On Reducing Maximum Independent Set to Minimum Satisfiability

Alexey Ignatiev[1], Antonio Morgado[1], and Joao Marques-Silva[1,2]

[1] INESC-ID/IST, Lisbon, Portugal
[2] CASL/CSI, University College Dublin, Ireland

17th International Conference on
Theory and Applications of Satisfiability Testing

Vienna, Austria
July 14, 2014

Problems statements

MIS: compute the **largest** number of pairwise non-connected vertices in $\mathcal{G}$.
MVC: compute the **smallest** number of vertices in $\mathcal{G}$ that are incident to all edges of $\mathcal{G}$.

Problems statements

MIS: compute the **largest** number of pairwise non-connected vertices in $\mathcal{G}$.
MVC: compute the **smallest** number of vertices in $\mathcal{G}$ that are incident to all edges of $\mathcal{G}$.

$$\Downarrow$$

Given $\mathcal{G}$, $\mathcal{I} \subseteq \mathcal{G}$ is an MIS solution $\Leftrightarrow$ $\mathcal{G} \setminus \mathcal{I}$ is an MVC solution.

Problems statements

MIS: compute the **largest** number of pairwise non-connected vertices in $\mathcal{G}$.
MVC: compute the **smallest** number of vertices in $\mathcal{G}$ that are incident to all edges of $\mathcal{G}$.

$$\Downarrow$$

Given $\mathcal{G}$, $\mathcal{I} \subseteq \mathcal{G}$ is an MIS solution $\Leftrightarrow$ $\mathcal{G} \setminus \mathcal{I}$ is an MVC solution.

$$\textbf{MaxClq} = \overline{\textbf{MIS}}$$

Problems statements

MIS: compute the **largest** number of pairwise non-connected vertices in $\mathcal{G}$.
MVC: compute the **smallest** number of vertices in $\mathcal{G}$ that are incident to all edges of $\mathcal{G}$.

$$\Downarrow$$

Given $\mathcal{G}$, $\quad \mathcal{I} \subseteq \mathcal{G}$ is an MIS solution $\quad \Leftrightarrow \quad \mathcal{G} \setminus \mathcal{I}$ is an MVC solution.

$$\textbf{MaxClq} = \overline{\textbf{MIS}}$$

MinSAT: compute the **smallest** number of simultaneously **satisfied** clauses in $\mathcal{F}$.
MaxFalse: compute the **largest** number of simultaneously **falsified** clauses in $\mathcal{F}$.

Problems statements

MIS: compute the **largest** number of pairwise non-connected vertices in $\mathcal{G}$.
MVC: compute the **smallest** number of vertices in $\mathcal{G}$ that are incident to all edges of $\mathcal{G}$.

$$\Downarrow$$

Given $\mathcal{G}$, $\mathcal{I} \subseteq \mathcal{G}$ is an MIS solution $\Leftrightarrow$ $\mathcal{G} \setminus \mathcal{I}$ is an MVC solution.

$$\textbf{MaxClq} = \overline{\textbf{MIS}}$$

MinSAT: compute the **smallest** number of simultaneously **satisfied** clauses in $\mathcal{F}$.
MaxFalse: compute the **largest** number of simultaneously **falsified** clauses in $\mathcal{F}$.

$$\Downarrow$$

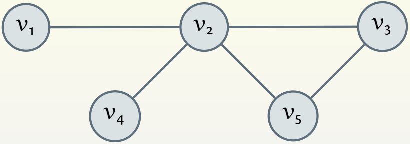
Given $\mathcal{F}$, $\mathcal{M} \subseteq \mathcal{F}$ is a MaxFalse solution $\Leftrightarrow$ $\mathcal{F} \setminus \mathcal{M}$ is a MinSAT solution.

MIS and MaxFalse

MIS $\leftrightarrow$ **MaxFalse**

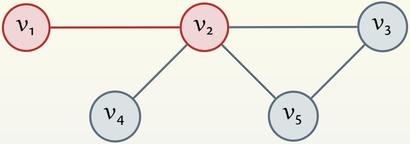
MVC $\leftrightarrow$ **MinSAT**

Basic reduction



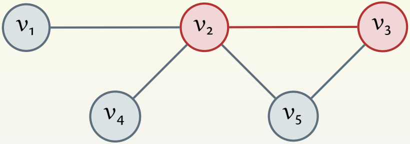
$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & x_{1,2} \\ c_2 & = & \neg x_{1,2} \vee x_{2,3} \vee x_{2,4} \vee x_{2,5} \\ c_3 & = & \neg x_{2,3} \vee x_{3,5} \\ c_4 & = & \neg x_{2,4} \\ c_5 & = & \neg x_{2,5} \vee \neg x_{3,5} \end{array} \right\}$$

Basic reduction



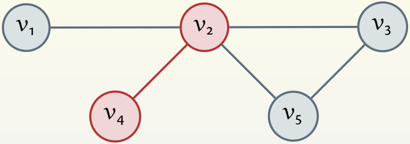
$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & x_{1,2} \\ c_2 & = & \neg x_{1,2} \vee x_{2,3} \vee x_{2,4} \vee x_{2,5} \\ c_3 & = & \neg x_{2,3} \vee x_{3,5} \\ c_4 & = & \neg x_{2,4} \\ c_5 & = & \neg x_{2,5} \vee \neg x_{3,5} \end{array} \right\}$$

Basic reduction



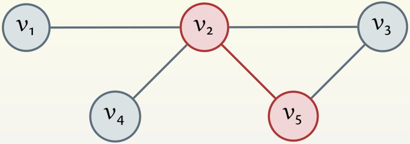
$$\mathcal{F} = \left\{ \begin{array}{rcl} c_1 & = & x_{1,2} \\ c_2 & = & \neg x_{1,2} \vee x_{2,3} \vee x_{2,4} \vee x_{2,5} \\ c_3 & = & \neg x_{2,3} \vee x_{3,5} \\ c_4 & = & \neg x_{2,4} \\ c_5 & = & \neg x_{2,5} \vee \neg x_{3,5} \end{array} \right\}$$

Basic reduction



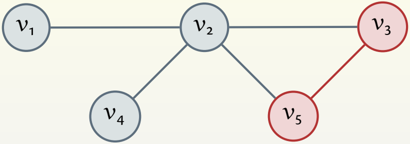
$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & x_{1,2} \\ c_2 & = & \neg x_{1,2} \vee x_{2,3} \vee x_{2,4} \vee x_{2,5} \\ c_3 & = & \neg x_{2,3} \vee x_{3,5} \\ c_4 & = & \neg x_{2,4} \\ c_5 & = & \neg x_{2,5} \vee \neg x_{3,5} \end{array} \right\}$$

Basic reduction



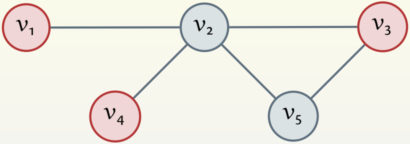
$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & x_{1,2} \\ c_2 & = & \neg x_{1,2} \vee x_{2,3} \vee x_{2,4} \vee x_{2,5} \\ c_3 & = & \neg x_{2,3} \vee x_{3,5} \\ c_4 & = & \neg x_{2,4} \\ c_5 & = & \neg x_{2,5} \vee \neg x_{3,5} \end{array} \right\}$$

Basic reduction



$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & x_{1,2} \\ c_2 & = & \neg x_{1,2} \vee x_{2,3} \vee x_{2,4} \vee x_{2,5} \\ c_3 & = & \neg x_{2,3} \vee x_{3,5} \\ c_4 & = & \neg x_{2,4} \\ c_5 & = & \neg x_{2,5} \vee \neg x_{3,5} \end{array} \right\}$$

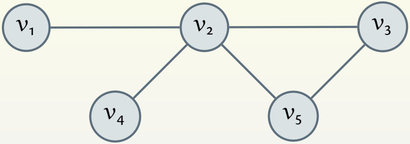
Basic reduction



$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & x_{1,2} \\ c_2 & = & \neg x_{1,2} \vee x_{2,3} \vee x_{2,4} \vee x_{2,5} \\ c_3 & = & \neg x_{2,3} \vee x_{3,5} \\ c_4 & = & \neg x_{2,4} \\ c_5 & = & \neg x_{2,5} \vee \neg x_{3,5} \end{array} \right\}$$

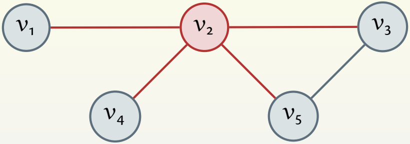
Given a graph $\mathcal{G} = (V, E)$, **basic reduction** constructs a formula $\mathcal{F}$ with exactly $|V|$ clauses and $|E|$ variables.

Greedy reduction



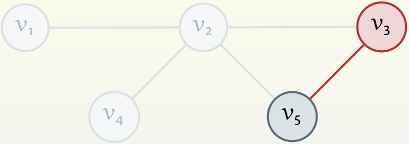
$$\mathcal{F} = \left\{ \begin{array}{ccl} c_1 & = & \neg x_2 \\ c_2 & = & x_2 \\ c_3 & = & \neg x_2 \vee x_3 \\ c_4 & = & \neg x_2 \\ c_5 & = & \neg x_2 \vee \neg x_3 \end{array} \right\}$$

Greedy reduction



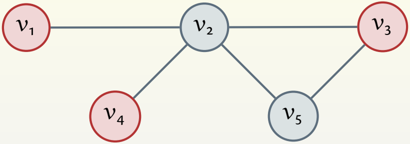
$$\mathcal{F} = \left\{ \begin{array}{rcl} c_1 & = & \neg x_2 \\ c_2 & = & x_2 \\ c_3 & = & \neg x_2 \vee x_3 \\ c_4 & = & \neg x_2 \\ c_5 & = & \neg x_2 \vee \neg x_3 \end{array} \right\}$$

Greedy reduction



$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & \neg x_2 \\ c_2 & = & x_2 \\ c_3 & = & \neg x_2 \vee x_3 \\ c_4 & = & \neg x_2 \\ c_5 & = & \neg x_2 \vee \neg x_3 \end{array} \right\}$$

Greedy reduction



$$\mathcal{F} = \left\{ \begin{array}{lcl} c_1 & = & \neg x_2 \\ c_2 & = & x_2 \\ c_3 & = & \neg x_2 \vee x_3 \\ c_4 & = & \neg x_2 \\ c_5 & = & \neg x_2 \vee \neg x_3 \end{array} \right\}$$

Given a graph $\mathcal{G} = (V, E)$, **greedy reduction** constructs a formula $\mathcal{F}$ with exactly $|V|$ clauses and $\leqslant |V|$ variables.

Variable compatibility

- original idea — *compatible states* in finite-state machines simplification

Variable compatibility

- original idea — *compatible states* in finite-state machines simplification
- compatible variables can replace each other

Variable compatibility

- original idea — *compatible states* in finite-state machines simplification
- compatible variables can replace each other
- variable compatibility rules:
  1. no tautology

     given a clause $\neg x_1 \vee x_2$, variables x_1 and x_2 are not compatible

Variable compatibility

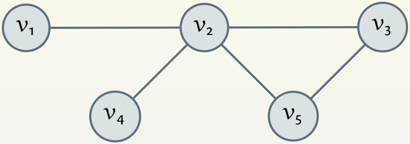
- original idea — *compatible states* in finite-state machines simplification
- compatible variables can replace each other
- variable compatibility rules:
  1. no tautology

     given a clause $\neg x_1 \vee x_2$, variables x_1 and x_2 are not compatible

  2. no new connection

     given two clauses $\neg x_1 \vee x_2$ and $\neg x_3$, variables x_2 and x_3 are not compatible

Variable compatibility[1]



(a) Graph $\mathcal{G}$

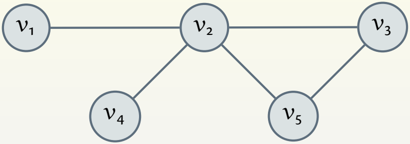
$$
\begin{aligned}
c_1 &= x_1 \\
c_2 &= \neg x_1 \vee x_2 \vee x_3 \vee x_4 \\
c_3 &= \neg x_2 \vee x_5 \\
c_4 &= \neg x_3 \\
c_5 &= \neg x_4 \vee \neg x_5
\end{aligned}
$$

(b) Set of clauses for $\mathcal{G}$

	x_1	x_2	x_3	x_4	x_5
x_1	–				
x_2	$*_2$ $*_{1,3}$	–			
x_3	$*_2$ $*_{1,4}$		–		
x_4	$*_2$ $*_{1,5}$			–	
x_5	$*_{1,5}$	$*_3$	$*_{3,4}$		–

[1] Literal compatibility is similar.

Variable compatibility[1]



(a) Graph $\mathcal{G}$

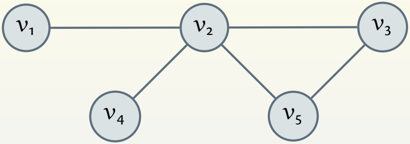
$$
\begin{aligned}
c_1 &= x_1 \\
c_2 &= \neg x_1 \vee x_2 \vee x_3 \vee x_4 \\
c_3 &= \neg x_2 \vee x_5 \\
c_4 &= \neg x_3 \\
c_5 &= \neg x_4 \vee \neg x_5
\end{aligned}
$$

(b) Set of clauses for $\mathcal{G}$

	x_1	x_2	x_3	x_4	x_5
x_1	–				
x_2	$*_{2}\ *_{1,3}$	–			
x_3	$*_{2}\ *_{1,4}$		–		
x_4	$*_{2}\ *_{1,5}$			–	
x_5	$*_{1,5}$	$*_{3}$	$*_{3,4}$		–

[1] Literal compatibility is similar.

Variable compatibility[1]



(a) Graph $\mathcal{G}$

$$
\begin{aligned}
c_1 &= x_1 \\
c_2 &= \neg x_1 \vee x_2 \\
c_3 &= \neg x_2 \vee x_5 \\
c_4 &= \neg x_2 \\
c_5 &= \neg x_2 \vee \neg x_5
\end{aligned}
$$

(b) Set of clauses for $\mathcal{G}$

	x_1	x_2	x_3	x_4	x_5
x_1	–				
x_2	$*_{2}$ $*_{1,3}$	–			
x_3	$*_{2}$ $*_{1,4}$		–		
x_4	$*_{2}$ $*_{1,5}$			–	
x_5	$*_{1,5}$	$*_{3}$	$*_{3,4}$		–

[1] Literal compatibility is similar.

Weighting clauses and removing duplicates

		basic			greedy			greedy+vc		
		vars	clauses	time[2]	vars	clauses	time[2]	vars	clauses	time[2]
Instance	c-fat200-1	18366	200	0.4	188	200	0.05	35	37	0
	c-fat200-2	16665	200	0.75	176	200	0.07	16	18	0
	c-fat200-5	11427	200	0.96	142	200	0.07	5	7	0
	c-fat500-1	120291	500	—	486	500	0.63	78	80	0
	c-fat500-10	78123	500	—	374	500	0.53	6	8	0
	c-fat500-2	115611	500	—	474	500	0.51	38	40	0
	c-fat500-5	101559	500	—	436	500	0.37	14	16	0

[2] Running time for MinSatz.

Experimental evaluation

1. Benchmarks (233 in total):

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver
   - MinSatz — branch and bound MinSAT solver

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver
   - MinSatz — branch and bound MinSAT solver
   - MaxSatz — branch and bound MaxSAT solver

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver
   - MinSatz — branch and bound MinSAT solver
   - MaxSatz — branch and bound MaxSAT solver
   - MiFuMaX — Fu&Malik algorithm for MaxSAT (**best** for MS industrial in 2013)

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver
   - MinSatz — branch and bound MinSAT solver
   - MaxSatz — branch and bound MaxSAT solver
   - MiFuMaX — Fu&Malik algorithm for MaxSAT (**best** for MS industrial in 2013)
3. Machine configuration:
   - Intel Xeon 5160@3GHz with 4GB RAM

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver
   - MinSatz — branch and bound MinSAT solver
   - MaxSatz — branch and bound MaxSAT solver
   - MiFuMaX — Fu&Malik algorithm for MaxSAT (**best** for MS industrial in 2013)
3. Machine configuration:
   - Intel Xeon 5160@3GHz with 4GB RAM
   - running Fedora Linux

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver
   - MinSatz — branch and bound MinSAT solver
   - MaxSatz — branch and bound MaxSAT solver
   - MiFuMaX — Fu&Malik algorithm for MaxSAT (**best** for MS industrial in 2013)
3. Machine configuration:
   - Intel Xeon 5160@3GHz with 4GB RAM
   - running Fedora Linux
   - 2GB memout

Experimental evaluation

1. Benchmarks (233 in total):
   - **crafted** MaxClq instances: DIMACS MaxClq, FRB, etc.
   - MIS instances obtained from Binate Covering Problem benchmarks (**BCP**)
2. Tools:
   - MaxCLQ — native MaxClq solver
   - MinSatz — branch and bound MinSAT solver
   - MaxSatz — branch and bound MaxSAT solver
   - MiFuMaX — Fu&Malik algorithm for MaxSAT (**best** for MS industrial in 2013)
3. Machine configuration:
   - Intel Xeon 5160@3GHz with 4GB RAM
   - running Fedora Linux
   - 2GB memout
4. Timeout value — 3600 seconds

Experimental results

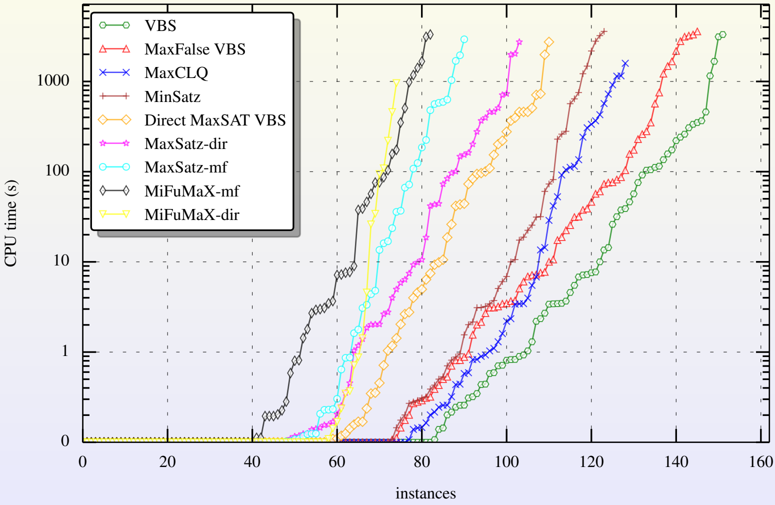
—	CLQ	MinSz	MaxSz-d	MaxSz-mf	FM-d	FM-mf	VBS-d	VBS-mf	VBS
Native	✓								✓
MaxFalse-based		✓		✓		✓		✓	✓
Direct MaxSAT			✓		✓		✓		✓

Experimental results

—	CLQ	MinSz	MaxSz-d	MaxSz-mf	FM-d	FM-mf	VBS-d	VBS-mf	VBS
Native	✓								✓
MaxFalse-based		✓		✓		✓		✓	✓
Direct MaxSAT			✓		✓		✓		✓

—	CLQ	MinSz	MaxSz-d	MaxSz-mf	FM-d	FM-mf	VBS-d	VBS-mf	VBS
Crafted Clq	**66**	59	43	36	19	30	45	74	76
BCP	63	**65**	61	55	56	53	66	72	76
Total	**129**	124	104	91	75	83	111	146	152

Performance of the considered approaches



Summary and future work

1. proposed a reduction from MIS to MinSAT

Summary and future work

1. proposed a reduction from MIS to MinSAT
2. heuristics to reduce obtained MinSAT formulas

Summary and future work

1. proposed a reduction from MIS to MinSAT
2. heuristics to reduce obtained MinSAT formulas
3. experimental results:
   - comparable to native MaxClq solvers

Summary and future work

1. proposed a reduction from MIS to MinSAT
2. heuristics to reduce obtained MinSAT formulas
3. experimental results:
   - comparable to native MaxClq solvers
   - outperforms MaxSAT-based approaches

Summary and future work

1. proposed a reduction from MIS to MinSAT
2. heuristics to reduce obtained MinSAT formulas
3. experimental results:
   - comparable to native MaxClq solvers
   - outperforms MaxSAT-based approaches
   - portfolios of solvers

Summary and future work

1. proposed a reduction from MIS to MinSAT
2. heuristics to reduce obtained MinSAT formulas
3. experimental results:
   - comparable to native MaxClq solvers
   - outperforms MaxSAT-based approaches
   - portfolios of solvers

1. further improvements to the proposed MinSAT models

Summary and future work

1. proposed a reduction from MIS to MinSAT
2. heuristics to reduce obtained MinSAT formulas
3. experimental results:
   - comparable to native MaxClq solvers
   - outperforms MaxSAT-based approaches
   - portfolios of solvers

1. further improvements to the proposed MinSAT models
2. portfolios of solvers for NP-hard graph problems

Summary and future work

1. proposed a reduction from MIS to MinSAT
2. heuristics to reduce obtained MinSAT formulas
3. experimental results:
   - comparable to native MaxClq solvers
   - outperforms MaxSAT-based approaches
   - portfolios of solvers

1. further improvements to the proposed MinSAT models
2. portfolios of solvers for NP-hard graph problems
3. development of efficient MinSAT solvers

Thank you for your attention!