On the non-Degeneracy of Unsatisfiability Proof Graphs produced by SAT Solvers

Rohan Fossé - Laurent Simon

Formal Method - Univ. Bordeaux, LaBRI, France

Preliminaries

- **Definition**

  Let $\Phi(a, b, ..)$ be a boolean formula.
  Is there an **interpretation** of $(a, b, ..)$ that satisfies Φ ?

- **Notations**

  Literals

  A literal $(a, b, \ldots)$ is either a boolean variable x or the negation of a boolean variable $\neg x$

  Clauses

  A clause C is a disjunction of literals *i.e*:
  $C = a \vee b \vee .. \vee z$

Formula

A formula Φ is a conjunction of clauses *i.e*: $\Phi = C_1 \wedge C_2 \wedge \ldots \wedge C_m$

Example

$\Phi = (a \vee \neg b) \wedge b \wedge (\neg a \vee \neg b)$

Resolution rule [Robinson '65]

Let C_1 and C_2 be two clauses, the resolution rule gives us:

$$(C_1 \vee x) \wedge (C_2 \vee \neg x) \vdash C_1 \vee C_2$$

Example

We apply the resolution rule on d:

$$\overbrace{(a \vee b \vee c \vee d)}^{C_1 \vee d} \wedge \overbrace{(\neg d \vee e \vee f)}^{C_2 \vee \neg d}$$
$$\vdash \underbrace{a \vee b \vee c \vee e \vee f}_{C_1 \vee C_2}$$

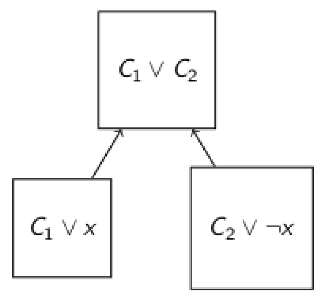

Figure: Representation of the resolution

Resolution graph

Definition

The resolution graph is a directed acyclic graph (or **DAG**) such that:

- Leaves are **initials** clauses;
- Internal nodes are **learnts** clauses;
- The root is the **empty** clause.

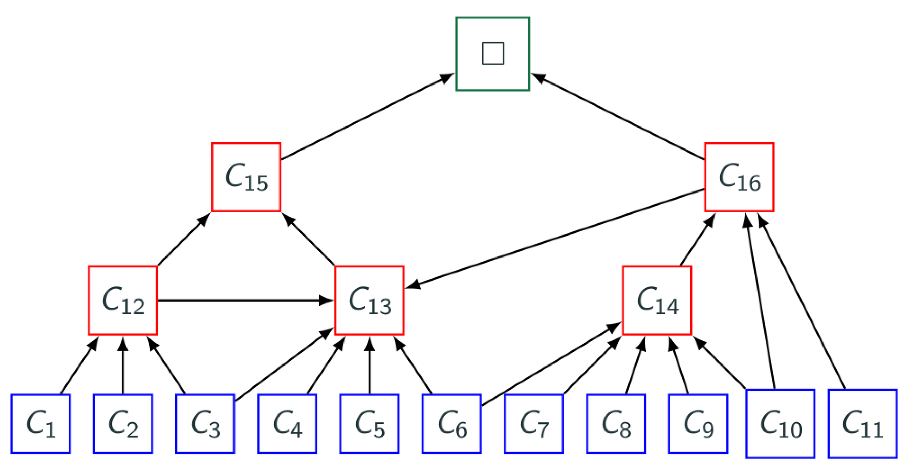

Figure: Sequence of resolution with a graph representation

We call useful a clause necessary for the proof, and useless otherwise.

Example C_{14} is useless here.

Representation of a real proof

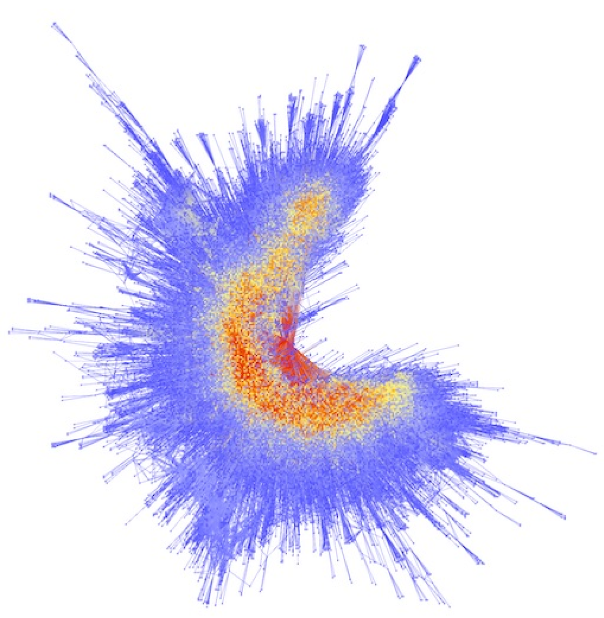

Figure: Force-Directed layout of the resolution Graph for the benchmark `een-pico-prop-05`. The color shows the **degree** of each node.

Information

Formula
clauses: 55585
variables: 50076

Conflicts
conflicts: 59792
CPU time: 6s

Graph
vertices: 51274
edges: 960620

Aim of this work

- Extend the work in [2] and focusing on the existence and importance of a very dense subgraph (the **K-core**) in all the proofs produced by SAT solvers.

Characterization of K-core

K-core

A *k*-core of a graph G is an undirected subgraph in which all vertices have degree at least *k*.

Coreness

The coreness of a vertex is *k* *iff* he belongs to a *k*-core but not to any *k+1*-core.

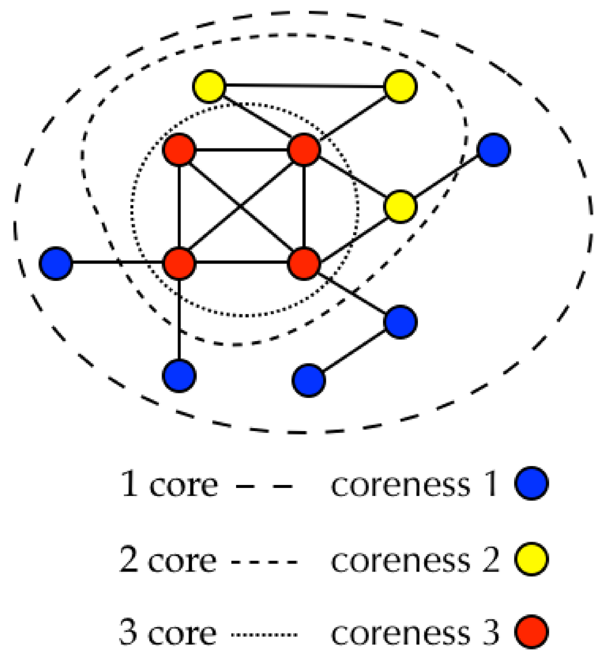

Figure: Graphical representation of the coreness

Experiments conditions

- 60 problems from the 2012-2017 SAT competitions;
- Run on the Cluster of Xeon E7-4870 processors from the *Mesocentre Aquitain de Calcul Intensif*;

Objectives and overview of the results

Objectives

- Identify during the analysis which clauses will be useful;
- Guess which variables will occur in the last learnt clauses, just before deriving the **empty** clause.

Overview of the results

- we correctly guess at least **90%** of useful learnt clauses for **half** of the problems;
- Some results of our prediction for literals occurring over 60 problems

Below, We count the number of problems in which **one** of the Top-Y variables occurs in **one** of the last X learnt clauses.

	20	**50**	**100**	**1000**
5	27	37	45	53
10	42	47	50	54
20	49	51	51	55

Table: Top-Y variables (*rows*) w.r.t the last X learnt clauses (*columns*)

References

Fossé R., Simon L.
On the Non-degeneracy of Unsatisfiability Proof Graphs Produced by SAT Solvers
Principles and Practice of Constraint Programming,Springer International Publishing,2018

Simon L.
Post Mortem Analysis of SAT Solver Proofs
POS-14. Fifth Pragmatics of SAT workshop

École doctorale Mathématiques et informatique