

# On Efficient Computation of Variable MUSes

Anton Belov<sup>1</sup>, Alexander Ivrii<sup>2</sup>,  
Arie Matsliah<sup>2</sup>, Joao Marques-Silva<sup>1</sup>

<sup>1</sup>Complex and Adaptive Systems Laboratory  
University College Dublin, Ireland

<sup>2</sup>IBM Research – Haifa, Israel

SAT 2012  
June 20, 2012  
Trento, Italy

# Introduction

$$\mathcal{F} = \{C_1, \dots, C_6\} \in \text{UNSAT}$$

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

# Introduction

$$\mathcal{F} = \{C_1, \dots, C_6\} \in \text{UNSAT}$$

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

Which part of  $\mathcal{F}$  is responsible for its inconsistency ?

# Introduction

$\mathcal{F} = \{C_1, \dots, C_6\} \in \text{UNSAT}$

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

Which part of  $\mathcal{F}$  is responsible for its inconsistency ?

# Introduction

$\mathcal{F} = \{C_1, \dots, C_6\} \in \text{UNSAT}$

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

Which part of  $\mathcal{F}$  is responsible for its inconsistency ?

$\{C_1, C_2, C_3\}$  is a subset-minimal set of clauses required to refute  $\mathcal{F}$ .

# Introduction

$\mathcal{F} = \{C_1, \dots, C_6\} \in \text{UNSAT}$

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

Which part of  $\mathcal{F}$  is responsible for its inconsistency ?

$\{C_1, C_2, C_3\}$  is a subset-minimal set of clauses required to refute  $\mathcal{F}$ .

$\mathcal{F}' \subseteq \mathcal{F}$  is *minimally unsatisfiable subformula (MUS)* of  $\mathcal{F}$  if  $\mathcal{F}' \in \text{UNSAT}$ , and  $\forall C \in \mathcal{F}', \mathcal{F}' \setminus \{C\} \in \text{SAT}$ .

# Introduction

$\mathcal{F} = \{C_1, \dots, C_6\} \in \text{UNSAT}$

$$C_1 = (p)$$

$$C_2 = (q)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_4 = (p \vee q)$$

$$C_5 = (\neg p \vee r)$$

$$C_6 = (\neg q \vee \neg r)$$

Which part of  $\mathcal{F}$  is responsible for its inconsistency ?

$\{C_1, C_2, C_3\}$  is a subset-minimal set of clauses required to refute  $\mathcal{F}$ .

$\mathcal{F}' \subseteq \mathcal{F}$  is *minimally unsatisfiable subformula (MUS)* of  $\mathcal{F}$  if  $\mathcal{F}' \in \text{UNSAT}$ , and  $\forall C \in \mathcal{F}', \mathcal{F}' \setminus \{C\} \in \text{SAT}$ .

$\{C_1, C_2, C_5, C_6\}$  is also an MUS of  $\mathcal{F}$ .

# Introduction

$\mathcal{F} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \mathcal{G}_3 \in \text{UNSAT}$  — partitioned into *groups* (sets) of clauses

$$\mathcal{G}_1 = \{C_1, C_2\}, \quad \mathcal{G}_2 = \{C_3, C_4\}, \quad \mathcal{G}_3 = \{C_5, C_6\}.$$

$$C_1 = (p)$$

$$C_2 = (q)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_4 = (p \vee q)$$

$$C_5 = (\neg p \vee r)$$

$$C_6 = (\neg q \vee \neg r)$$



# Introduction

$\mathcal{F} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \mathcal{G}_3 \in \text{UNSAT}$  — partitioned into *groups* (sets) of clauses

$$\mathcal{G}_1 = \{C_1, C_2\}, \quad \mathcal{G}_2 = \{C_3, C_4\}, \quad \mathcal{G}_3 = \{C_5, C_6\}.$$

$$C_1 = (p)$$

$$C_2 = (q)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_4 = (p \vee q)$$

$$C_5 = (\neg p \vee r)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of *groups* required to refute  $\mathcal{F}$  ?

# Introduction

$\mathcal{F} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \mathcal{G}_3 \in \text{UNSAT}$  — partitioned into *groups* (sets) of clauses

$$\mathcal{G}_1 = \{C_1, C_2\}, \quad \mathcal{G}_2 = \{C_3, C_4\}, \quad \mathcal{G}_3 = \{C_5, C_6\}.$$

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of *groups* required to refute  $\mathcal{F}$  ?

$\{\mathcal{G}_1, \mathcal{G}_2\}$  is a *group-MUS* of (the partitioned)  $\mathcal{F}$ .

# Introduction

$\mathcal{F} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \mathcal{G}_3 \in \text{UNSAT}$  — partitioned into *groups* (sets) of clauses

$$\mathcal{G}_1 = \{C_1, C_2\}, \quad \mathcal{G}_2 = \{C_3, C_4\}, \quad \mathcal{G}_3 = \{C_5, C_6\}.$$

$$C_1 = (p)$$

$$C_2 = (q)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_4 = (p \vee q)$$

$$C_5 = (\neg p \vee r)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of *groups* required to refute  $\mathcal{F}$  ?

$\{\mathcal{G}_1, \mathcal{G}_2\}$  is a *group-MUS* of (the partitioned)  $\mathcal{F}$ .

$\{\mathcal{G}_1, \mathcal{G}_3\}$  is also a *group-MUS* of (the partitioned)  $\mathcal{F}$ .

# Introduction

What about the variables of  $\mathcal{F}$  ?

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of variables required to refute  $\mathcal{F}$  ?

# Introduction

What about the variables of  $\mathcal{F}$  ?

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of variables required to refute  $\mathcal{F}$  ?

$\{p, q\}$  is such a subset of variables.

# Introduction

What about the variables of  $\mathcal{F}$  ?

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of variables required to refute  $\mathcal{F}$  ?

$\{p, q\}$  is such a subset of variables.

$\{p, q\}$  is a *variable-MUS* of  $\mathcal{F}$ .

# Introduction

What about the variables of  $\mathcal{F}$  ?

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of variables required to refute  $\mathcal{F}$  ?

$\{p, q\}$  is such a subset of variables.

$\{p, q\}$  is a *variable-MUS* of  $\mathcal{F}$ .

This paper is about algorithms for efficient computation of variable-MUSes.

# Introduction

What about the variables of  $\mathcal{F}$  ?

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

What is a subset-minimal set of variables required to refute  $\mathcal{F}$  ?

$\{p, q\}$  is such a subset of variables.

$\{p, q\}$  is a *variable-MUS* of  $\mathcal{F}$ .

This paper is about algorithms for efficient computation of variable-MUSes.

Some applications: finding vertex-critical subgraphs; abstraction in abstraction refinement framework; satisfying assignments minimization.



## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  induced by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

- ▶ The subformula induced by  $\{p, q\}$  is  $\mathcal{F}|_{\{p,q\}} = \{C_1, C_2, C_3, C_4\}$ .
  - ▶ variable  $r$  is “removed” from  $\mathcal{F}$ .

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  induced by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

- ▶ The subformula induced by  $\{p, q\}$  is  $\mathcal{F}|_{\{p,q\}} = \{C_1, C_2, C_3, C_4\}$ .
  - ▶ variable  $r$  is “removed” from  $\mathcal{F}$ .
- ▶ The subformula induced by  $\{p\}$ ,  $\mathcal{F}|_{\{p\}} = \{C_1\}$ .
  - ▶ variables  $q, r$  are “removed” from  $\mathcal{F}$ .

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

## Definition

A set  $V \subseteq \text{Var}(\mathcal{F})$  is a *variable-MUS (VMUS)* of  $\mathcal{F}$  if  $\mathcal{F}|_V \in \text{UNSAT}$ , and for any  $V' \subset V$ ,  $\mathcal{F}|_{V'} \in \text{SAT}$ .

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

## Definition

A set  $V \subseteq \text{Var}(\mathcal{F})$  is a *variable-MUS (VMUS)* of  $\mathcal{F}$  if  $\mathcal{F}|_V \in \text{UNSAT}$ , and for any  $V' \subset V$ ,  $\mathcal{F}|_{V'} \in \text{SAT}$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

## Definition

A set  $V \subseteq \text{Var}(\mathcal{F})$  is a *variable-MUS (VMUS)* of  $\mathcal{F}$  if  $\mathcal{F}|_V \in \text{UNSAT}$ , and for any  $V' \subset V$ ,  $\mathcal{F}|_{V'} \in \text{SAT}$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

- ▶  $\mathcal{F}|_{\{p,q\}} = \{C_1, C_2, C_3, C_4\} \in \text{UNSAT}$ .

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

## Definition

A set  $V \subseteq \text{Var}(\mathcal{F})$  is a *variable-MUS (VMUS)* of  $\mathcal{F}$  if  $\mathcal{F}|_V \in \text{UNSAT}$ , and for any  $V' \subset V$ ,  $\mathcal{F}|_{V'} \in \text{SAT}$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

►  $\mathcal{F}|_{\{p\}} = \{C_1\} \in \text{SAT}$ .



## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

## Definition

A set  $V \subseteq \text{Var}(\mathcal{F})$  is a *variable-MUS (VMUS)* of  $\mathcal{F}$  if  $\mathcal{F}|_V \in \text{UNSAT}$ , and for any  $V' \subset V$ ,  $\mathcal{F}|_{V'} \in \text{SAT}$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

- ▶  $\mathcal{F}|_{\{q\}} = \{C_2\} \in \text{SAT}$ .

## Definition

Let  $V \subseteq \text{Var}(\mathcal{F})$ . The subformula of  $\mathcal{F}$  *induced* by  $V$  is the formula  $\mathcal{F}|_V = \{C \mid C \in \mathcal{F} \text{ and } \text{Var}(C) \subseteq V\}$ .

I.e.  $\mathcal{F}|_V$  includes only those clauses of  $\mathcal{F}$  whose variables are in  $V$ .

## Definition

A set  $V \subseteq \text{Var}(\mathcal{F})$  is a *variable-MUS (VMUS)* of  $\mathcal{F}$  if  $\mathcal{F}|_V \in \text{UNSAT}$ , and for any  $V' \subset V$ ,  $\mathcal{F}|_{V'} \in \text{SAT}$ .

$$C_1 = (p)$$

$$C_3 = (\neg p \vee \neg q)$$

$$C_5 = (\neg p \vee r)$$

$$C_2 = (q)$$

$$C_4 = (p \vee q)$$

$$C_6 = (\neg q \vee \neg r)$$

- ▶ Hence,  $\{p, q\}$  is a VMUS of  $\mathcal{F}$ . Notation:  $\{p, q\} \in \text{VMUS}(\mathcal{F})$ .

# Computing VMUSes

Basic algorithms are similar to MUS extraction algorithms: based on detection of *necessary* variables.

Notation: for  $v \in \text{Var}(\mathcal{F})$ ,  $\mathcal{F}^v = \{C \mid C \in \mathcal{F} \text{ and } v \in \text{Var}(C)\}$ .

## Definition

$v \in \text{Var}(\mathcal{F})$  is *necessary* for  $\mathcal{F}$  if  $\mathcal{F} \in \text{UNSAT}$  and  $\mathcal{F} \setminus \mathcal{F}^v \in \text{SAT}$ .

# Computing VMUSes

Basic algorithms are similar to MUS extraction algorithms: based on detection of *necessary* variables.

Notation: for  $v \in \text{Var}(\mathcal{F})$ ,  $\mathcal{F}^v = \{C \mid C \in \mathcal{F} \text{ and } v \in \text{Var}(C)\}$ .

## Definition

$v \in \text{Var}(\mathcal{F})$  is *necessary* for  $\mathcal{F}$  if  $\mathcal{F} \in \text{UNSAT}$  and  $\mathcal{F} \setminus \mathcal{F}^v \in \text{SAT}$ .

Properties:

1.  $V \in \text{VMUS}(\mathcal{F})$  if and only if every  $v \in V$  is necessary for  $\mathcal{F}|_V$ .
2. If  $v$  is necessary for  $\mathcal{F}$ , then  $v$  is necessary for any unsatisfiable  $\mathcal{F}' \subseteq \mathcal{F}$ .

# Hybrid VMUS Computation w/o optimizations

**Input**  $\mapsto$  **Output**: Unsatisfiable CNF Formula  $\mathcal{F} \mapsto V \in \text{VMUS}(\mathcal{F})$

```
 $V \leftarrow \emptyset$  // VMUS under-approximation
 $V_w \leftarrow \text{Var}(\mathcal{F})$  // Working ('untested') set of variables
 $\mathcal{F}_w \leftarrow \mathcal{F}$  // Working formula
while  $V_w \neq \emptyset$  do // Inv:  $\mathcal{F}_w = \mathcal{F}|_{V \cup V_w}$  and  $\forall v \in V$  is nec. for  $\mathcal{F}_w$ 
   $v \leftarrow \text{PickVariable}(V_w)$ 
   $V_w \leftarrow V_w \setminus \{v\}$ 
   $\mathcal{R} \leftarrow \text{CNF}(\neg \mathcal{F}_w^v)$  // Redundancy removal
   $\text{st} = \text{SAT}(\mathcal{F}_w \setminus \mathcal{F}_w^v)$ 
  if  $\text{st} = \text{false}$  then //  $v$  is not necessary for  $\mathcal{F}_w$ 
    // Variable-set refinement
     $V_w \leftarrow V_w \cap \text{Var}(\mathcal{U})$ 
     $\mathcal{F}_w \leftarrow \mathcal{F}_w \setminus \mathcal{F}_w^v$ 
  else //  $v$  is necessary for  $\mathcal{F}_w$ 
     $V \leftarrow V \cup \{v\}$ 
     $V_w \leftarrow V_w \setminus V$ 
return  $V$  //  $V \in \text{VMUS}(\mathcal{F})$  and  $\mathcal{F}_w = \mathcal{F}|_V \in \text{VMU}$ 
```

# Hybrid VMUS Computation w/o optimizations

Similar to the Removal algorithm [C. Desrosiers, et al, J. Comb. Optim. 2008]

Does not scale:

- ▶ Number of SAT solver calls:  $|Var(\mathcal{F})|$ .
- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.
  - ▶ W/o optimizations solved: 157, avg.vars: 6198, max.vars: 49490.

# Hybrid VMUS Computation w/o optimizations

Similar to the Removal algorithm [C. Desrosiers, et al, J. Comb. Optim. 2008]

Does not scale:

- ▶ Number of SAT solver calls:  $|Var(\mathcal{F})|$ .
- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.
  - ▶ W/o optimizations solved: 157, avg.vars: 6198, max.vars: 49490.
  - ▶ With optimizations solved: 265, avg.vars: 105044; max.vars: 1198007.

# Hybrid VMUS Computation w/o optimizations

Similar to the Removal algorithm [C. Desrosiers, et al, J. Comb. Optim. 2008]

Does not scale:

- ▶ Number of SAT solver calls:  $|Var(\mathcal{F})|$ .
- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.
  - ▶ W/o optimizations solved: 157, avg.vars: 6198, max.vars: 49490.
  - ▶ With optimizations solved: 265, avg.vars: 105044; max.vars: 1198007.
- ▶ Bottleneck: SAT solver calls.



# Hybrid VMUS Computation w/o optimizations

Similar to the Removal algorithm [C. Desrosiers, et al, J. Comb. Optim. 2008]

Does not scale:

- ▶ Number of SAT solver calls:  $|Var(\mathcal{F})|$ .
- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.
  - ▶ W/o optimizations solved: 157, avg.vars: 6198, max.vars: 49490.
  - ▶ With optimizations solved: 265, avg.vars: 105044; max.vars: 1198007.
- ▶ Bottleneck: SAT solver calls.

Optimizations are aimed at:

- ▶ Making fewer SAT solver calls.
- ▶ Making SAT solver calls easier.

- ▶ **Fact:** Let  $\mathcal{U}$  be an unsatisfiable core of  $\mathcal{F}$ . Then,  $\text{Var}(\mathcal{U})$  contains at least one VMUS of  $\mathcal{F}$ .

# Optimizations: variable-set refinement

- ▶ **Fact:** Let  $\mathcal{U}$  be an unsatisfiable core of  $\mathcal{F}$ . Then,  $\text{Var}(\mathcal{U})$  contains at least one VMUS of  $\mathcal{F}$ .
- ▶ Hence, if  $\mathcal{U}$  is an unsatisfiable core of  $\mathcal{F}$ , all variables outside of  $\text{Var}(\mathcal{U})$  can be removed from  $\mathcal{F}$  — *variable-set refinement*.

# Optimizations: variable-set refinement

- ▶ **Fact:** Let  $\mathcal{U}$  be an unsatisfiable core of  $\mathcal{F}$ . Then,  $\text{Var}(\mathcal{U})$  contains at least one VMUS of  $\mathcal{F}$ .
- ▶ Hence, if  $\mathcal{U}$  is an unsatisfiable core of  $\mathcal{F}$ , all variables outside of  $\text{Var}(\mathcal{U})$  can be removed from  $\mathcal{F}$  — *variable-set refinement*.
- ▶ Relies on the capability of SAT solvers to return unsatisfiable core.

# Optimizations: variable-set refinement

- ▶ **Fact:** Let  $\mathcal{U}$  be an unsatisfiable core of  $\mathcal{F}$ . Then,  $\text{Var}(\mathcal{U})$  contains at least one VMUS of  $\mathcal{F}$ .
- ▶ Hence, if  $\mathcal{U}$  is an unsatisfiable core of  $\mathcal{F}$ , all variables outside of  $\text{Var}(\mathcal{U})$  can be removed from  $\mathcal{F}$  — *variable-set refinement*.
- ▶ Relies on the capability of SAT solvers to return unsatisfiable core.
- ▶ Applied to the working set of variables  $V_w$  inside the main loop,  $\mathcal{F}_w$  is updated accordingly.

# Optimizations: variable-set refinement

- ▶ **Fact:** Let  $\mathcal{U}$  be an unsatisfiable core of  $\mathcal{F}$ . Then,  $\text{Var}(\mathcal{U})$  contains at least one VMUS of  $\mathcal{F}$ .
- ▶ Hence, if  $\mathcal{U}$  is an unsatisfiable core of  $\mathcal{F}$ , all variables outside of  $\text{Var}(\mathcal{U})$  can be removed from  $\mathcal{F}$  — *variable-set refinement*.
- ▶ Relies on the capability of SAT solvers to return unsatisfiable core.
- ▶ Applied to the working set of variables  $V_w$  inside the main loop,  $\mathcal{F}_w$  is updated accordingly.
- ▶ Effect: remove multiple unnecessary variables in one SAT solver call.

# Hybrid VMUS Computation: variable-set refinement

**Input**  $\mapsto$  **Output**: Unsatisfiable CNF Formula  $\mathcal{F} \mapsto V \in \text{VMUS}(\mathcal{F})$

```
 $V \leftarrow \emptyset$  // VMUS under-approximation
 $V_w \leftarrow \text{Var}(\mathcal{F})$  // Working ('untested') set of variables
 $\mathcal{F}_w \leftarrow \mathcal{F}$  // Working formula
while  $V_w \neq \emptyset$  do // Inv:  $\mathcal{F}_w = \mathcal{F}|_{V \cup V_w}$  and  $\forall v \in V$  is nec. for  $\mathcal{F}_w$ 
   $v \leftarrow \text{PickVariable}(V_w)$ 
   $V_w \leftarrow V_w \setminus \{v\}$ 
   $\mathcal{R} \leftarrow \text{CNF}(\neg \mathcal{F}_w^v)$  // Redundancy removal
   $\text{st} = \text{SAT}(\mathcal{F}_w \setminus \mathcal{F}_w^v)$ 
  if  $\text{st} = \text{false}$  then //  $v$  is not necessary for  $\mathcal{F}_w$ 
    // Variable-set refinement
     $V_w \leftarrow V_w \cap \text{Var}(\mathcal{U})$ 
     $\mathcal{F}_w \leftarrow \mathcal{F}_w \setminus \mathcal{F}_w^v$ 
  else //  $v$  is necessary for  $\mathcal{F}_w$ 
     $V \leftarrow V \cup \{v\}$ 
     $V_w \leftarrow V_w \setminus V$ 
return  $V$  //  $V \in \text{VMUS}(\mathcal{F})$  and  $\mathcal{F}_w = \mathcal{F}|_V \in \text{VMU}$ 
```

# Hybrid VMUS Computation: variable-set refinement

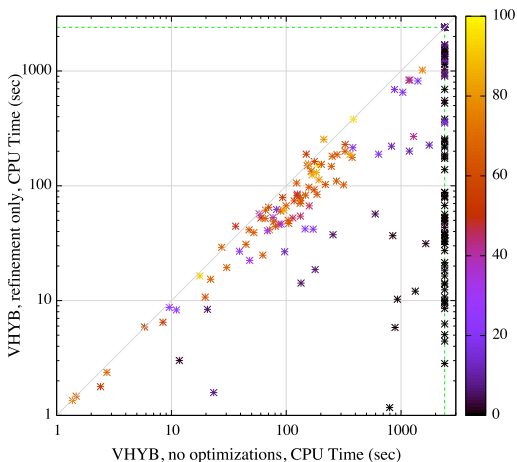
**Input**  $\mapsto$  **Output**: Unsatisfiable CNF Formula  $\mathcal{F} \mapsto V \in \text{VMUS}(\mathcal{F})$

```
 $V \leftarrow \emptyset$  // VMUS under-approximation
 $V_w \leftarrow \text{Var}(\mathcal{F})$  // Working (''untested'') set of variables
 $\mathcal{F}_w \leftarrow \mathcal{F}$  // Working formula
while  $V_w \neq \emptyset$  do // Inv:  $\mathcal{F}_w = \mathcal{F}|_{V \cup V_w}$  and  $\forall v \in V$  is nec. for  $\mathcal{F}_w$ 
   $v \leftarrow \text{PickVariable}(V_w)$ 
   $V_w \leftarrow V_w \setminus \{v\}$ 
   $\mathcal{R} \leftarrow \text{CNF}(\neg \mathcal{F}_w^v)$  // Redundancy removal
   $\langle \text{st}, \mathcal{U} \rangle = \text{SAT}(\mathcal{F}_w \setminus \mathcal{F}_w^v)$ 
  if  $\text{st} = \text{false}$  then //  $v$  is not necessary for  $\mathcal{F}_w$ 
    // Variable-set refinement
     $V_w \leftarrow V_w \cap \text{Var}(\mathcal{U})$ 
     $\mathcal{F}_w \leftarrow \mathcal{F}_w|_{V \cup V_w}$ 
  else //  $v$  is necessary for  $\mathcal{F}_w$ 
     $V \leftarrow V \cup \{v\}$ 
     $V_w \leftarrow V_w \setminus V$ 
return  $V$  //  $V \in \text{VMUS}(\mathcal{F})$  and  $\mathcal{F}_w = \mathcal{F}|_V \in \text{VMU}$ 
```



# Impact of variable-set refinement

- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.



- ▶ CPU Time, w/o optimizations (#sol=157) vs refinement (#sol=239)
- ▶ Color: VMUS size (% of the number of variables in the input).

## Optimizations: variable-based model rotation (VMR)

- ▶ **Fact:** Take  $\mathcal{F} \in \text{UNSAT}$  and an assignment  $\tau$  to  $\text{Var}(\mathcal{F})$ . Then, any variable shared among the clauses falsified by  $\tau$  is necessary for  $\mathcal{F}$ .

# Optimizations: variable-based model rotation (VMR)

- ▶ **Fact:** Take  $\mathcal{F} \in \text{UNSAT}$  and an assignment  $\tau$  to  $\text{Var}(\mathcal{F})$ . Then, any variable shared among the clauses falsified by  $\tau$  is necessary for  $\mathcal{F}$ .
- ▶ Why ?
  - ▶ Let  $\mathcal{F}'$  be the clauses of  $\mathcal{F}$  falsified by  $\tau$ .
  - ▶ Let  $v$  be any variable shared among the clauses of  $\mathcal{F}'$ 
    - ▶ i.e.  $\forall C \in \mathcal{F}', v \in \text{Var}(C)$ .
  - ▶ Remove  $v$  from  $\mathcal{F}$ .
  - ▶ All clauses of  $\mathcal{F}'$  will be gone, because they all contain  $v$ .
  - ▶ We get a subformula of  $\mathcal{F} \setminus \mathcal{F}'$ .
  - ▶ But,  $\mathcal{F} \setminus \mathcal{F}' \in \text{SAT}$ , since  $\tau$  is its model.
  - ▶ Hence,  $\mathcal{F} \setminus \mathcal{F}' \in \text{SAT}$ , and so  $v$  is necessary for  $\mathcal{F}$ .

# Optimizations: variable-based model rotation (VMR)

- ▶ **Fact:** Take  $\mathcal{F} \in \text{UNSAT}$  and an assignment  $\tau$  to  $\text{Var}(\mathcal{F})$ . Then, any variable shared among the clauses falsified by  $\tau$  is necessary for  $\mathcal{F}$ .
- ▶ Why ?
  - ▶ Let  $\mathcal{F}'$  be the clauses of  $\mathcal{F}$  falsified by  $\tau$ .
  - ▶ Let  $v$  be any variable shared among the clauses of  $\mathcal{F}'$ 
    - ▶ i.e.  $\forall C \in \mathcal{F}', v \in \text{Var}(C)$ .
  - ▶ Remove  $v$  from  $\mathcal{F}$ .
  - ▶ All clauses of  $\mathcal{F}'$  will be gone, because they all contain  $v$ .
  - ▶ We get a subformula of  $\mathcal{F} \setminus \mathcal{F}'$ .
  - ▶ But,  $\mathcal{F} \setminus \mathcal{F}' \in \text{SAT}$ , since  $\tau$  is its model.
  - ▶ Hence,  $\mathcal{F} \setminus \mathcal{F}' \in \text{SAT}$ , and so  $v$  is necessary for  $\mathcal{F}$ .
- ▶ So, any assignment  $\tau$ , such that  $v \in \bigcap_{C \in \text{Unsat}(\mathcal{F}, \tau)} \text{Var}(C)$ , is a *witness of necessity* of  $v$  in  $\mathcal{F}$ .

# Optimizations: variable-based model rotation (VMR)

- ▶ Note: in VHYB, when  $\mathcal{F}_w \setminus \mathcal{F}_w^v \in \text{SAT}$ , the assignment returned by the SAT solver (call it  $\tau$ ) is a witness of  $v$ .

# Optimizations: variable-based model rotation (VMR)

- ▶ Note: in VHYB, when  $\mathcal{F}_w \setminus \mathcal{F}_w^v \in \text{SAT}$ , the assignment returned by the SAT solver (call it  $\tau$ ) is a witness of  $v$ .
- ▶  $\tau$  might be a witness for another variable too. E.g. when  $\text{Unsat}(\mathcal{F}_w, \tau) = \{C\}$ , every variable in  $C$  is necessary for  $\mathcal{F}_w$ .

# Optimizations: variable-based model rotation (VMR)

- ▶ Note: in VHYB, when  $\mathcal{F}_w \setminus \mathcal{F}_w^v \in \text{SAT}$ , the assignment returned by the SAT solver (call it  $\tau$ ) is a witness of  $v$ .
- ▶  $\tau$  might be a witness for another variable too. E.g. when  $\text{Unsat}(\mathcal{F}_w, \tau) = \{C\}$ , every variable in  $C$  is necessary for  $\mathcal{F}_w$ .
- ▶ **Variable-based model rotation**: take  $\tau$  and try to modify it into a witness  $\tau'$  for another variable. How?
  1. Flip  $v$  to get new assignment  $\tau'$ : all clauses in  $\text{Unsat}(\mathcal{F}_w, \tau)$  are now satisfied.
  2. But, some other clauses *must* be falsified by  $\tau'$ .
  3. Any variable shared among the clauses falsified by  $\tau'$  is necessary.
  4. Continue recursively (but watch for loops).

# Optimizations: variable-based model rotation (VMR)

- ▶ Note: in VHYB, when  $\mathcal{F}_w \setminus \mathcal{F}_w^v \in \text{SAT}$ , the assignment returned by the SAT solver (call it  $\tau$ ) is a witness of  $v$ .
- ▶  $\tau$  might be a witness for another variable too. E.g. when  $\text{Unsat}(\mathcal{F}_w, \tau) = \{C\}$ , every variable in  $C$  is necessary for  $\mathcal{F}_w$ .
- ▶ **Variable-based model rotation**: take  $\tau$  and try to modify it into a witness  $\tau'$  for another variable. How?
  1. Flip  $v$  to get new assignment  $\tau'$ : all clauses in  $\text{Unsat}(\mathcal{F}_w, \tau)$  are now satisfied.
  2. But, some other clauses *must* be falsified by  $\tau'$ .
  3. Any variable shared among the clauses falsified by  $\tau'$  is necessary.
  4. Continue recursively (but watch for loops).
- ▶ Effect: detect multiple necessary variables in a single SAT solver call.



# Optimizations: variable-based model rotation (VMR)

- ▶ Note: in VHYB, when  $\mathcal{F}_w \setminus \mathcal{F}_w^v \in \text{SAT}$ , the assignment returned by the SAT solver (call it  $\tau$ ) is a witness of  $v$ .
- ▶  $\tau$  might be a witness for another variable too. E.g. when  $\text{Unsat}(\mathcal{F}_w, \tau) = \{C\}$ , every variable in  $C$  is necessary for  $\mathcal{F}_w$ .
- ▶ **Variable-based model rotation**: take  $\tau$  and try to modify it into a witness  $\tau'$  for another variable. How?
  1. Flip  $v$  to get new assignment  $\tau'$ : all clauses in  $\text{Unsat}(\mathcal{F}_w, \tau)$  are now satisfied.
  2. But, some other clauses *must* be falsified by  $\tau'$ .
  3. Any variable shared among the clauses falsified by  $\tau'$  is necessary.
  4. Continue recursively (but watch for loops).
- ▶ Effect: detect multiple necessary variables in a single SAT solver call.
- ▶ Details and an “extended” (improved) version of VMR are in the paper.

# Hybrid VMUS Computation: VMR

**Input**  $\mapsto$  **Output**: Unsatisfiable CNF Formula  $\mathcal{F} \mapsto V \in \text{VMUS}(\mathcal{F})$

```
 $V \leftarrow \emptyset$  // VMUS under-approximation
 $V_w \leftarrow \text{Var}(\mathcal{F})$  // Working (''untested'') set of variables
 $\mathcal{F}_w \leftarrow \mathcal{F}$  // Working formula
while  $V_w \neq \emptyset$  do // Inv:  $\mathcal{F}_w = \mathcal{F}|_{V \cup V_w}$  and  $\forall v \in V$  is nec. for  $\mathcal{F}_w$ 
   $v \leftarrow \text{PickVariable}(V_w)$ 
   $V_w \leftarrow V_w \setminus \{v\}$ 
   $\mathcal{R} \leftarrow \text{CNF}(\neg \mathcal{F}_w^v)$  // Redundancy removal
   $\langle \text{st}, \mathcal{U} \rangle = \text{SAT}(\mathcal{F}_w \setminus \mathcal{F}_w^v)$ 
  if  $\text{st} = \text{false}$  then //  $v$  is not necessary for  $\mathcal{F}_w$ 
    // Variable-set refinement
     $V_w \leftarrow V_w \cap \text{Var}(\mathcal{U})$ 
     $\mathcal{F}_w \leftarrow \mathcal{F}_w|_{V \cup V_w}$ 
  else //  $v$  is necessary for  $\mathcal{F}_w$ 
     $V \leftarrow V \cup \{v\}$ 
     $V_w \leftarrow V_w \setminus V$ 
return  $V$  //  $V \in \text{VMUS}(\mathcal{F})$  and  $\mathcal{F}_w = \mathcal{F}|_V \in \text{VMU}$ 
```

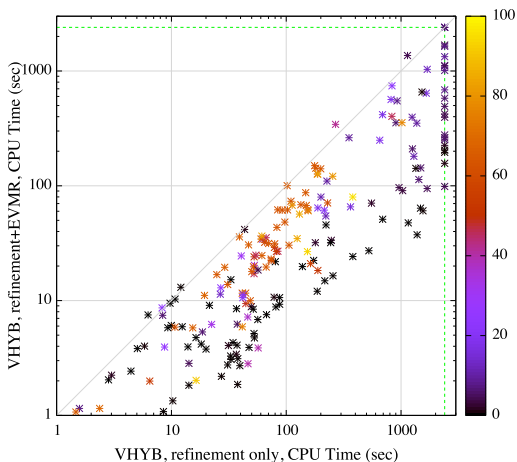
# Hybrid VMUS Computation: VMR

**Input**  $\mapsto$  **Output**: Unsatisfiable CNF Formula  $\mathcal{F} \mapsto V \in \text{VMUS}(\mathcal{F})$

```
 $V \leftarrow \emptyset$  // VMUS under-approximation
 $V_w \leftarrow \text{Var}(\mathcal{F})$  // Working (“untested”) set of variables
 $\mathcal{F}_w \leftarrow \mathcal{F}$  // Working formula
while  $V_w \neq \emptyset$  do // Inv:  $\mathcal{F}_w = \mathcal{F}|_{V \cup V_w}$  and  $\forall v \in V$  is nec. for  $\mathcal{F}_w$ 
   $v \leftarrow \text{PickVariable}(V_w)$ 
   $V_w \leftarrow V_w \setminus \{v\}$ 
   $\mathcal{R} \leftarrow \text{CNF}(\neg \mathcal{F}_w^v)$  // Redundancy removal
   $\langle \text{st}, \mathcal{U}, \tau \rangle = \text{SAT}(\mathcal{F}_w \setminus \mathcal{F}_w^v)$ 
  if  $\text{st} = \text{false}$  then //  $v$  is not necessary for  $\mathcal{F}_w$ 
    // Variable-set refinement
     $V_w \leftarrow V_w \cap \text{Var}(\mathcal{U})$ 
     $\mathcal{F}_w \leftarrow \mathcal{F}_w|_{V \cup V_w}$ 
  else //  $v$  is necessary for  $\mathcal{F}_w$ 
    //  $v \in V$  after this call
     $\text{VMModelRotation}(\mathcal{F}_w, V, \tau)$ 
     $V_w \leftarrow V_w \setminus V$ 
return  $V$  //  $V \in \text{VMUS}(\mathcal{F})$  and  $\mathcal{F}_w = \mathcal{F}|_V \in \text{VMU}$ 
```

# Impact of variable-based model rotation

- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.



- ▶ CPU Time, refinement (#sol=239) vs ref.+EVMR (#sol=264)
- ▶ Color: VMUS size (% of the number of variables in the input).

# Optimizations: redundancy removal

- ▶ **Fact:** If  $\mathcal{F} \in \text{UNSAT}$ , then for any  $v \in \text{Var}(\mathcal{F})$ ,  $\mathcal{F} \setminus \mathcal{F}^v \in \text{SAT}$  if and only if  $(\mathcal{F} \setminus \mathcal{F}^v) \cup \text{CNF}(\neg\mathcal{F}^v) \in \text{SAT}$ .
  - ▶  $\neg\text{CNF}(\mathcal{F}^v)$  stands for a satisfiability-preserving transformation of the formula  $\neg\mathcal{F}^v$  (we used Plaisted-Greenbaum).
  - ▶ During hybrid VMUS extraction: add the clauses of  $\text{CNF}(\neg\mathcal{F}^v)$  to the formula before SAT solver call.

# Optimizations: redundancy removal

- ▶ **Fact:** If  $\mathcal{F} \in \text{UNSAT}$ , then for any  $v \in \text{Var}(\mathcal{F})$ ,  $\mathcal{F} \setminus \mathcal{F}^v \in \text{SAT}$  if and only if  $(\mathcal{F} \setminus \mathcal{F}^v) \cup \text{CNF}(\neg\mathcal{F}^v) \in \text{SAT}$ .
  - ▶  $\neg\text{CNF}(\mathcal{F}^v)$  stands for a satisfiability-preserving transformation of the formula  $\neg\mathcal{F}^v$  (we used Plaisted-Greenbaum).
  - ▶ During hybrid VMUS extraction: add the clauses of  $\text{CNF}(\neg\mathcal{F}^v)$  to the formula before SAT solver call.
- ▶ Effect: make SAT calls easier.

# Optimizations: redundancy removal

- ▶ **Fact:** If  $\mathcal{F} \in \text{UNSAT}$ , then for any  $v \in \text{Var}(\mathcal{F})$ ,  $\mathcal{F} \setminus \mathcal{F}^v \in \text{SAT}$  if and only if  $(\mathcal{F} \setminus \mathcal{F}^v) \cup \text{CNF}(\neg\mathcal{F}^v) \in \text{SAT}$ .
  - ▶  $\neg\text{CNF}(\mathcal{F}^v)$  stands for a satisfiability-preserving transformation of the formula  $\neg\mathcal{F}^v$  (we used Plaisted-Greenbaum).
  - ▶ During hybrid VMUS extraction: add the clauses of  $\text{CNF}(\neg\mathcal{F}^v)$  to the formula before SAT solver call.
- ▶ Effect: make SAT calls easier.
- ▶ But: if the clauses of  $\text{CNF}(\neg\mathcal{F}^v)$  are included in the unsatisfiable core (in case of UNSAT outcome), the core cannot be used safely for the variable-set refinement.

# Hybrid VMUS Extraction: redundancy removal

**Input**  $\mapsto$  **Output**: Unsatisfiable CNF Formula  $\mathcal{F} \mapsto V \in \text{VMUS}(\mathcal{F})$

```
 $V \leftarrow \emptyset$  // VMUS under-approximation
 $V_w \leftarrow \text{Var}(\mathcal{F})$  // Working (“untested”) set of variables
 $\mathcal{F}_w \leftarrow \mathcal{F}$  // Working formula
while  $V_w \neq \emptyset$  do // Inv:  $\mathcal{F}_w = \mathcal{F}|_{V \cup V_w}$  and  $\forall v \in V$  is nec. for  $\mathcal{F}_w$ 
   $v \leftarrow \text{PickVariable}(V_w)$ 
   $V_w \leftarrow V_w \setminus \{v\}$ 
   $\mathcal{R} \leftarrow \text{CNF}(\neg \mathcal{F}_w^v)$  // Redundancy removal
   $\langle \text{st}, \mathcal{U}, \tau \rangle = \text{SAT}(\mathcal{F}_w \setminus \mathcal{F}_w^v)$ 
  if  $\text{st} = \text{false}$  then //  $v$  is not necessary for  $\mathcal{F}_w$ 
    // Variable-set refinement
     $V_w \leftarrow V_w \cap \text{Var}(\mathcal{U})$ 
     $\mathcal{F}_w \leftarrow \mathcal{F}_w|_{V \cup V_w}$ 
  else //  $v$  is necessary for  $\mathcal{F}_w$ 
    //  $v \in V$  after this call
     $\text{VMModelRotation}(\mathcal{F}_w, V, \tau)$ 
     $V_w \leftarrow V_w \setminus V$ 
return  $V$  //  $V \in \text{VMUS}(\mathcal{F})$  and  $\mathcal{F}_w = \mathcal{F}|_V \in \text{VMU}$ 
```



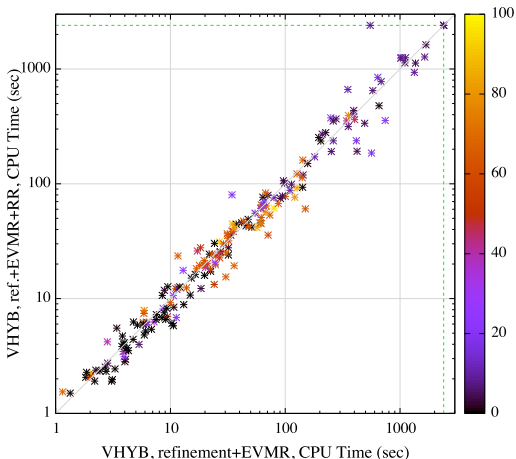
# Hybrid VMUS Extraction: redundancy removal

**Input**  $\mapsto$  **Output**: Unsatisfiable CNF Formula  $\mathcal{F} \mapsto V \in \text{VMUS}(\mathcal{F})$

```
 $V \leftarrow \emptyset$  // VMUS under-approximation
 $V_w \leftarrow \text{Var}(\mathcal{F})$  // Working (“untested”) set of variables
 $\mathcal{F}_w \leftarrow \mathcal{F}$  // Working formula
while  $V_w \neq \emptyset$  do // Inv:  $\mathcal{F}_w = \mathcal{F}|_{V \cup V_w}$  and  $\forall v \in V$  is nec. for  $\mathcal{F}_w$ 
   $v \leftarrow \text{PickVariable}(V_w)$ 
   $V_w \leftarrow V_w \setminus \{v\}$ 
   $\mathcal{R} \leftarrow \text{CNF}(\neg \mathcal{F}_w^v)$  // Redundancy removal
   $\langle \text{st}, \mathcal{U}, \tau \rangle = \text{SAT}((\mathcal{F}_w \setminus \mathcal{F}_w^v) \cup \mathcal{R})$ 
  if  $\text{st} = \text{false}$  then //  $v$  is not necessary for  $\mathcal{F}_w$ 
    if  $\mathcal{U} \cap \mathcal{R} = \emptyset$  then  $V_w \leftarrow V_w \cap \text{Var}(\mathcal{U})$  // “Safe” refinement
     $\mathcal{F}_w \leftarrow \mathcal{F}_w|_{V \cup V_w}$ 
  else //  $v$  is necessary for  $\mathcal{F}_w$ 
     $\text{VModelRotation}(\mathcal{F}_w, V, \tau)$  //  $v \in V$  after this call
     $V_w \leftarrow V_w \setminus V$ 
return  $V$  //  $V \in \text{VMUS}(\mathcal{F})$  and  $\mathcal{F}_w = \mathcal{F}|_V \in \text{VMU}$ 
```

# Impact of redundancy removal

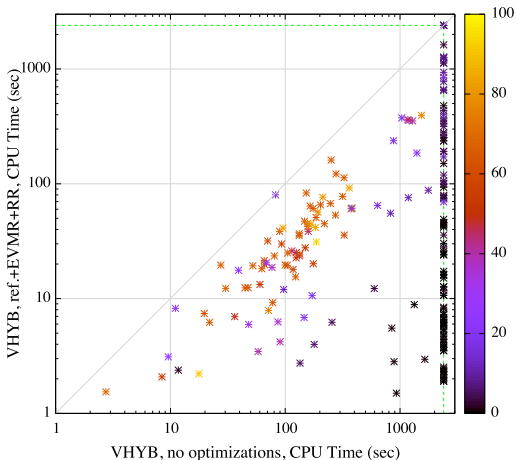
- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.



- ▶ CPU Time, ref.+EVMR vs ref.+EVMR+RR. Average speedup: 15%.
- ▶ Color: VMUS size (% of the number of variables in the input).

# Impact of all optimizations

- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.



- ▶ W/o optimizations solved: 157, avg.vars: 6198, max.vars: 49490.
- ▶ With optimizations solved: 265, avg.vars: 105044; max.vars: 1198007.

## Additional approaches (in the paper)

Relaxation-variable based approach.

- ▶ Lets the SAT solver to *find* a necessary variable.
- ▶ Probably won't scale (have not tried it).

## Additional approaches (in the paper)

Relaxation-variable based approach.

- ▶ Lets the SAT solver to *find* a necessary variable.
- ▶ Probably won't scale (have not tried it).

Translation to group-MUS computation problem.

- ▶ For dense formulas: for each  $v \in \text{Var}(\mathcal{F})$  introduce two variables  $v_p$  and  $v_n$ , and
  1. replace  $v$  with  $v_p$  and  $\neg v$  with  $v_n$ ;
  2. put the result into group 0;
  3. make a group that says  $v_p \leftrightarrow v_n$ .

# Additional approaches (in the paper)

Relaxation-variable based approach.

- ▶ Lets the SAT solver to *find* a necessary variable.
- ▶ Probably won't scale (have not tried it).

Translation to group-MUS computation problem.

- ▶ For dense formulas: for each  $v \in \text{Var}(\mathcal{F})$  introduce two variables  $v_p$  and  $v_n$ , and
  1. replace  $v$  with  $v_p$  and  $\neg v$  with  $v_n$ ;
  2. put the result into group 0;
  3. make a group that says  $v_p \leftrightarrow v_n$ .
- ▶ For sparse formulas: for each  $v \in \text{Var}(\mathcal{F})$  introduce an activation variable  $a_v$ , and
  1. add an activation variable  $a_v$  to clause  $C$  if  $v \in C$ ;
  2. put all translated clauses to group 0;
  3. for each variable make a group  $\{a_v\}$ .

## Additional approaches (in the paper)

Relaxation-variable based approach.

- ▶ Lets the SAT solver to *find* a necessary variable.
- ▶ Probably won't scale (have not tried it).

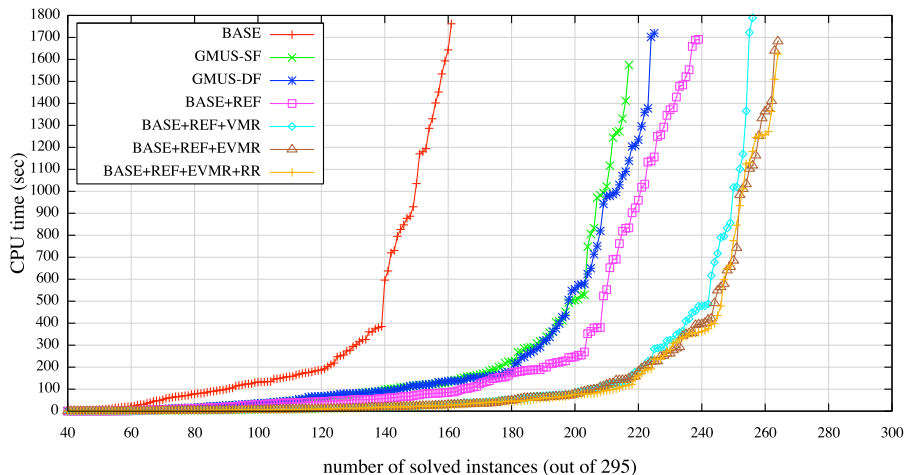
Translation to group-MUS computation problem.

- ▶ For dense formulas: for each  $v \in \text{Var}(\mathcal{F})$  introduce two variables  $v_p$  and  $v_n$ , and
  1. replace  $v$  with  $v_p$  and  $\neg v$  with  $v_n$ ;
  2. put the result into group 0;
  3. make a group that says  $v_p \leftrightarrow v_n$ .
- ▶ For sparse formulas: for each  $v \in \text{Var}(\mathcal{F})$  introduce an activation variable  $a_v$ , and
  1. add an activation variable  $a_v$  to clause  $C$  if  $v \in C$ ;
  2. put all translated clauses to group 0;
  3. for each variable make a group  $\{a_v\}$ .

*Outperformed by Hybrid VMUS algorithm by a wide margin (see paper)*

# Performance comparison: run-time

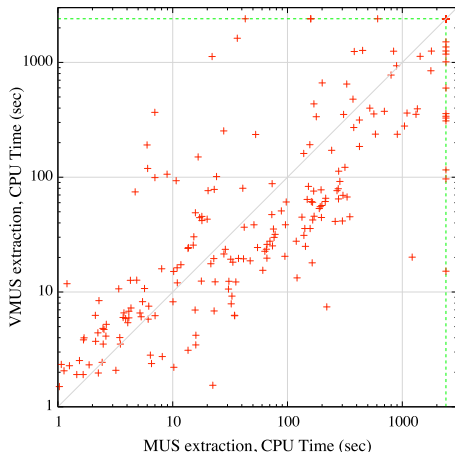
- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.





# VMUS extraction vs MUS extraction

- ▶ 295 benchmarks from SAT Comp 2011, TO = 1800 sec, MO = 4GB.



- ▶ CPU Time, MUS (#sol=245) vs VMUS extraction (#sol=265).

# Summary

- ▶ Hybrid VMUS Extraction — significant performance gains vs state-of-the-art due to optimizations.

# Summary

- ▶ Hybrid VMUS Extraction — significant performance gains vs state-of-the-art due to optimizations.
- ▶ Implemented in MUSer2 (binary distro), at <http://logos.ucd.ie/wiki/doku.php?id=muser>

# Summary

- ▶ Hybrid VMUS Extraction — significant performance gains vs state-of-the-art due to optimizations.
- ▶ Implemented in MUSer2 (binary distro), at <http://logos.ucd.ie/wiki/doku.php?id=muser>
- ▶ A number of alternative approaches (not as efficient).

# Summary

- ▶ Hybrid VMUS Extraction — significant performance gains vs state-of-the-art due to optimizations.
- ▶ Implemented in MUSer2 (binary distro), at <http://logos.ucd.ie/wiki/doku.php?id=muser>
- ▶ A number of alternative approaches (not as efficient).
- ▶ VMUS computation is in many cases faster than MUS computation  
⇒ perhaps some applications of MUSes could be re-considered.

# Summary

- ▶ Hybrid VMUS Extraction — significant performance gains vs state-of-the-art due to optimizations.
- ▶ Implemented in MUSEr2 (binary distro), at <http://logos.ucd.ie/wiki/doku.php?id=muser>
- ▶ A number of alternative approaches (not as efficient).
- ▶ VMUS computation is in many cases faster than MUS computation  
⇒ perhaps some applications of MUSes could be re-considered.

Thank you for your attention !