

# SAT and SMT Solving

---

Sarah Winkler

SS 2018

Department of Computer Science  
University of Innsbruck

## Outline

- Summary of Last Week
- Unsatisfiable Cores
- Algorithm by Fu and Malik
- Unsatisfiable Cores in Practice

1

## Summary of Last Week

---

## Maximum Satisfiability

Consider CNF formulas  $\chi$  and  $\varphi$  as sets of clauses.

### Definitions

- ▶  $\text{maxSAT}(\varphi)$  is maximal  $|\psi|$  such that  $\psi \subseteq \varphi$  and  $\bigwedge_{C \in \psi} C$  satisfiable
- ▶  $\text{pmaxSAT}(\varphi, \chi)$  is maximal  $|\psi|$  such that  $\psi \subseteq \varphi$  and  $\chi \wedge \bigwedge_{C \in \psi} C$  satisfiable

### Definitions

given weights  $w_C \in \mathbb{Z}$  for all  $C \in \varphi$ ,

- ▶  $\text{maxSAT}_w(\varphi)$  is maximal  $\sum_{C \in \psi} w_C$  such that  $\psi \subseteq \varphi$  and  $\bigwedge_{C \in \psi} C$  satisfiable
- ▶  $\text{pmaxSAT}_w(\varphi, \chi)$  is maximal  $\sum_{C \in \psi} w_C$  such that  $\psi \subseteq \varphi$  and  $\chi \wedge \bigwedge_{C \in \psi} C$  satisfiable

### Definition

$\text{minUNSAT}(\varphi)$  is minimal  $|\psi|$  such that  $\psi \subseteq \varphi$  and  $\bigwedge_{C \in \psi} \neg C$  is satisfiable

### Lemma

$$|\varphi| = |\text{minUNSAT}(\varphi)| + |\text{maxSAT}(\varphi)|$$

2

## Branch & Bound

### Idea

- ▶ gets list of clauses  $\varphi$  as input return  $\text{minUNSAT}(\varphi)$
- ▶ explores assignments in depth-first search

### Notation

- ▶  $\varphi_x$  is formula  $\varphi$  with all occurrences of  $x$  replaced by T
- ▶  $\varphi_{\bar{x}}$  is formula  $\varphi$  with all occurrences of  $x$  F
- ▶ for list of clauses  $\varphi$ , function  $\text{simp}(\varphi)$ 
  - ▶ replace  $\neg T$  by F and  $\neg F$  by T
  - ▶ drops all clauses which contain T
  - ▶ removes F from all remaining clauses
- ▶  $\square$  denotes empty clause and  $\#\text{empty}(\varphi)$  number of empty clauses in  $\varphi$

3

## Algorithm (Branch & Bound)

```
function BnB( $\varphi$ , UB)
   $\varphi = \text{simp}(\varphi)$ 
  if  $\varphi$  contains only empty clauses then
    return  $\#\text{empty}(\varphi)$ 
  if  $\#\text{empty}(\varphi) \geq \text{UB}$  then
    return UB
   $x = \text{selectVariable}(\varphi)$ 
   $\text{UB} := \min(\text{UB}, \text{BnB}(\varphi_x, \text{UB}))$ 
  return  $\min(\text{UB}, \text{BnB}(\varphi_{\bar{x}}, \text{UB}))$ 
```

### Theorem

$\text{BnB}(\varphi, |\varphi|) = \text{minUNSAT}(\varphi)$

4

## Binary Search

### Idea

- ▶ gets list of clauses  $\varphi$  as input and returns  $\text{minUNSAT}(\varphi)$
- ▶ repeatedly call SAT solver in binary search fashion

### Definitions

- ▶ cardinality constraint is

$$\sum_{x \in X} x \bowtie N$$

where  $\bowtie$  is  $=$ ,  $<$ ,  $>$ ,  $\leq$ , or  $\geq$ ,  $X$  is set of propositional variables, and  $N \in \mathbb{N}$

- ▶ valuation  $v$  satisfies  $\sum_{x \in X} x \bowtie N$  iff  $k \bowtie N$   
where  $k$  is number of variables  $x \in X$  such that  $v(x) = T$

### Remark

cardinality constraints are expressible in CNF

5

## Algorithm (Binary Search)

```
function BinarySearch( $\{C_1, \dots, C_m\}$ )
   $\varphi := \{C_1 \vee b_1, \dots, C_m \vee b_m\}$ 
  return search( $\varphi, 0, m$ )
```

$b_1, \dots, b_m$  are fresh variables

```
function search( $\varphi, L, U$ )
  if  $L \geq U$  then
    return U
   $\text{mid} := \lfloor \frac{U+L}{2} \rfloor$ 
  if SAT( $\varphi \wedge \text{CNF}(\sum_{i=1}^m b_i \leq \text{mid})$ ) then
    return search( $\varphi, L, \text{mid}$ )
  else
    return search( $\varphi, \text{mid} + 1, U$ )
```

### Theorem

$\text{BinarySearch}(\varphi) = \text{minUNSAT}(\varphi)$

6

# Unsatisfiable Cores

## Definitions

for unsatisfiable CNF formula  $\varphi$  given as set of clauses

- ▶  $\psi \subseteq \varphi$  such that  $\bigwedge_{C \in \psi} C$  is unsatisfiable is **unsatisfiable core (UC)** of  $\varphi$
- ▶ **minimal unsatisfiable core**  $\psi$  is UC such that every subset of  $\psi$  is satisfiable
- ▶ **MUC** (minimum unsatisfiable core) is UC such that  $|\psi|$  is minimal

## Example

$$\varphi = \{\neg x, \quad x \vee z, \quad \neg y \vee \neg z, \quad x, \quad y \vee \neg z\}$$

unsatisfiable cores are

- ▶  $\varphi$
- ▶  $\{\neg x, x \vee z, \neg y \vee \neg z, y \vee \neg z\}$
- ▶  $\{\neg x, x\}$

minimal  
minimal and MUC

## Remark

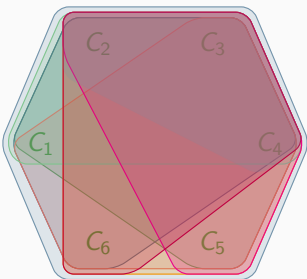
MUC is always minimal unsatisfiable core

## Example

$$\varphi = \{C_1, \dots, C_6\}$$

$$\begin{array}{lll} C_1: x_1 \vee \neg x_3 & C_2: x_2 & C_3: \neg x_2 \vee x_3 \\ C_4: \neg x_2 \vee \neg x_3 & C_5: x_2 \vee x_3 & C_6: \neg x_1 \vee x_2 \vee \neg x_3 \end{array}$$

$\varphi$  has 9 unsatisfiable cores:



- $UC_1 = \{C_1, C_2, C_3, C_4, C_5, C_6\}$
- $UC_2 = \{C_1, C_2, C_3, C_4, C_5\}$
- $UC_3 = \{C_1, C_2, C_3, C_4, C_6\}$
- $UC_4 = \{C_1, C_3, C_4, C_5, C_6\}$
- $UC_5 = \{C_2, C_3, C_4, C_5, C_6\}$
- $UC_6 = \{C_1, C_2, C_3, C_4\}$
- $UC_7 = \{C_2, C_3, C_4, C_5\}$
- $UC_8 = \{C_2, C_3, C_4, C_6\}$
- $UC_9 = \{C_2, C_3, C_4\}$

minimal and MUC

## Application: FPGA Routing

### Field Programmable Gate Arrays (FPGAs)

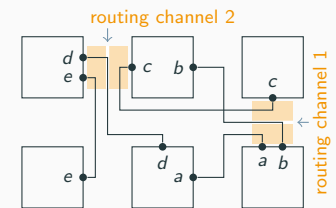
- ▶ can simulate microprocessors but faster for special tasks (from complex combinatorics to mere logic)
- ▶ logic blocks connected by "routing channels"
- ▶ "routing": determine which channels are used for what



### Example (Encoding Routing Requirements)

- ▶ consider connections  $a, b, c, d, e$  of 2 bits each
- ▶ **liveness**: want to route  $\geq 1$  bit of  $a, b, c, d, e$
- ▶ 2 routing channels of 2 tracks each
- ▶ **exclusivity**: each channel has only 2 tracks
- ▶ **unsatisfiable**: UCs indicate problems

- |                |      |                          |    |                          |    |
|----------------|------|--------------------------|----|--------------------------|----|
| $a_0 \vee a_1$ | ●●●  | $\neg a_0 \vee \neg b_0$ | ●● | $\neg c_0 \vee \neg d_0$ | ●● |
| $b_0 \vee b_1$ | ●●●  | $\neg a_0 \vee \neg c_0$ | ●● | $\neg c_0 \vee \neg e_0$ | ●● |
| $c_0 \vee c_1$ | ●●●● | $\neg b_0 \vee \neg c_0$ | ●● | $\neg d_0 \vee \neg e_0$ | ●● |
| $d_0 \vee d_1$ | ●●●  | $\neg a_1 \vee \neg b_1$ | ●● | $\neg c_1 \vee \neg d_1$ | ●● |
| $e_0 \vee e_1$ | ●●●  | $\neg a_1 \vee \neg c_1$ | ●● | $\neg c_1 \vee \neg e_1$ | ●● |
|                |      | $\neg b_1 \vee \neg c_1$ | ●● | $\neg d_1 \vee \neg e_1$ | ●● |



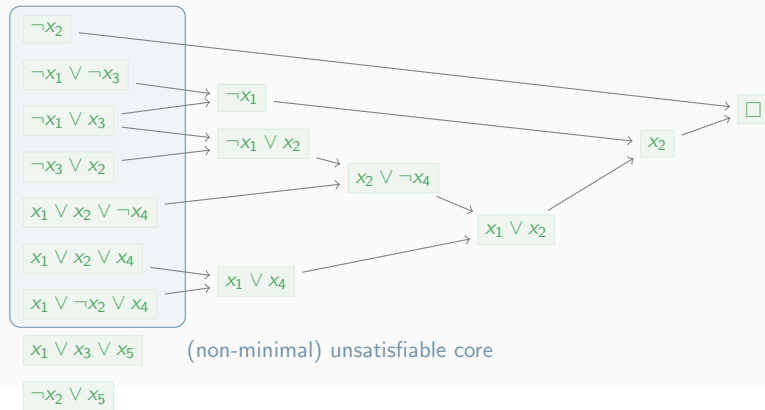
- $UC_1$ : channel 1 capacity exceeded
- $UC_2$ : channel 2 capacity exceeded
- $UC_3$ : c is overconstrained
- $UC_4$ : c is overconstrained

# Finding Minimal Unsatisfiable Cores by Resolution

## Idea

- repeatedly pick clause  $C$  from  $\varphi$  and check satisfiability: if  $\varphi \setminus \{C\}$  is satisfiable, keep  $C$ , otherwise drop  $C$
- SAT solvers can give **resolution proof** if conflict detected: use resolution graphs for more efficient implementation of this idea

## Example (Resolution Graph)



10

## Definition (Resolution Graph)

directed acyclic graph  $G = (V, E)$  is resolution graph for set of clauses  $\varphi$  if

- $V = V_i \uplus V_c$  is set of clauses and  $V_i = \varphi$ , initial nodes
- $V_i$  nodes have no incoming edges,
- there is exactly one node  $\square$  without outgoing edges,
- $\forall C \in V_c \exists$  edges  $D \rightarrow C, D' \rightarrow C$  such that  $C$  is resolvent of  $D$  and  $D'$ , and
- there are no other edges.

## Notation

- $Reach_G(C)$  is set of nodes reachable from  $C$  in  $G$
- $Reach_G^E(C)$  is set of edges reachable from  $C$  in  $G$
- $BReach(C)$  is set of nodes backwards reachable from  $C$  in  $G$
- $\overline{N}$  is  $V \setminus N$  for any set of nodes  $N$

11

## Algorithm minUnsatCore( $\varphi$ )

**Input:** unsatisfiable formula  $\varphi$

**Output:** minimal unsatisfiable core of  $\varphi$

build refutation graph  $G = (V_i \uplus V_c, E)$  for  $\varphi$

**while**  $\exists$  unmarked clause in  $V_i$  **do**

$C \leftarrow$  unmarked clause in  $V_i$

**if**  $SAT(Reach_G(C))$  **then**

$\triangleright$  subgraph without  $C$  satisfiable?

        mark  $C$

$\triangleright C$  is UC member

**else**

        build refutation graph  $G' = (V'_i \uplus V'_c, E')$  for  $\overline{Reach_G(C)}$

$V'_i \leftarrow V_i \setminus \{C\}$  and  $V'_c \leftarrow V'_c \cup (V_c \setminus Reach_G^E(C))$

$E \leftarrow E' \cup (E \setminus Reach_G^E(C))$

$G \leftarrow (V_i \cup V_c, E)$

$G \leftarrow G|_{BReach_G(\square)}$

$\triangleright$  restrict to nodes with path to  $\square$

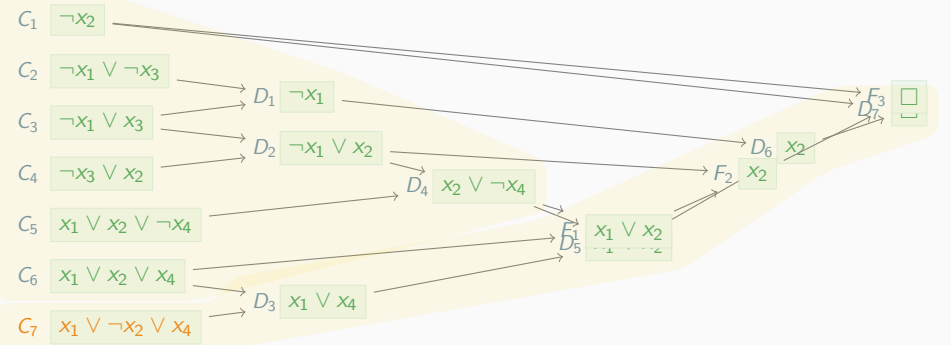
**return**  $V_i$

## Theorem

if  $\varphi$  unsatisfiable then  $minUnsatCore(\varphi)$  is minimal unsatisfiable core of  $\varphi$

12

## Example



## minUnsatCore( $\varphi$ )

- pick  $C_7$
- $Reach_G(C_7) = \{C_7, D_3, D_5, D_6, D_7\}$ , so  $\overline{Reach_G(C_7)} = \{C_1, \dots, C_6, D_1, D_2, D_4\}$
- check  $SAT(\overline{Reach_G(C_7)})$
- unsatisfiable: get new resolution graph  $G_7$  for  $\varphi \cup \{D_1, D_2, D_4\}$
- construct resolution graph  $G'$  for  $\varphi$  by adding edges from  $G$  to  $G_7$
- set  $G$  to  $G'$  restricted to  $BReach_{G'}(\square)$
- after 5 more loop iterations: return  $\{C_1, C_3, \dots, C_6\}$

re-use relevant resolvents:  
fewer steps to  $\square$

13

## Bounds for Maximum Satisfiability

### Definition

if  $\varphi = C_1 \wedge \dots \wedge C_m$  is CNF formula then **blocked formula**

$$\varphi_B = (C_1 \vee b_1) \wedge \dots \wedge (C_m \vee b_m)$$

for fresh variables  $b_1, \dots, b_m$

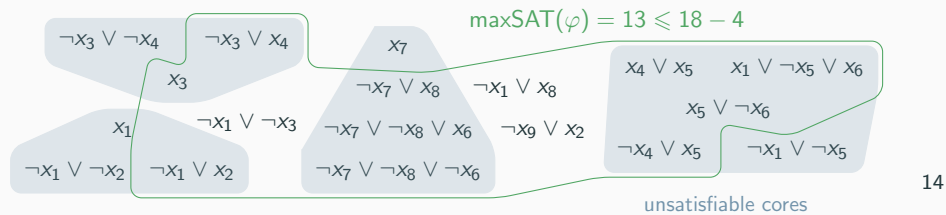
### Lemma (Lower Bound)

Let  $v$  satisfy  $\varphi_B$  and  $B_T = \{b_i \mid v(b_i) = T\}$ . Then  $\text{maxSAT}(\varphi) \geq |\varphi| - |B_T|$

### Lemma (Upper Bound)

If  $\varphi$  contains  $k$  disjoint unsatisfiable cores then  $\text{maxSAT}(\varphi) \leq |\varphi| - k$

### Example (Upper Bound)



## Algorithm by Fu and Malik

## Algorithm by Fu and Malik

### Definition (Partial minUNSAT)

$\text{pminUNSAT}(\chi, \varphi)$  is minimal  $|\psi|$  such that  $\psi \subseteq \varphi$  and  $\chi \wedge \bigwedge_{C \in \psi} \neg C$  satisfiable

### Lemma

$$|\varphi| = |\text{pminUNSAT}(\chi, \varphi)| + |\text{pmaxSAT}(\chi, \varphi)|$$

### Idea

- ▶ maximal satisfying valuation falsifies at least one clause in unsatisfiable core
- ▶ repeatedly call SAT solver on relaxed formula excluding unsatisfiable core until resulting formula is satisfiable
- ▶ returns  $\text{pminUNSAT}$  for hard clauses  $\chi$ , soft clauses  $\varphi$

### Algorithm FuMalik( $\chi, \varphi$ )

**Input:** clause set  $\varphi$  and satisfiable clause set  $\chi$   
**Output:**  $\text{minUNSAT}(\chi, \varphi)$

$cost \leftarrow 0$

**while**  $\neg \text{SAT}(\chi \cup \varphi)$  **do**

$UC \leftarrow \text{unsatCore}(\chi \cup \varphi)$

$B \leftarrow \emptyset$

**for**  $C \in UC \cap \varphi$  **do**

        ▷ loop over soft clauses in core

$b \leftarrow$  new blocking variable

$\varphi \leftarrow \varphi \setminus \{C\} \cup \{C \vee b\}$

$B \leftarrow B \cup \{b\}$

$\chi \leftarrow \chi \cup \text{CNF}(\sum_{b \in B} b = 1)$

        ▷ cardinality constraint is hard

$cost \leftarrow cost + 1$

**return**  $cost$

### Theorem

$$\text{FuMalik}(\chi, \varphi) = \text{pminUNSAT}(\chi, \varphi)$$

## Example

$\chi$ :  $\neg x_1 \vee x_3$        $\neg x_7 \vee x_2$        $x_7 \vee x_2$        $x_1 \vee \neg x_2$   
 $\varphi$ :  $\neg x_1 \vee \neg x_2 \vee b_1$     $\neg x_1 \vee x_2 \vee b_2$     $\neg x_1 \vee x_7$        $x_1 \vee b_3$   
 $\neg x_3 \vee x_4 \vee c_1$     $x_3 \vee c_2$        $\neg x_3 \vee \neg x_4 \vee c_3$     $x_4 \vee x_5$   
 $\neg x_4 \vee x_5$        $x_1 \vee \neg x_5 \vee x_6$     $x_5 \vee \neg x_6$        $x_7 \vee d_1$   
 $\neg x_7 \vee x_8 \vee d_2$     $\neg x_7 \vee \neg x_8 \vee x_6 \vee d_3$     $\neg x_7 \vee \neg x_8 \vee \neg x_6 \vee d_4$     $\neg x_1 \vee \neg x_3 \vee e_1$

- ▶ unsatisfiable core:  $\neg x_1 \vee \neg x_2, \neg x_1 \vee x_2, x_1$   
 $\chi = \chi \cup \text{CNF}(b_1 + b_2 + b_3 = 1)$   
 $\text{cost} = 1$
- ▶ unsatisfiable core:  $\neg x_3 \vee x_4, x_3, \neg x_3 \vee \neg x_4$   
 $\chi = \chi \cup \text{CNF}(c_1 + c_2 + c_3 = 1)$   
 $\text{cost} = 2$
- ▶ unsatisfiable core:  $x_7, \neg x_7 \vee x_8, \neg x_7 \vee \neg x_8 \vee x_6, \neg x_7 \vee \neg x_8 \vee \neg x_6$   
 $\chi = \chi \cup \text{CNF}(d_1 + d_2 + d_3 + d_4 = 1)$   
 $\text{cost} = 3$
- ▶ unsatisfiable core:  $\neg x_1 \vee x_3, \neg x_7 \vee x_2, x_7 \vee x_2, x_1 \vee \neg x_2, \neg x_1 \vee \neg x_3$   
 $\chi = \chi \cup \text{CNF}(e_1 = 1)$   
 $\text{cost} = 4$
- ▶ satisfiable:  $v(x_1) = v(x_2) = v(x_3) = v(x_5) = v(x_7) = \text{T}$  and  $v(x_i) = \text{F}$  otherwise
- ▶  $\text{pminUNSAT}(\chi, \varphi) = 4$  and  $\text{pmaxSAT}(\chi, \varphi) = 12$

17

## Unsatisfiable Cores in Practice

## Unsatisfiable Cores in z3

```
from z3 import *

x1,x2,x3 = Bool("x1"), Bool("x2"), Bool("x3")
phi = [ Or(Not(x1), Not(x2)), Or(Not(x1), x2),\
         Or(Not(x1), x3), x1, Or(Not(x3), x2)]




solver = Solver()
solver.set(unsat_core=True)

# assert clauses in phi with names phi0 ... phi4
for i,c in enumerate(phi):
    solver.assert_and_track(c, "phi" + str(i))

if solver.check() == z3.unsat:
    uc = solver.unsat_core()
    print(uc) # [phi0, phi1, phi3]
```

18

## Literature

-  Nachum Dershowitz, Ziyad Hanna, and Alexander Nadel.  
**A Scalable Algorithm for Minimal Unsatisfiable Core Extraction.**  
Proc. Theory and Applications of Satisfiability Testing, pp. 36–41, 2006.
-  Yoonna Oh, Maher Mneimneh, Zaher Andraus, Karem Sakallah, and Igor Markov  
**AMUSE: A Minimally-Unsatisfiable Subformula Extractor.**  
Proc. 41st Design Automation Conference, pp. 518–523, 2004.
-  Zhaohui Fu and Sharad Malik.  
**On solving the partial MAX-SAT problem.**  
In Proc. Theory and Applications of Satisfiability Testing, pp. 252–265, 2006

19