# Modelling Satisfiability Problems Theory and Practice

**Valentin Mayer-Eichberger**
NICTA and Data61
University of New South Wales
Sydney, Australia

## Abstract

Boolean Satisfiability (SAT) solvers are a mature technology to solve hard combinatorial problems. The input to a SAT solver is the problem translated to propositional logic in conjunctive normal form (CNF). This thesis studies such translations and aims to make SAT solvers more accessible to non-encoding experts.

## 1 Research Problem

Many important problems in computer science have a combinatorial structure and are computationally difficult to solve. Among them are problems such as routing, timetabling, planning and scheduling, and problems from hardware and software synthesis and verification. Typically, these problems share a basic theoretical core: their computational complexity lies in the class of NP-Completeness, i.e. informally speaking, these are decision problems for which it is unlikely that a polynomial time algorithm exists (unless $NP = P$), however checking the correctness of the solution is computationally easy.

The canonical NP-Complete problem is Boolean Satisfiability (SAT) and attempts to solve this problem in practice unites a large research community. Due to major advances in SAT solving technology in the last two decades, it has become a promising approach to translate NP-Complete problems into SAT and solve them by highly optimised SAT solvers.

A major barrier to apply SAT solving is the translation of the problem to CNF due to the narrow representation format comprising a set of clauses in propositional logic. An important research program is thus the study of translations to CNF, their theoretical properties and the effectiveness for practical SAT solving.

**Motivating Case Study** A traditional benchmark problem in the operation research community is the *car sequencing* problem [Smith, ]. The problem is to sequence cars along a production line such that capacity constraints on a set of work stations are not exceeded. This is a well-studied problem often solved by established methods such as constraint programming, integer programming or local search. We were able to show that SAT solving is a competitive alternative for solving hard instances and our approach was able to show unsatisfiability for some open instances [Artigues *et al.*, 2014]. The main contribution is a sophisticated translation of the problem to CNF with strong theoretical properties. For comparison we also show that SAT solving using a naive translations to CNF is not competitive.

## 2 Contributions

SAT problems are usually given as set of constraints in some formalism. A rich modeling formalism is pursued in the field of *constraint programming* (CP), that allows general and global constraints to express a problem. CP solvers usually apply a richer set of reasoning and pruning techniques, and use special tailored constraint propagation algorithms in their engines. However, SAT solvers have shown to be competitive for certain types of constraint problems [Tamura *et al.*, 2009]. The SAT community studies compilations from such constraints to CNF.

### 2.1 Properties of CNF Representations

There are two main properties that characterize the quality of constraint decompositions: the size, i.e. the number of auxiliary variables and the number of clauses introduced by the translation. The second property is the consistency maintained by unit propagation (UP). Unit propagation is a reasoning technique typically performed in every node of the search tree.

Unit propagation removes literals in clauses that are known to be false, i.e. it resolves unit clauses with all clauses that share variables. This is repeated until no new unit clauses are produced in this process.

We are interested in what consistency UP maintains throughout the search on the CNF decomposition. More formally, given a constraint $C$ and its CNF decomposition $F_C$, the following equivalence is a desirable property, i.e. on any partial assignment $A$ on variables of $F_C$, it holds that UP computes the literal $l$ if and only if $l$ is logically entailed:

$$A \wedge F_C \vdash_{\text{UP}} l \iff A \wedge F_C \models l$$

If $A$ and $l$ contain variables from the original constraint and do not include the auxiliary variables introduced by the decomposition, we call this property *Generalized Arc Consistent* (GAC). A slightly weaker property is if $l = \bot$, i.e. UP

detects inconsistencies on all partial assignments. This definition can also be extended to partial assignments on auxiliary variables.

Higher consistency usually leads to more pruning of the search tree and as such fast SAT solving. However, the trade-off between size and consistency in CNF representations is hard to get right and requires experimentation and theoretical analysis.

### 2.2 Decision Diagrams and CNF Translations

In [Abío *et al.*, 2016] we study different translations from decisions diagrams such as Binary Decision Diagrams (BDDs), Multi-valued Decision Diagrams (MDDs) and Negation Normal Forms (NNFs) to CNF. Such decision diagrams provide a powerful tool for modelling complex constraints in discrete satisfaction and optimization problems. Generic CP propagators for these global constraints exist, but they are complex and hard to implement.

The most widely used CNF encoding of BDDs does not maintain GAC in arbitrary BDDs. We also distinguish between consistency on the primary variables of the constraint and auxiliary variables introduced by the compilation. This is yet an unexplored field and we consider multiple encodings and review their size and consistency. Furthermore, we introduce a refined encoding that has GAC consistency also on the auxiliary variables, i.e. the maximal consistency possible. Our experiments show that there is no single best encoding and it depends on the benchmark which encodings work best.

### 2.3 Combined Translations of Pseudo-Boolean Constraints

Pseudo-Boolean (PB) constraints appear in many applications. A PB constraint is of the form $\sum_{i=1}^{n} a_i \cdot X_i \leq k$ where $a_i, k \in \mathbb{Z}$ and $X_i \in \{0, 1\}$ for $i = 1 \dots n$. To translate this constraint to CNF is in itself already a challenging task. The literature usually considers translations of such constraints one by one without taking the rest of the problem into account. In [Abío *et al.*, 2015] we introduce a new approach by identifying cases where it is beneficial in terms of encoding size and consistency to translate by multiple constraints at once.

PB constraints often intersect with a set of binary clauses. If these binary clauses form a chain $x_1 \leftarrow x_2 \dots \leftarrow x_k$ it is likely that the translation benefits from an integrated approach. In this work we develop a framework where these combined translations are detected automatically.

In particular, we improve the BDD based decomposition of PBs by taking the additional information of such chains into account. The resulting BDD is more compact and the CNF decomposition of the BDD maintains GAC on the conjunction of the PB and the chains. We are able to show theoretically and empirically that this has positive effect on CNF size and consistency and improves SAT solving.

### 2.4 Translations of AtMostSeqCardinality

As mentioned in the introduction, we studied in depth CNF translations of the *car sequencing problem*. One central constraint in this problem is AtMostSeqCardinality , which models both the demand and capacity constraints. In previous CNF decompositions these constraints were translated separately. In [Artigues *et al.*, 2014] we we introduce a translation to CNF that maintains GAC by UP on that global constraint. We compare our SAT approach to a sophisticated CP global propagator for this constraint and are able to show it is more efficient. We also close several open instances in the CSPLIB [Smith, ].

## 3 Directions of Future Work

We plan to extend our studies as follows.

Pseudo Boolean Constraints: The three encodings of PB constraints presented in the seminal work of Sörensson and Eén [Eén and Sörensson, 2006] are based on BDDs, Adder Networks and Sorting Networks respectively. Subsequently, these approaches have been refined in the CP and SAT literature over the last decade. We are currently working on a survey article that reviews the last decade's result on CNF translations of PBs. Furthermore, we investigate generalisations to *Linear Constraints*, where variables have integer domain.

The constraint All-Different $([x_1, \dots x_n])$ enforces that the values taken by the integer variables are all different, i.e. that $x_i \neq x_j$ for $i < j$. This is a well studied constraint in the CP community and many filtering algorithms and decompositions are known. We plan to study CNF translations of All-Different to CNF and evaluate against native propagators.

## References


[Abío *et al.*, 2015] Ignasi Abío, Valentin Mayer-Eichberger, and Peter J. Stuckey. Encoding Linear Constraints with Implication Chains to CNF. In *Principles and Practice of Constraint Programming - CP*, 2015.

[Abío *et al.*, 2016] Ignasi Abío, Graeme Gange, Valentin Mayer-Eichberger, and Peter Stuckey. On CNF Encodings for Decision Diagrams. In *Integration of AI and OR Techniques in Constraint Programming, CPAIOR, to appear*, 2016.

[Artigues *et al.*, 2014] Christian Artigues, Emmanuel Hebrard, Valentin Mayer-Eichberger, Mohamed Siala, and Toby Walsh. SAT and hybrid models of the car sequencing problem. In *Integration of AI and OR Techniques in Constraint Programming, CPAIOR*, 2014.

[Eén and Sörensson, 2006] Niklas Eén and Niklas Sörensson. Translating Pseudo-Boolean Constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 2(1-4):1–26, 2006.

[Smith, ] Barbara Smith. CSPLib problem 001: Car sequencing. http://www.csplib.org/Problems/prob001.

[Tamura *et al.*, 2009] Naoyuki Tamura, Akiko Taga, Satoshi Kitagawa, and Mutsunori Banbara. Compiling finite linear CSP into SAT. *Constraints*, 14(2):254–272, 2009.