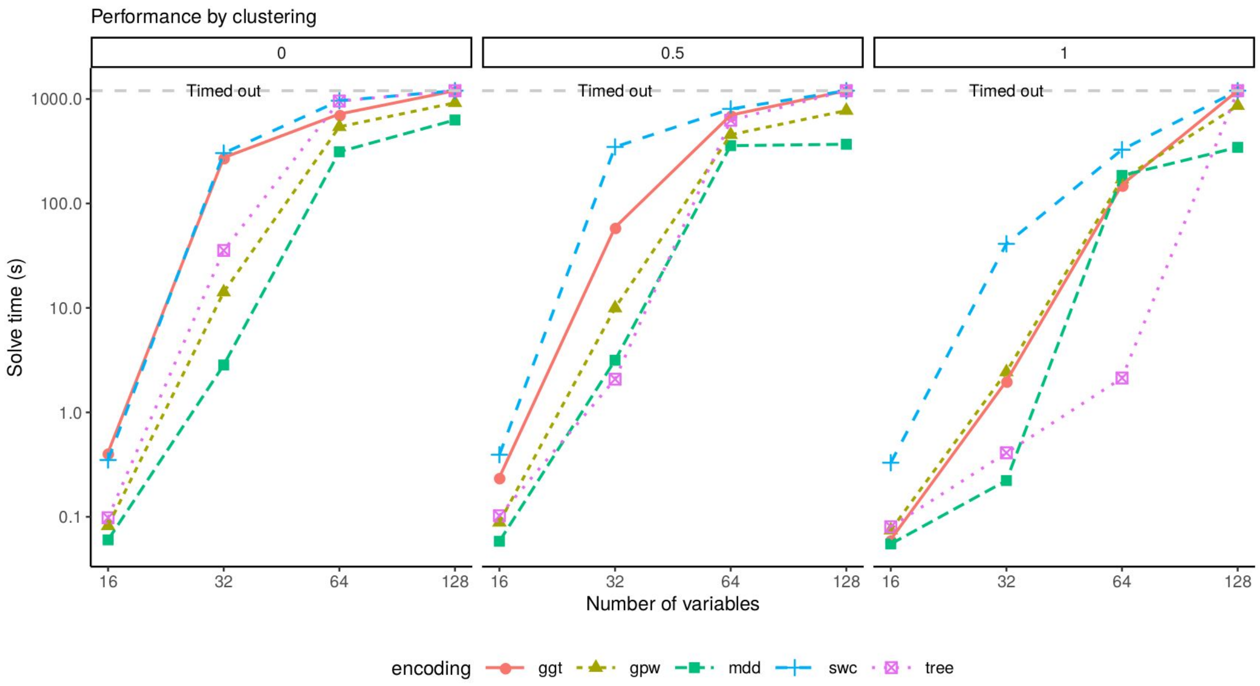
Performance summary from *Bofill, Coll, Suy, Villaret: SAT Encodings of Pseudo-Boolean Constraints with At-Most-One Relations*, in CPAIOR 2019
https://doi.org/10.1007/978-3-030-19212-9_8

# Preliminary Experiments




Distributions of weights/coefficients for generated instances - instances with 64 variables shown here.

# Preliminary Experiments






As we vary the amount of *clustering*, the relative performance of the encodings changes (clustering = 0 means weights are completely randomly spread, 1 means all the weights are in clusters)

# Learning: Features

```
$ mzn2feat -P
========== CONSTRAINT FEATURES (27) ==========
c_avg_domdeg_cons...Avg of the ratio constraints domain/degree
c_sum_ari_cons......Sum of constraints arity
========== DOMAIN FEATURES (18) ==========
d_array_cons........No. of array constraints
d_int_cons..........No. of integer constraints
========== GLOBAL CONSTRAINT FEATURES (4) ==========
gc_diff_globs.......No. of different global constraints
========== OBJECTIVE FEATURES (8) ==========
o_deg...............Degree of the objective variable
========== VARIABLE FEATURES (27) ==========
v_def_vars..........Number of defined variables
v_min_domdeg_vars...Minimum of the ratio variables domain/degree
```

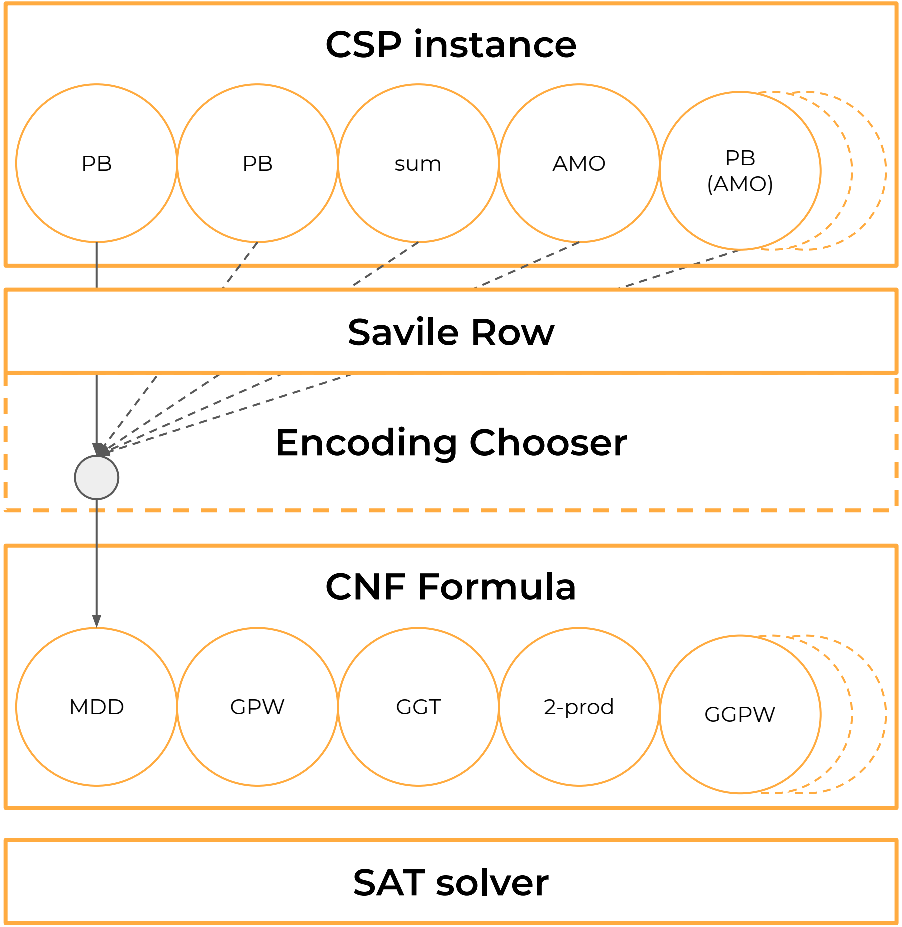


A small sample of the features extracted by the mzn2feat/fzn2feat tool which operates on minizinc or flatzinc files. With thanks to Roberto Amadini for the project at [https://github.com/CP-Unibo/mzn2feat](https://github.com/CP-Unibo/mzn2feat)

# Learning: First Steps

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| ggt | 0.50 | 0.67 | 0.57 | 3 |
| gpw | 1.00 | 0.50 | 0.67 | 2 |
| mdd | 0.60 | 0.60 | 0.60 | 5 |
| swc | 0.33 | 0.50 | 0.40 | 2 |
| tree | 0.67 | 0.50 | 0.57 | 4 |
| accuracy | | | 0.56 | 16 |
| macro avg | 0.62 | 0.55 | 0.56 | 16 |
| weighted avg | 0.61 | 0.56 | 0.57 | 16 |

# The plan...




Why SAT?

Encoding

Experiments

**Learning**