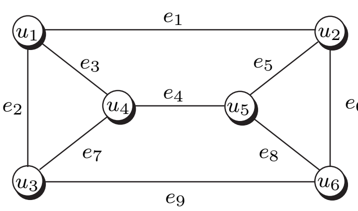
SAT Benchmarks based on 3-Regular Graphs

Harri Haanpää Matti Järvisalo* Petteri Kaski Ilkka Niemelä

Unsatisfiable instances. For a positive integer $n \geq 2$, let A be a $2n \times 3n$ incidence matrix of a random $2n$-vertex 3-regular graph, and select $b = (b_1, b_2, \ldots, b_{2n}) \in \{0,1\}^{2n}$ uniformly at random so that $\sum_{i=1}^{2n} b_i \equiv 1 \pmod 2$. The resulting system $Ax \equiv b \pmod 2$ clearly has no solution (take the sum of the equations). We then transform the system into a CNF formula by introducing for every equation $x_{i_1} + x_{i_2} + x_{i_3} \equiv b_j \pmod 2$ a set of four clauses that forbid the combinations of truth values that violate the equation; for example, the equation $x_1 + x_2 + x_3 \equiv 0 \pmod 2$ transforms into the clauses $\{\bar{x}_1, \bar{x}_2, \bar{x}_3\}$, $\{\bar{x}_1, x_2, x_3\}$, $\{x_1, \bar{x}_2, x_3\}$, and $\{x_1, x_2, \bar{x}_3\}$. The resulting formula has $v = 3n$ variables and $c = 8n$ clauses of length 3. The construction is illustrated in Fig. 1.



$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{gathered} \{\{\bar{x}_1, \bar{x}_2, x_3\}, \{\bar{x}_1, x_2, \bar{x}_3\}, \{x_1, \bar{x}_2, \bar{x}_3\}, \{x_1, x_2, x_3\}, \\ \{\bar{x}_1, \bar{x}_5, \bar{x}_6\}, \{\bar{x}_1, x_5, x_6\}, \{x_1, \bar{x}_5, x_6\}, \{x_1, x_5, \bar{x}_6\}, \\ \{\bar{x}_2, \bar{x}_7, x_9\}, \{\bar{x}_2, x_7, \bar{x}_9\}, \{x_2, \bar{x}_7, \bar{x}_9\}, \{x_2, x_7, x_9\}, \\ \{\bar{x}_3, \bar{x}_4, x_7\}, \{\bar{x}_3, x_4, \bar{x}_7\}, \{x_3, \bar{x}_4, \bar{x}_7\}, \{x_3, x_4, x_7\}, \\ \{\bar{x}_4, \bar{x}_5, \bar{x}_8\}, \{\bar{x}_4, x_5, x_8\}, \{x_4, \bar{x}_5, x_8\}, \{x_4, x_5, \bar{x}_8\}, \\ \{\bar{x}_6, \bar{x}_8, \bar{x}_9\}, \{\bar{x}_6, x_8, x_9\}, \{x_6, \bar{x}_8, x_9\}, \{x_6, x_8, \bar{x}_9\}\} \end{gathered}$$

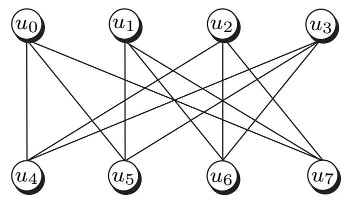
Figure 1: Constructing an unsatisfiable instance

*Corresponding author. Address: Helsinki University of Technology, Laboratory for Theoretical Computer Science, P.O. Box 5400, FI-02015 TKK, Finland. Email: `matti.jarvisalo@tkk.fi`.

Satisfiable instances. For an integer $n \geq 3$, select a random $2n$-vertex 3-regular bipartite graph G. Labeling the vertices suitably, we can assume that the adjacency matrix of G has the form

$$M = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix},$$

where A is an $n \times n$ matrix with exactly three ones in every row and every column. After the matrix A has been constructed, we select uniformly at random a $z \in \{0,1\}^n$ and let $b \in \{0,1\}^n$ such that $b \equiv Az \pmod 2$. Then, we transform the system $Ax \equiv b \pmod 2$ into a CNF formula as above. The formula has $v = n$ variables and $c = 4n$ clauses of length 3. The construction is illustrated in Fig. 2. If an instance with a unique satisfying truth assignment is required, then the matrix A must be invertible modulo 2.



$$A = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \quad z = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\begin{gathered} \{\{\bar{x}_1, \bar{x}_2, \bar{x}_4\}, \{\bar{x}_1, x_2, x_4\}, \{x_1, \bar{x}_2, x_4\}, \{x_1, x_2, \bar{x}_4\}, \\ \{\bar{x}_2, \bar{x}_3, \bar{x}_4\}, \{\bar{x}_2, x_3, x_4\}, \{x_2, \bar{x}_3, x_4\}, \{x_2, x_3, \bar{x}_4\}, \\ \{\bar{x}_1, \bar{x}_3, x_4\}, \{\bar{x}_1, x_3, \bar{x}_4\}, \{x_1, \bar{x}_3, \bar{x}_4\}, \{x_1, x_3, x_4\}, \\ \{\bar{x}_1, \bar{x}_2, \bar{x}_3\}, \{\bar{x}_1, x_2, x_3\}, \{x_1, \bar{x}_2, x_3\}, \{x_1, x_2, \bar{x}_3\}\} \end{gathered}$$

Figure 2: Constructing a satisfiable instance

Hiding Linearity. A system of linear equations modulo 2 cannot in itself be considered hard; both the existence and nonexistence of a solution can easily be demonstrated through Gaussian elimination. There exist specialized SAT solvers that recognize the linear structure in a set of clauses and apply a form of Gaussian elimination and/or equivalence reasoning to expedite the solution process. However, it appears that a simple disguise suffices to fool contemporary SAT solvers. In this connection we camouflage the instance as follows.

The *simple camouflage* is obtained by selecting a minimal set of variables such that every clause contains at least one selected variable. For each selected variable, x, we introduce a new variable, y, and then substitute each occurrence of x (respectively, $\bar{x}$) in the clauses with $x \wedge y$ (respectively, $\overline{x \wedge y} \equiv \bar{x} \vee \bar{y}$). After all the substitutions have been performed, we expand any conjuncts inside the clauses out of the clauses to obtain a camouflaged set of clauses. For the family of unsatisfiable instances, an appropriate set of variables is computed using an algorithm for maximum matching in the underlying (regular bipartite) graph. For the satisfiable instances, the greedy approximation algorithm for the set cover problem is employed; each vertex in the underlying graph associated with a variable covers the adjacent vertices associated with linear equations/clauses.

Available Benchmarks. The following set of benchmarks are available at

`http://www.tcs.hut.fi/~mjj/benchmarks/`.

- mod2-rand3bip-sat

  Satisfiable instances with the number of variables $v = 200, 210, \ldots, 300$. For each v there are 15 instances. Each instance has a unique satisfying truth assignment. The underlying graphs are random 3-regular bipartite graphs with.

- mod2c-rand3bip-sat

  As mod2-rand3bip-sat but with "simple camouflage" applied on the instances.

- mod2-3g14-sat.cnf

  A satisfiable instance with 192 variables. The underlying graph is the smallest known 3-regular graph with girth[1] 14.

- mod2c-3g14-sat.cnf

  As mod2-3g14-sat but with "simple camouflage" applied on the instance. The number of variables is 266.

- mod2-rand3bip-unsat

  Unsatisfiable instances with the number of variables $v = 90, 105, 120, 135, 150$. For each v there are 15 instances. The underlying graphs are random 3-regular bipartite graphs.

- mod2c-rand3bip-unsat

  As mod2-rand3bip-unsat but with "simple camouflage" applied on the instances.

- mod2-3cage-unsat

  The underlying graphs are $(3, g)$-cages[2] with girth $g \in \{9, 10, 11, 12\}$. 18 $(3, 9)$-cages, 3 $(3, 10)$-cages, 1 $(3, 11)$-cages, and 1 $(3, 12)$-cage exist.

- mod2c-3cage-unsat

  As mod2-3cage-unsat but with "simple camouflage" applied on the instances.

On Instance Hardness. Our experiments suggest the following. For instances based on random 3-regular graphs, unsatisfiable instances with 120 variables and satisfiable instances with 240 variables already require in the order of 2^{20} decisions to solve. This includes state-of-the-art DPLL-based solvers as well as the local search solver `WalkSAT`. Reflecting this to run times, the run times in seconds as reported by `zChaff` version 2004.5.13 on the randomly generated mod2-rand3bip-sat instances with 220 variables on a machine with a 1.667GHz AMD Athlon processor and 1 GB of RAM are shown Table 1. The mean of the run times is approximately 100 minutes.

Compared to random graphs, instances based on cages are still harder. Moreover, while e.g. `march_eq` can solve uncamouflaged instances in a matter of seconds due to Gaussian elimination -type equivalence reasoning, the camouflaged instances seems at least as hard for equivalence reasoners as the uncamouflaged instances are for solvers with no equivalence reasoning.

More detailed experiments appear in [1].

References

[1] Harri Haanpää, Matti Järvisalo, Petteri Kaski, and Ilkka Niemelä. Hard satisfiability

[1] The *girth* of a graph is the length of its shortest cycle.

[2] For $d, g \geq 3$, a (d, g)-*cage* is a d-regular graph of girth g with the minimum possible number of vertices.

Table 1: `zChaff` on mod2-rand3bip-sat instances with 220 variables on a 1.667GHz AMD Athlon processor with 1-GB RAM.

instance	**time (s)**
1	5595.81
2	9852.17
3	421.08
4	489.38
5	8858.91
6	1554.91
7	3087.83
8	2900.33
9	8428.14
10	21338.50
11	12418.90
12	18320.80
13	1217.79
14	856.29
15	12302.10

instances from random regular graphs. 2005. Manuscript.