Karlsruher Institut für Technologie

Dinosat: A SAT Solver with Native DNF Support

Thomas Bartel (CAS), Tomáš Balyo (CAS), Markus Iser
Institute of Theoretical Informatics, Algorithm Engineering

Context

- CAS Merlin: SAT-based Product Configuration
- Not CNF: also AMO and DNF subformulas
- Slighly different problem: online algorithm produces optimal solutions

This Work

- Introduce format RichCNF (Clauses + DNF and AMO constraints)
- Write DPLL solver which also propagates DNF and AMO
- Write CDCL solver which learns from conflicts with DNF and AMO
- Evaluate Phase Transitions: SAT/UNSAT, Performance

Motivation

Karlsruher Institut für Technologie

- Modern SAT solvers solve formulas in their „conjunctive normal form“ (CNF) [1]

$$\overbrace{\underbrace{(a \vee b)}_{\text{Disjunction}} \wedge (c \vee d)}^{\text{Conjunction}}$$

Other constraint types

- The „disjunctive normal form“ (DNF) is a disjunction of conjunctions

$$\overbrace{\underbrace{(a \wedge b)}_{\text{Conjunction}} \vee (c \wedge d)}^{\text{Disjunction}}$$

- The „At most one constraint“ (AMO) only allows at most one literal to be true

$$AMO(a, b, c, d)$$

Two major SAT solving algorithms [1]

- Davis-Putnam-Logemann-Loveland (DPLL) algorithm
  - Exhaustive depth-first search of the space of variable assignments
  - Resolves conflict, by trying another value
- Conflict-Driven-Clause-Learning (CDCL) algorithm
  - Evolved from the DPLL-algorithm
  - Resolves conflict by learning new clauses

Resolution [2]

- The current best SAT solvers use a concept called „resolution“

$$\frac{\overbrace{a \lor b \qquad c \lor \neg b}^{\text{Two conflicting clauses}}}{a \lor c}$$

This clause is implied by the two clauses above

Notation in the next slides

Karlsruher Institut für Technologie

- The next slides use the following notation for DNF constraints:

$$(a \wedge b) \vee (c \wedge d) \Leftrightarrow DNF((a,b),(c,d))$$

First AMO-AMO conflict

Complementary literals!

$$AMO(a, b, d, c) \qquad AMO(e, g, f, \neg c)$$

$$d = 1 \qquad f = 1$$

$$\Downarrow \qquad \Downarrow$$

$$a = 0, b = 0, c = 0 \qquad e = 0, g = 0, \neg c = 0$$

Conflict!

First AMO-AMO conflict

Karlsruher Institut für Technologie

$$\frac{AMO(a, b, d, c) \qquad AMO(e, g, f, \neg c)}{(\neg a \wedge \neg b \wedge \neg d) \vee (\neg e \wedge \neg g \wedge \neg f)}$$

Not a CNF at the moment!

First AMO-AMO conflict

$$(\neg a \wedge \neg b \wedge \neg d) \vee (\neg e \wedge \neg g \wedge \neg f)$$

$$\Downarrow$$

$$(\neg a \vee \neg e) \wedge (\neg a \vee \neg g) \wedge (\neg a \vee \neg f) \wedge$$
$$(\neg b \vee \neg e) \wedge (\neg b \vee \neg g) \wedge (\neg b \vee \neg f) \wedge$$
$$(\neg d \vee \neg e) \wedge (\neg d \vee \neg g) \wedge (\neg d \vee \neg f)$$

First AMO-AMO conflict

Karlsruher Institut für Technologie

> **Lemma 4.2.1.** *Consider two AMO constraints AMO_1 and AMO_2. Both constraints are complementary on the variable x_a. AMO_1 contains exactly one literal x_a with AMO_2 containing the complementary literal $\neg x_a$. Let $\{y_1, ..., y_k\}$ be the set of literals of AMO_1 without x_a and $\{z_1, ..., z_m\}$ be the set of literals of AMO_2 without $\neg x_a$. If a conflict occurs between these constraints, then it can be resolved by learning the following clauses:*
>
> $$\{\neg y_i \vee \neg z_j | y_i \in \{y_1, ..., y_k\}, z_j \in \{z_1, ..., z_m\}, \{y_i, z_j\} \notin \{x_a, \neg x_a\}\}$$

First AMO-AMO conflict

Karlsruher Institut für Technologie

$$(\neg a \land \neg b \land \neg d) \lor (\neg e \land \neg g \land \neg f)$$

$$\Downarrow$$

$$(\neg a \lor \neg e) \land (\neg a \lor \neg g) \land (\neg a \lor \neg f) \land$$
$$(\neg b \lor \neg e) \land (\neg b \lor \neg g) \land (\neg b \lor \neg f) \land$$
$$(\neg d \lor \neg e) \land (\neg d \lor \neg g) \land (\neg d \lor \neg f)$$

Too many clauses?

First AMO-AMO conflict

$AMO(a, b, d, c)$

$d = 1$

$\Downarrow$

$a = 0, b = 0, c = 0$

$AMO(e, g, f, \neg c)$

$f = 1$

$\Downarrow$

$e = 0, g = 0, \neg c = 0$

First AMO-AMO conflict

$$\frac{AMO(a,b,d,c) \qquad AMO(e,g,f,\neg c)}{(\neg d \vee \neg f)}$$

$$d=1, f=1$$

First AMO-AMO conflict

Karlsruher Institut für Technologie

Lemma 4.2.2. *Consider two AMO constraints AMO_1 and AMO_2. Both constraints are complementary on the variable x_a. AMO_1 contains exactly one literal x_a with AMO_2 containing the complementary literal $\neg x_a$. Let y_i be a literal that is true in AMO_1 and not x_a. Let z_j be a literal that is true in AMO_2 and not $\neg x_a$. Then this specific conflict can be avoided by learning the following clause:*

$$\neg y_i \vee \neg z_j$$

DNF-DNF conflict

Complementary literals!

$$DNF((a,b),(c,d))$$

$$DNF((e,\neg b),(f,g))$$

$$c=0$$

$$f=0$$

$$a=1, b=1$$

$$e=1, \neg b=1$$

Conflict!

DNF-DNF conflict

$$\frac{DNF\left(\left(a,b\right),\left(c,d\right)\right) \quad DNF\left(\left(e,\neg b\right),\left(f,g\right)\right)}{DNF\left(\left(c,d\right),\left(f,g\right)\right)}$$

DNF-DNF conflict

Lemma 4.3.1. *Consider two DNF constraints DNF_1 and DNF_2, that are complementary on the variable x. DNF_1 contains the literal x and DNF_2 contains the literal $\neg x$. A conflict, that is caused by the variable x, can then be resolved by learning the following constraints:*

$$DNF_1/x \vee DNF_2/\neg x$$

DNF-DNF conflict

$$DNF((c,d),(f,g))$$

$$\Downarrow$$

$$(c \vee f) \wedge (c \vee g) \wedge (d \vee f) \wedge (d \vee g)$$

Every learnt DNF is effectively a large amount of clauses

DNF-DNF conflict

$$DNF\left(\left(a, b\right), \left(c, d\right)\right)$$

$$c = 0$$

$$\Downarrow$$

$$a = 1, b = 1$$

$$DNF\left(\left(e, \neg b\right), \left(f, g\right)\right)$$

$$f = 0$$

$$\Downarrow$$

$$e = 1, \neg b = 1$$

DNF-DNF conflict

$$\frac{DNF((a,b),(c,d)) \quad DNF((e,\neg b),(f,g))}{}$$

Signifcantly fewer constraints

$$c \vee f$$

$$c=0, f=0$$

DNF-DNF conflict

Lemma 4.3.2. *Consider two DNF constraints DNF_1 and DNF_2 and the variable x, that the two constraints are complementary on. DNF_1 contains the literal x and DNF_2 contains the literal $\neg x$. Let $\{y_1, ..., y_n\}$ be the literals of DNF_1, that turned false and therefore forced a unit propagation of the literal x. Let $\{z_1, ..., z_m\}$ be the literals of DNF_2, that turned false and therefore caused the propagation of the literal $\neg x$. Then this specific conflict can be resolved by learning the following clause:*

$$y_1 \vee ... \vee y_n \vee z_1 \vee ... \vee z_m$$

Phase transition [3]

Karlsruher Institut für Technologie

- Phase transition point is the clause to variable ratio, where about 50 percent of the randomly generated formulas are unsatisfiable
- At the phase transition point of the clauses, there are difficult instances
- Does there exist a phase transition point for the other constraint types?

Phase transition

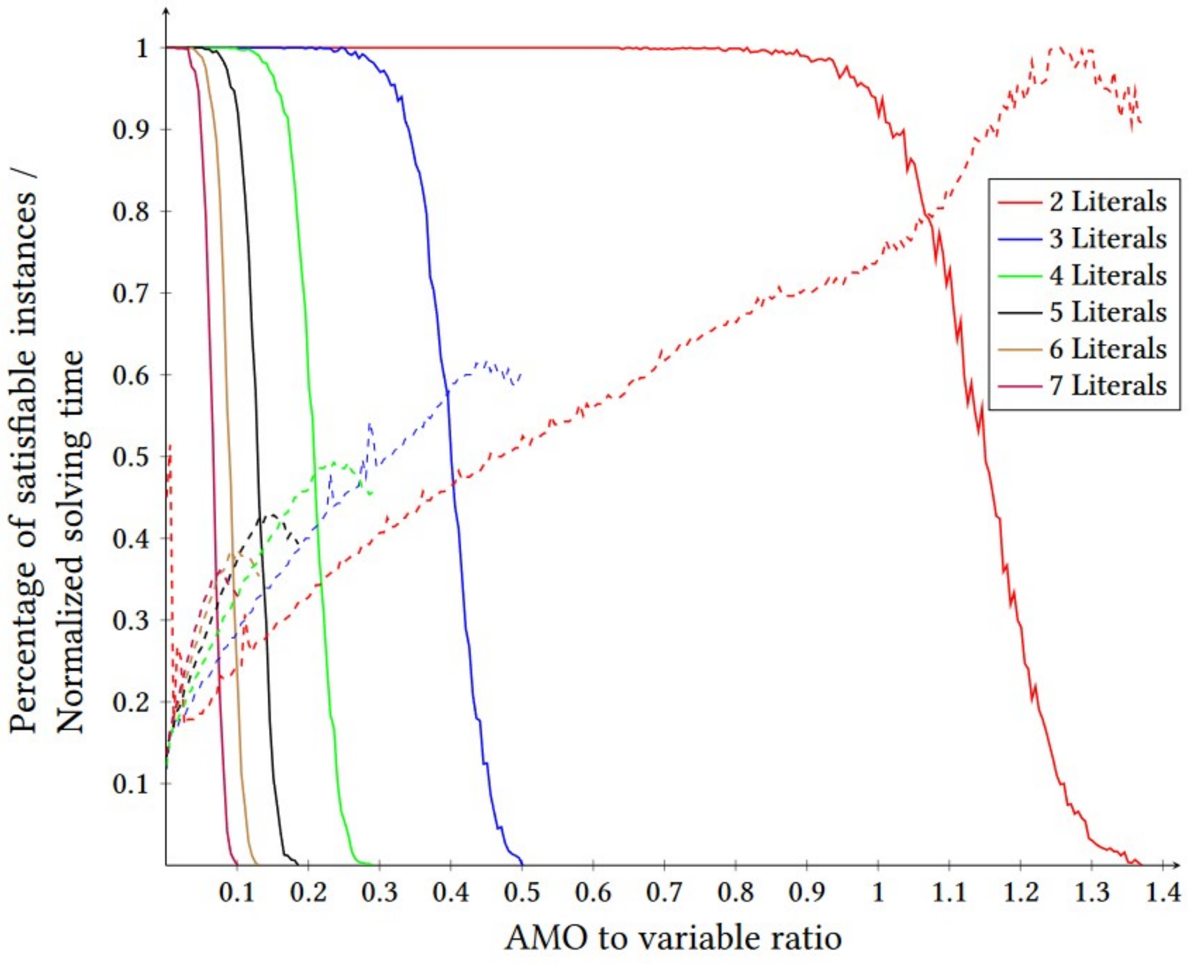

Figure 6.1: Phase transition of AMO constraints with 1000 variables and an increment of five AMO constraints per iteration

Phase transition

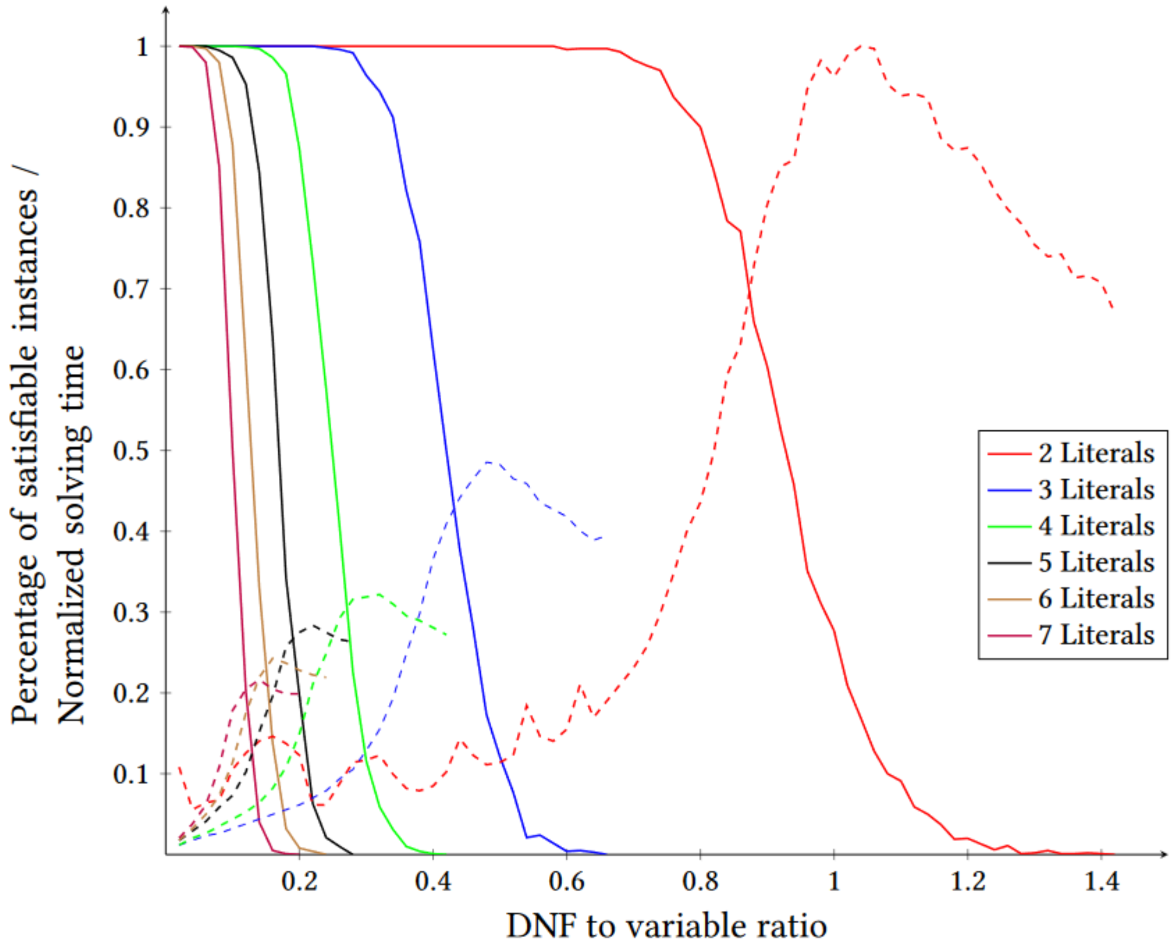

Figure 6.2: Phase transition and solving time of DNF constraints with a constant term count of 3, 50 variables, and a DNF increment of 1 per iteration

Phase transition

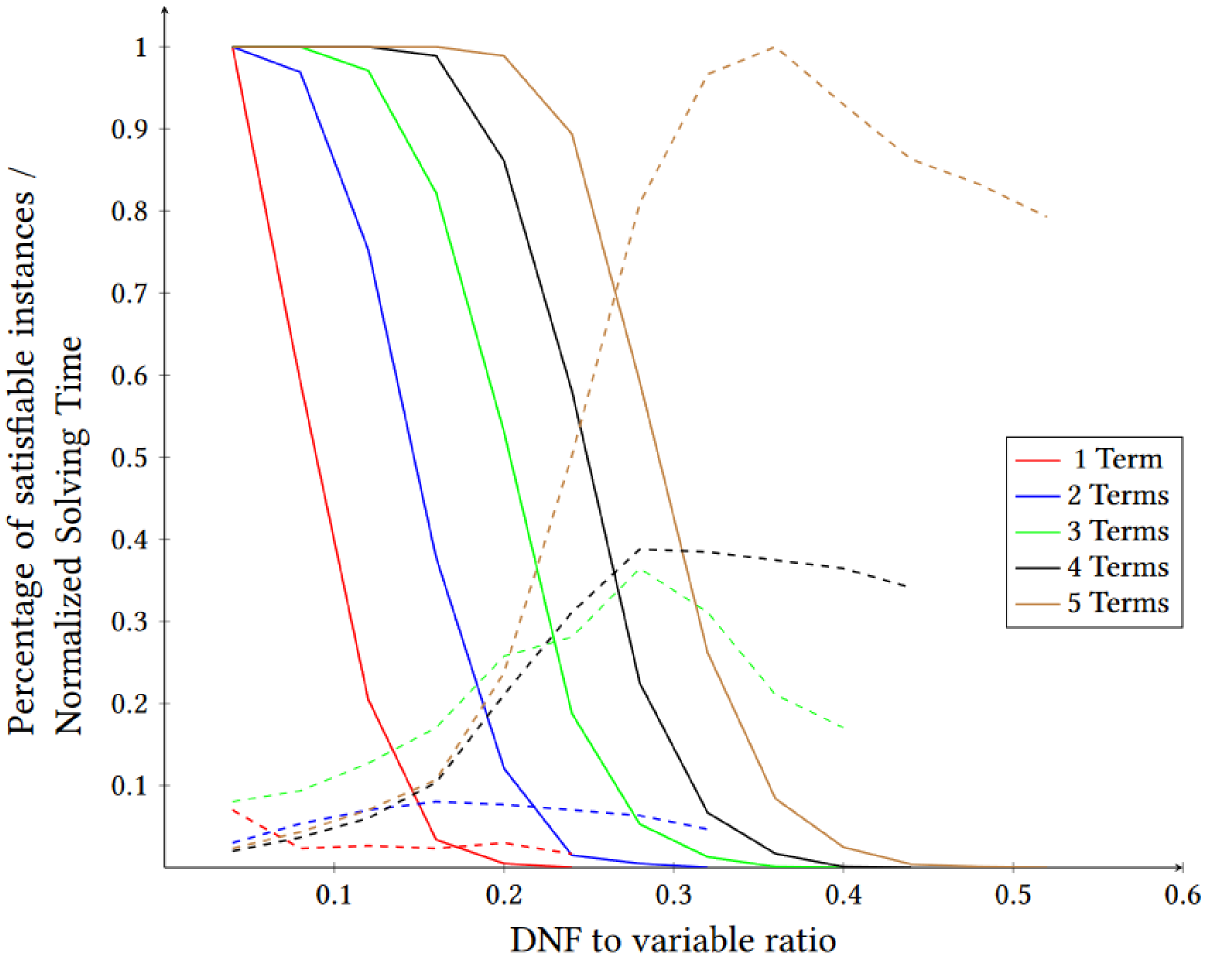

Figure 6.3: Phase transition and solving time of DNF constraints with a constant term length of 5, 25 variables and a DNF increment of 1 per iteration

Evaluation

Karlsruher Institut für Technologie

- Industrial benchmarks provided by CAS
- Randomized benchmarks
- Performance comparison with Sat4j
- Encoding of benchmarks, so that Sat4j can solve them

Industrial benchmarks

Table 6.1: Composition of the industrial benchmark sets with the number of Formulas (F), the average number of variables (V), the average number of DNF constraints (DNF), the average number of terms per DNF (TPD), average number of literals per term (LPT), the average number of clauses (C), the average number of literals per clause (LPC), the average number of AMO constraints (AMO) and the average number of literals per AMO constraint (LPA)

Name	F	V	DNF	TPD	LPT	C	LPC	AMO	LPA
Ind_{large}	20	27815.40	2645.55	7.50	32.38	114661.60	11.63	917.30	16.25
Ind_{medium}	21	13795.14	546.52	3.98	9.67	18504.38	5.87	393.00	12.40
Ind_{small}	26	6949.23	155.96	2.72	5.42	8626.42	4.02	134.19	16.00
$(CNF)Ind_{large}$	20	49742.00	0.00	0.00	0.00	797924.50	4.18	0.00	0.00
$(CNF)Ind_{medium}$	21	19329.38	0.00	0.00	0.00	53697.86	3.71	0.00	0.00
$(CNF)Ind_{small}$	26	9057.96	0.00	0.00	0.00	16935.81	3.15	0.00	0.00

Industrial benchmarks

Table 6.2: Ind_{large} and $(CNF)Ind_{large}$ performance evaluation with the solving time (ST), number of solved instances (SI), number of timeouts (TO), average number of branching decisions (D(A)), average number of unit propagations (P(A)) and the average number of conflicts (C(A))

Solver	ST	SI	TO	D(A)	P(A)	C(A)
$DPLL_{NR\_F}$	12.144	20	0	26318.40	9797.55	510.10
$DPLL_{R\_F}$	10.668	20	0	35914.35	15574.10	656.30
$CDCL_{R\_F}$	42.942	20	0	671542.00	945384.80	9085.85
$CDCL_{R\_F\_CNF}$	99.89	20	0	1030202.95	4973969.20	2583.40
$Sat4j$	18.867	20	0	82588.10	726588.50	48.25

Industrial benchmarks

Karlsruher Institut für Technologie

Table 6.3: Ind_{medium} and $(CNF)Ind_{medium}$ performance evaluation with the solving time (ST), number of solved instances (SI), number of timeouts (TO), average number of branching decisions (D(A)), average number of unit propagations (P(A)) and the average number of conflicts (C(A))

Solver	ST	SI	TO	D(A)	P(A)	C(A)
$DPLL_{NR\_F}$	2.558	21	0	11403.43	2391.71	0.00
$DPLL_{R\_F}$	1.849	21	0	11403.43	2391.71	0.00
$CDCL_{R\_F}$	1.576	21	0	11403.43	2391.71	0.00
$CDCL_{R\_F\_CNF}$	2.086	21	0	10888.57	10895.29	1.76
$Sat4j$	1.579	21	0	6822.62	16236.81	0.00

Industrial benchmarks

Table 6.4: Ind_{small} and $(CNF)Ind_{small}$ performance evaluation with the solving time (ST), number of solved instances (SI), number of timeouts (TO), average number of branching decisions (D(A)), average number of unit propagations (P(A)) and the average number of conflicts (C(A))

Solver	ST	SI	TO	D(A)	P(A)	C(A)
$DPLL_{NR\_F}$	1.799	26	0	5712.42	1245.27	2.15
$DPLL_{R\_F}$	0.876	26	0	5712.42	1245.27	2.15
$CDCL_{R\_F}$	1.011	26	0	6012.42	1895.88	4.08
$CDCL_{R\_F\_CNF}$	0.958	26	0	5485.58	6132.15	3.15
$Sat4j$	0.914	26	0	3900.38	8410.46	0.85

Randomized benchmarks

Table 3: DNF constraints with 15 terms each with 3 literals, performance evaluation with the solving time (ST), number of solved instances (SI), number of timeouts (TO), average number of branching decisions (D(A)), average number of unit propagations (P(A)) and the average number of conflicts (C(A))

Solver	ST	SI	TO	D(A)	P(A)	C(A)
Satisfiable Benchmark Set						
$DPLL_{NR\_F}$	169.861	99	0	29979.13	83177.88	29969.36
$DPLL_{R\_F}$	1495.094	99	0	265148.13	730542.03	263769.10
$CDCL_{R\_F}$	6019.026	70	29	348181.73	718620.80	270602.02
$CDCL_{R\_F\_CNF}$	10873.713	18	81	400048.77	46125526.05	119606.17
SAT4J	355.249	99	0	24897.67	6970444.00	22712.35
Unsatisfiable Benchmark Set						
$DPLL_{NR\_F}$	371.622	101	0	69360.25	192030.95	69361.25
$DPLL_{R\_F}$	5170.768	101	0	951018.38	2620153.44	946667.61
$CDCL_{R\_F}$	12120.204	0	101	691474.96	1439292.20	536616.08
$CDCL_{R\_F\_CNF}$	12120.63	0	101	479496.85	55210042.55	143715.06
SAT4J	802.786	101	0	51116.41	14393933.36	47212.17

Conclusions and future work

- The new solver was able to outperform Sat4j in specific benchmarks
- The other constraints aren't always beneficial
- Sat4j generally has significantly less branching decisions
  - Has a major impact on the solving time
- Sat4j is faster in pure CNF solving
  - An improvement in this area might lead to faster solving times for the new constraint types