Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

# Just-in-Time Hierarchical Constraint Decomposition

**Valentin Mayer-Eichberger**
NICTA and University of New South Wales
Valentin.Mayer-Eichberger@nicta.com.au
Level 4, 223 Anzac Parade, Sydney, Australia
Website: vale1410.github.io Phone: +610405338860

**Abstract**

Lazy Clause Generation (LCG) solvers dominate the current constraint programming competitions. These solvers successfully combine systematic propagation based search, global constraints and conflict clause learning from SAT solving into a hybrid approach. My research project extends the LCG methodology by using a mix of eager and lazy encodings and a richer set of constraint decompositions. Global Constraints exhibit a whole hierarchy of different decomposition into more basic constraints. In our work we want to take advantage of such hierarchies and identify criteria on how constraints could be decomposed before and during search.

## Lazy Clause Generation and Extensions

Boolean Satisfiability Solving (SAT) and finite domain Constraint Programming (CP) solve hard combinatorial-structured problems. Pracical SAT solving emerged from the Model Checking and Verification community and constraint programming has some of its success in solving scheduling problems. One attempt to integrate both paradigms into a hybrid solver came with the idea of LCG solvers by (Ohrimenko, Stuckey, and Codish 2009). Early LCG systems collect a clause for each domain filtering of a constraint propagator which can then be used in SAT-style conflict clause resolution. The work of (Abío and Stuckey 2012; Abío et al. 2013) extends this approach to reactively decompose Pseudo-Boolean constraints *lazily* during search. If the number of generate clauses during search exceeds a certain limit they decompose to a compact encoding using auxiliary variables. We want to follow up on this strategy and extend it to other constraints.

Constraint decomposition faces two challenges: Firstly, the size of the decomposition might be to large. This is especially true in case of decompositions to conjunctive normal form (CNF). Encodings of size $O(n^3)$ in the domain size of integer variables might already exceed moderate size restrictions. Secondly, the decomposition loses the global view on the constraint and this might hinder propagation, i.e. not all inferences can be found by the filtering algorithms on the decomposed constraints. It is not surprising that larger decompositions tend to enforce higher consistency (more propagation). Thus, the key to a good decomposition is a good trade-off between size and consistency.

Contrary to the constraint decomposition approach we have the space-efficient propagation algorithms that often achieve higher consistency than a decomposition. Especially when domains are large, eager CNF compilation often exceed reasonable size restrictions. The drawback with pure propagation is the lack of auxiliary variables that often benefit the learning mechanism. To find the right balance between global constraint propagation and decomposition we want to take advantage of a fine-grained view on different levels of decomposition. Constraints can be decomposed into other simpler global constraints that maintain a middle way of representation size and consistency. Furthermore, such extensive analysis will be a fruitful source for auxiliary variables for learning beneficial clauses.

The *lazy decomposition* approach profits from extensive knowledge in constraint decompositions and SAT encodings. The next step in this direction of research is to target a healthy mix of global constraint propagation, explanations for conflict clause learning and decompositions into both non-clausal constraints and CNF. In the next section we will show such a hierarchy by listing decompositions of the ALL-DIFFERENT constraint. In addition to decomposition hierarchies, we will consider various criteria to support the decision of when to decompose and when to propagate such constraints (due to space limitation not mentioned in this extended abstract). We coin the name *Just in Time Hierarchical Constraint Decomposition* (JIT-HCD) for this methodology.

## A Concrete Example: ALL-DIFFERENT

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. In this section we describe the hierarchy of ALL-DIFFERENT decompositions.

The standard domain representation in LCG solvers uses the Boolean variables $[\![x = v]\!]$ and $[\![x \geq v]\!]$ for each value $v \in D(x)$ in the domain of variable $x$, with $[\![x \leq v]\!] \Leftrightarrow \neg[\![x \geq v + 1]\!]$. Let $D$ be union of all domains.

The following decompositions are direct translations of the definition of ALL-DIFFERENT:

**BinIn:** Binary inequalities: $x_i \neq x_j$ for $i < j$.

**BinCl:** Binary clauses: $\neg[\![x_i = v]\!] \vee \neg[\![x_j = v]\!]$ for $i < j$ and $v \in D(x_i) \cap D(x_j)$.

**Card:** Set of cardinalities: $\sum_{i=1}^{n}[\![x_i = v]\!] \leq 1$ for $v \in D$. In the special case of $|D| = n$ the equation becomes $\sum_{i=1}^{n}[\![x_i = v]\!] = 1$.

**Log:** Without going into details, this encoding uses the bit representation of the values and specifies that each pair of variables have to differ in at least one bit of their values.

The following decompositions introduce auxiliary variables (Boolean and Integer) and achieve a higher consistency on ALL-DIFFERENT.

**Feydy:** Let $c_v \in \{0, 1\}$ with $v \in V$ denote if value $v$ is taken by one of the variables, i.e. $c_v = \sum_{i=1}^{n}[\![x = v]\!]$ for all $v \in V$. Then we can decompose to a set of linear equations as in (Feydy and Stuckey 2009) by further introducing counter variables $s_v \in V$, with $s_v = s_{v-1} + c_v$. The set of equations $s_v = \sum_{i=1}^{n}[\![x \leq v]\!]$ enforces a weaker form of bounds consistency.

**Bounds:** Let $A_{ilu}$ be auxiliary Boolean variables identifying if $x_i \in [l, u]$, which can be encoded as $A_{ilu} \Leftrightarrow ([\![x_i \geq l]\!] \wedge \neg[\![x_i \geq u + 1]\!])$. The linear inequalities $\sum_{i=1}^{n} A_{ilu} \leq u - l + 1$ will enforce bounds consistency, $l < u$ and $l, u \in V$. See (Bessiere et al. 2009) for details.

The CARDINALITY and ATMOSTONE constraints used in the decompositions above can be translated to CNF in various ways. Without going into details, we denote common translations as **Sort**, **Count** and **Split** in this brief analysis.

Propagators on ALL-DIFFERENT that achieve a higher consistency can be explained by Hall sets. These are sets of variables that are known only to take values from a set (interval) of values that has the same size, i.e. these values are blocked for other variables. The following clauses can be extracted from such an explanation based propagator, see (Downing, Feydy, and Stuckey 2012) for details:

**Hall-I:** Given a Hall-set $H$ with the value interval $[l \dots u]$, an increased lower bound for $x_i \notin H$ can be explained by $\left([\![x_i \geq l]\!] \wedge \bigwedge_{h \in H}([\![x_h \geq l]\!] \wedge [\![x_h \leq u]\!])\right) \Rightarrow [\![x_i \geq u + 1]\!]$.

**Hall-S:** Likewise if a Hall-set $H$ with values $S$ is found, the propagation can be explained by $\left(\bigwedge_{h \in H, v \in V \setminus S}[\![x_h \neq v]\!]\right) \Rightarrow [\![x_i \neq j]\!]$.

Figure 1 visualizes a possible hierarchy of decompositions of ALL-DIFFERENT that were presented in this section. We choose to outline the graph in three levels where the first is the constraint itself, the second are different decompositions in linear constraints, and on the bottom are decompositions in propositional clauses. An Arc corresponds to a choice in a decomposition.

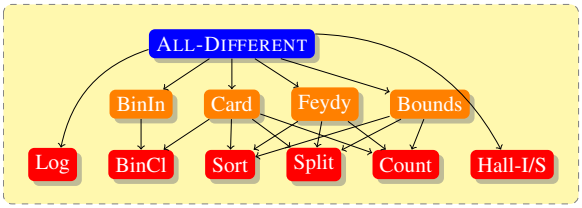


Figure 1: Hierarchy of ALL-DIFFERENT constraint decompositions.

## Preliminary Results and Next Steps

Preliminary work for this PhD has been in eager SAT encodings for Pseudo Boolean and SEQUENCE constraint (Abío et al. 2012; Artigues et al. 2014). Currently, the focus is on decompositions of ALL-DIFFERENT to extend the LCG solver CHUFFED to take advantage of the JIT-HCD approach. We plan to contribute strategies on when and how to decompose ALL-DIFFERENT using the rich hierarchy available as described in the previous section and possibly extend this to other global constraints.

## Supervisors and Collaborators

This PhD is supervised by Toby Walsh with co-supervisors Michael Thielscher and Serge Gaspers (University of New South Wales and NICTA). The current project on extensions of LCG solvers is done in collaboration with Peter Stuckey and Ignasi Abío (University of Melbourne and NICTA).

## References


Abío, I., and Stuckey, P. 2012. Conflict directed lazy decomposition. *CP* 70–85.

Abío, I.; Nieuwenhuis, R.; Oliveras, A.; Rodríguez-Carbonell, E.; and Mayer-Eichberger, V. 2012. A New Look at BDDs for Pseudo-Boolean Constraints. *J. Artif. Intell. Res. (JAIR)* 45:443–480.

Abío, I.; Nieuwenhuis, R.; Oliveras, A.; Rodríguez-Carbonell, E.; and Stuckey, P. J. 2013. To Encode or to Propagate? The Best Choice for Each Constraint in SAT. In *CP*, 97–106.

Artigues, C.; Hebrard, E.; Mayer-Eichberger, V.; Siala, M.; and Walsh, T. 2014. SAT and Hybrid models of the Car-Sequencing Problem. In *CPAIOR*.

Bessiere, C.; Katsirelos, G.; Narodytska, N.; Quimper, C.-G.; Walsh, T.; et al. 2009. Decompositions of all different, global cardinality and related constraints. In *IJCAI*, 419–424.

Downing, N.; Feydy, T.; and Stuckey, P. J. 2012. Explaining alldifferent. In *ACSC*, 115–124.

Feydy, T., and Stuckey, P. J. 2009. Lazy clause generation reengineered. In *CP*, 352–366. Springer Berlin Heidelberg.

Ohrimenko, O.; Stuckey, P.; and Codish, M. 2009. Propagation via lazy clause generation. *Constraints* 14(3):357–391.