Algorithms for Computational Logic

Overconstrained Problems

Emmanuel Hebrard (adapted from) João Marques Silva

Outline

LAAS CNRS

Outline

LAAS CNRS

Maximum satisfiability

$x_6 \vee x_2$	$\neg x_6 \vee x_2$	$\neg x_2 \vee x_1$	$\neg x_1$
$\neg x_6 \vee x_8$	$x_6 \vee \neg x_8$	$x_2 \vee x_4$	$\neg x_4 \vee x_5$
$x_7 \vee x_5$	$\neg x_7 \vee x_5$	$\neg x_5 \vee x_3$	$\neg x_3$

- Unsatisfiable formula
- Find largest subset of clauses that is satisfiable: the complement of a *minimum-size correction set*
- For above example, MaxSAT solution is 2:
  - By removing 2 clauses, the remaining are satisfiable

MaxSAT problem(s)

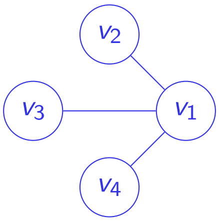
		Hard Clauses?	
		No	Yes
Weights?	No	Plain	Partial
	Yes	Weighted	Weighted Partial

- Must satisfy hard clauses, if any
- Compute set of satisfied soft clauses with maximum cost
  - Without weights, cost of each falsified soft clause is 1
- **Or**, compute set of falsified soft clauses with minimum cost (s.t. hard & remaining soft clauses are satisfied)
- **Note**: goal is to compute **set** of satisfied (or falsified) clauses; **not** just the cost **!**

Outline

Minimum vertex cover

- The problem:
  - Graph $G = (V, E)$
  - Vertex cover $U \subseteq V$
    - For each $(v_i, v_j) \in E$, either $v_i \in U$ or $v_j \in U$
  - Minimum vertex cover: vertex cover U of minimum size

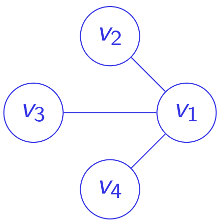


Vertex cover: $\{v_2, v_3, v_4\}$

Min vertex cover: $\{v_1\}$

Minimum vertex cover – MaxSAT formulation

- Partial MaxSAT formulation:
  - Variables: x_i for each $v_i \in V$, with $x_i = 1$ iff $v_i \in U$
  - Hard clauses: $(x_i \vee x_j)$ for each $(v_i, v_j) \in E$
  - Soft clauses: $(\neg x_i)$ for each $v_i \in V$
    - I.e. preferable not to include vertices in U



$$\begin{aligned}\mathcal{F}_H &= \{(x_1 \vee x_2), (x_1 \vee x_3), (x_1 \vee x_4)\}\\ \mathcal{F}_S &= \{(\neg x_1), (\neg x_2), (\neg x_3), (\neg x_4)\}\end{aligned}$$

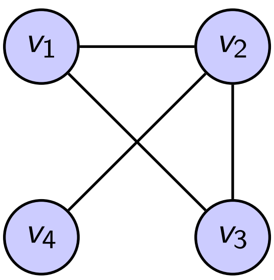
  - Hard clauses have cost ∞
  - Soft clauses have cost 1

Independent sets and cliques

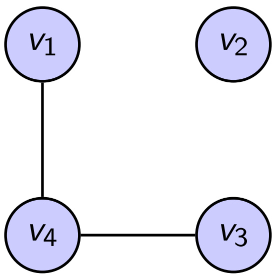
- Given undirected graph $G = (V, E)$:
  - A clique is a complete subgraph of G, i.e. it is a set $L \subseteq V$ such that $\forall_{u,v \in L} (u \neq v) \rightarrow (u, v) \in E$
  - A vertex cover $C \subseteq V$ is such that $\forall_{(u,v) \in E}\ u \in C \lor v \in C$
  - An independent set $I \subseteq V$ is such that $\forall_{u,v \in I}\ (v, u) \notin E$
- Properties:
  - If I is an independent set of $G = (V, E)$, then
    - ★ $V - I$ is a vertex cover of G
    - ★ I is a clique of the complement graph of G, G^C
  - A maximum independent set of G corresponds to a maximum clique of G^C

Independent sets and cliques – examples

- G:

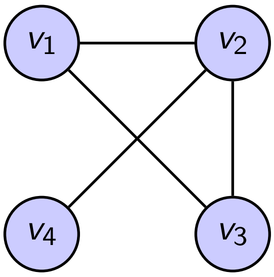


- G^C:



- $\{v_1, v_2, v_3\}$ is clique of G and an independent set of G^C
- $\{v_4\}$ is a vertex cover of G^C

Maximum clique with MaxSAT



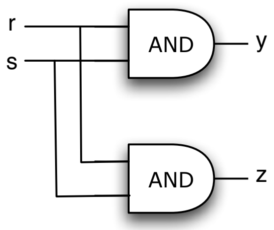
$$\mathcal{F}_H \triangleq (\neg x_1 \vee \neg x_4) \wedge (\neg x_3 \vee \neg x_4)$$
$$\mathcal{F}_S \triangleq \{(x_1), (x_2), (x_3), (x_4)\}$$

- MaxSAT formulation:
  - x_i: assigned 1 if $v_i \in V$ included in clique
  - If $\{x_i, x_j\} \notin E$, add hard clause $(\neg x_i \vee \neg x_j)$
  - Soft clauses (x_i) for $v_i \in V$
  - **Why?** Add as many vertices as possible to the clique such that non-adjacent vertices are not both selected

Design debugging

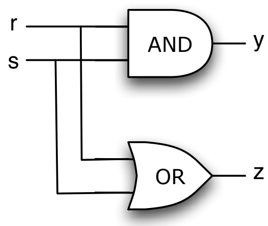
[SMVLS'07]

Correct circuit



Input stimuli: $\langle r, s\rangle = \langle 0, 1\rangle$
Valid output: $\langle y, z\rangle = \langle 0, 0\rangle$

Faulty circuit



Input stimuli: $\langle r, s\rangle = \langle 0, 1\rangle$
Invalid output: $\langle y, z\rangle = \langle 0, 0\rangle$

- The model:
  - Hard clauses: Input and output values
  - Soft clauses: CNF representation of circuit, each gate aggregated in group of clauses
- The problem:
  - Maximize number of satisfied clauses (i.e. circuit gates)

Software package upgrades with MaxSAT

[MBCV'06,TSJL'07,AL'08,ALMS'09,ALBL'10]

- Universe of software packages: $\{p_1, \ldots, p_n\}$
- Difference with respect to original installation: $\{p_1^\Delta, \ldots, p_n^\Delta\}$
- Incompatibilies, dependencies and non-regression
  - Hard clauses
- Objective: minimize $\sum_{i=1}^{n} p_i^\Delta$
  - Soft clauses $(p_1^\Delta) \land (p_2^\Delta) \land \ldots \land (p_i^\Delta)$

Many other applications

- Error localization in C code [JM'11]
- Haplotyping with pedigrees [GLMSO'10]
- Course timetabling [AN'10]
- Combinatorial auctions [HLGS'08]
- Minimizing Disclosure of Private Information in Credential-Based Interactions [AVFPS'10]
- Reasoning over Biological Networks [GL'12]
- Binate/unate covering
  - Haplotype inference [GMSLO'11]
  - Digital filter design [ACFM'08]
  - FSM synthesis [e.g. HS'96]
  - Logic minimization [e.g. HS'96]
  - ...
- ...

Problems with unit propagation

- Example formula:

$$\mathcal{F} \triangleq (x_1) \wedge (x_2) \wedge (x_3) \wedge (\neg x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_3)$$

- Unit propagation falsifies two clauses: $(\neg x_1 \vee \neg x_2)$ and $(\neg x_1 \vee \neg x_3)$
- But, the MaxSAT solution is 1; $\mathcal{S} \subseteq \mathcal{F}$ is satisfiable:

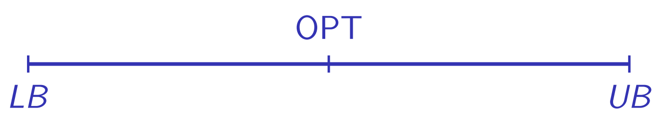
$$\mathcal{S} \triangleq (x_2) \wedge (x_3) \wedge (\neg x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_3)$$

- **Cannot** apply unit propagation when solving MaxSAT
- **Cannot** apply hallmarks of CDCL SAT solving
- MaxSAT solving requires dedicated algorithms

Outline

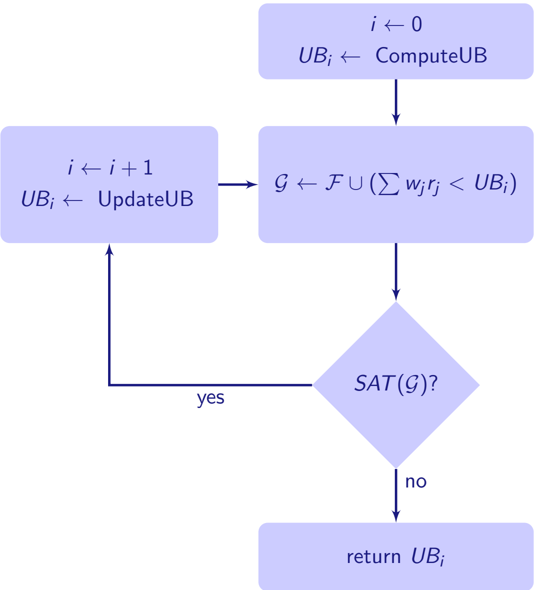
MaxSAT with iterative SAT solving – definitions

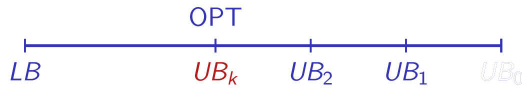
- Cost of assignment:
  - Sum of weights of falsified clauses



- Optimum solution (OPT):
  - Assignment with minimum cost
- Upper Bound (UB):
  - Assignment with cost $\geq$ OPT
  - E.g. $\sum_{c_j \in \mathcal{F}} w_j + 1$; hard clauses may be inconsistent
- Lower Bound (LB):
  - No assignment with cost $\leq$ LB
  - E.g. -1; it may be possible to satisfy all soft clauses
- Relax each soft clause c_j: $(c_j \vee r_j)$ (on-demand in core-guided)

MaxSAT with iterative SAT solving – refine UB





- Worst-case # of iterations **exponential** on instance size (# bits)
  - Improvement: use binary search instead
- Many example solvers: Minisat+, SAT4J, QMaxSat [ES06,LBP10,KZFH12]

MaxSAT with iterative SAT solving – complete example

$x_6 \vee x_2 \quad \vee r_1 \qquad \neg x_6 \vee x_2 \quad \vee r_2 \qquad \neg x_2 \vee x_1 \quad \vee r_3 \qquad \neg x$

$\neg x_6 \vee x_8 \quad \vee r_5 \qquad x_6 \vee \neg x_8 \quad \vee r_6 \qquad x_2 \vee x_4 \quad \vee r_7 \qquad \neg x_4$

$x_7 \vee x_5 \quad \vee r_9 \qquad \neg x_7 \vee x_5 \quad \vee r_{10} \qquad \neg x_5 \vee x_3 \quad \vee r_{11} \qquad \neg x$

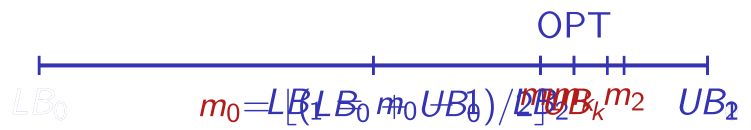
$\sum_{i=1}^{12} r_i \leq 12 \quad \sum_{i=1}^{12} r_i \leq 2 \quad \sum_{i=1}^{12} r_i \leq 1$

Example CNF formula Relax all clauses; Set $UB = 12 + 1$ Formula is SAT; E.g. all $x_i = 0$ and $r_1 = r_7 = r_9 = 1$ (i.e. cost $= 3$) Refine $UB = 3$ Formula is SAT; E.g. $x_1 = x_2 = 1$; $x_3 = \ldots =$ ... ost $= 2$) Refine $UB = 2$ ... ; terminate MaxSAT s... $B = 2$

AtMost$k$/PB constraints over **all** relaxation variables

All (possibly many) soft clauses relaxed

MaxSAT with iterative SAT solving – binary search



- Invariant: $LB_k \leq UB_k - 1$
- Require $\sum w_i\, r_i \leq m_0$
- Repeat
  - If UNSAT, refine $LB_1 = m_0, \ldots$
  - Compute new mid value $m_1, \ldots$
  - If SAT, refine $UB_3 = m_2, \ldots$
- Until $LB_k = UB_k - 1$
- Worst-case # of iterations **linear** on instance size

Branch&bound MaxSAT algorithm

Input: $\textit{max-sat}(\phi, UB)$: A CNF formula ϕ and an upper bound UB

$\phi \leftarrow \textit{simplifyFormula}(\phi)$;
if $\phi = \emptyset$ or ϕ only contains empty clauses **then**
  return $\#\textit{emptyClauses}(\phi)$;
end if
$LB \leftarrow \#\textit{emptyClauses}(\phi) + \textit{underestimation}(\phi, UB)$;
if $LB \geq UB$ **then**
  return UB;
end if
$x \leftarrow \textit{selectVariable}(\phi)$;
$UB \leftarrow \min(UB, \textit{max-sat}(\phi_{\bar{x}}, UB))$;
return $\min(UB, \textit{max-sat}(\phi_x, UB))$;

[LMP'07] **Output:** The minimal number of unsatisfied clauses of ϕ

- Many techniques for computing lower bounds, i.e. for lower bounding the search

LAAS CNRS

Outline

LAAS CNRS

Core-guided MaxSAT

$x_6 \vee x_2$	$\neg x_6 \vee x_2$	$\neg x_2 \vee x_1$	$\neg x_1$
$\neg x_6 \vee x_8$	$x_6 \vee \neg x_8$	$x_2 \vee x_4$	$\neg x_4 \vee x_5$
$x_7 \vee x_5$	$\neg x_7 \vee x_5$	$\neg x_5 \vee x_3$	$\neg x_3$

- **Goal**: Do **not** relax all clauses
  - **Why?**
    - ★ Some clauses never relevant for computing MaxSAT solution
    - ★ Simplify cardinality/PB constraints
- How to relax clauses **on demand**?
  - Relax clauses given computed unsatisfiable cores
    - ★ Many alternative ways to instrument code-guided algorithms

Fu&Malik's (FM) core-guided algorithm

$x_6 \vee x_2 \vee r_7$	$\neg x_6 \vee x_2 \vee r_8$	$\neg x_2 \vee x_1 \vee r_1 \vee r_9$	$\neg x_1 \vee r_2 \vee r_{10}$
$\neg x_6 \vee x_8$	$x_6 \vee \neg x_8$	$x_2 \vee x_4 \vee r_3$	$\neg x_4 \vee x_5 \vee r_4$
$x_7 \vee x_5 \vee r_{11}$	$\neg x_7 \vee x_5 \vee r_{12}$	$\neg x_5 \vee x_3 \vee r_5 \vee r_{13}$	$\neg x_3 \vee r_6 \vee r_{14}$
$\sum_{i=1}^{6} r_i \leq 1$	$\sum_{i=7}^{14} r_i \leq 1$		

Example CNF formula Formula is UNSAT; OPT $\leq |\varphi| - 1$; Get unsat core Add relaxation variables and AtMost1 constraint Formula is (again) UNSAT; OPT $\leq |\varphi| - 2$; Get unsat core Add new relax nd AtMost ance is n solution is $|\varphi| - \mathcal{I} = 12$

Only AtMost1 constraints used

Some clauses not relaxed

Relaxed soft clauses remain **soft**

Another example

$$\mathcal{F}_S \triangleq (x_1) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_3) \wedge (\neg x_3) \wedge (x_4 \vee \neg x_5) \wedge (\neg x_4 \vee x_5)$$

MSU3 core-guided algorithm

$x_6 \vee x_2 \vee r_7$ $\neg x_6 \vee x_2 \vee r_8$ $\neg x_2 \vee x_1 \vee r_1$ $\neg x_1 \vee r_2$

$\neg x_6 \vee x_8$ $x_6 \vee \neg x_8$ $x_2 \vee x_4 \vee r_3$ $\neg x_4 \vee x_5 \vee r_4$

$x_7 \vee x_5 \vee r_9$ $\neg x_7 \vee x_5 \vee r_{10}$ $\neg x_5 \vee x_3 \vee r_5$ $\neg x_3 \vee r_6$

$\sum_{i=1}^{6} r_i \leq 1$ $\sum_{i=1}^{10} r_i \leq 2$

Example CNF formula Formula is UNSAT; OPT $\leq |\varphi| - 1$; Get unsat core Add relaxation variables and AtMost1 constraint Formula is (again) UNSAT; OPT $\leq |\varphi| - 2$; Get unsat core Add new relax nd AtMo tance is solution is $|\varphi| - \mathcal{I} = 12$

AtMost$k$/PB constraints used

Some clauses not relaxed

Relaxed soft clauses become **hard**

Another example

$$\mathcal{F}_S \triangleq (x_1) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_3) \wedge (\neg x_3) \wedge (x_4 \vee \neg x_5) \wedge (\neg x_4 \vee x_5)$$

Outline

MHS approach for MaxSAT

- **Remark 1**: The MaxSAT solution is a smallest MCS
- **Remark 2**: Any MCS is a hitting set of all MUSes
- **Approach**: [DB'11]
  1. Let $\mathcal{K}$ be a set of unsatisfiable cores (or MUSes)
  2. Find a minimum hitting set $\mathcal{H}$ of the set $\mathcal{K}$ of already computed cores (or MUSes)
  3. Check satistisfability of $\mathcal{F} \setminus \mathcal{H}$
  4. If satisfiable, then $\mathcal{H}$ is a smallest MCS; **terminate** and return $\mathcal{H}$
  5. Otherwise, compute core (or MUS) and add it to $\mathcal{K}$
  6. Loop from 2
- **Issue**: worst-case number of iterations worst-case exponential on number of clauses
  - But, quite effective in practice

$c_1 = x_6 \vee x_2 \vee A_1$ $c_2 = \neg x_6 \vee x_2 \vee A_2$ $c_3 = \neg x_2 \vee x_1 \vee A_3$ $c_4 = \neg x_1 \vee A_4$

$c_5 = \neg x_6 \vee x_8 \vee A_5$ $c_6 = x_6 \vee \neg x_8 \vee A_6$ $c_7 = x_2 \vee x_4 \vee A_7$ $c_8 = \neg x_4 \vee x_5 \vee A_8$

$c_9 = x_7 \vee x_5 \vee A_9$ $c_{10} = \neg x_7 \vee x_5 \vee A_{10}$ $c_{11} = \neg x_5 \vee x_3 \vee A_{11}$ $c_{12} = \neg x_3 \vee A_{12}$

$\mathcal{K} = \emptyset$

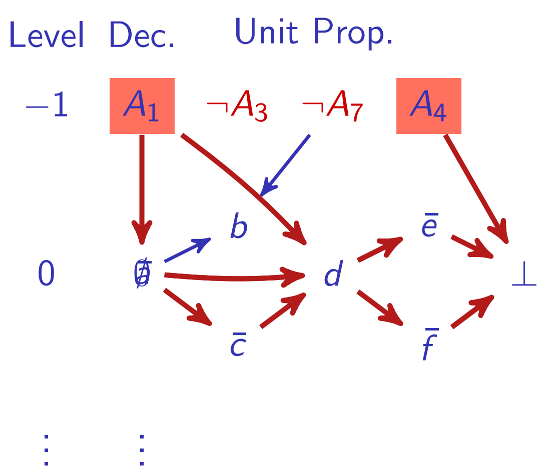
$\mathcal{K} = \{\{c_1, c_2, c_3, c_4\}\}$

$\mathcal{K} = \{\{c_1, c_2, c_3, c_4\}, \{c_9, c_{10}, c_{11}, c_{12}\}\}$

$\mathcal{K} = \{\{c_1, c_2, c_3, c_4\}, \{c_9, c_{10}, c_{11}, c_{12}\}, \{c_3, c_4, c_7, c_8, c_{11}, c_{12}\}\}$

- To every $c_i \in \mathcal{F}$, add a new literal A_i. Set A_i to **true** to relax c_i, or to **false** to activate it
- Find MHS of $\mathcal{K}$: $\emptyset$
- SAT($\mathcal{F} \setminus \emptyset$)? No
- Core of $\mathcal{F}$: $\{c_1, c_2, c_3, c_4\}$. Update $\mathcal{K}$
- Find MHS of $\mathcal{K}$: E.g. $\{c_1\}$
- SAT($\mathcal{F} \setminus \{c_1\}$)? No
- Core of $\mathcal{F}$: $\{c_9, c_{10}, c_{11}, c_{12}\}$. Update $\mathcal{K}$
- Find MHS of $\mathcal{K}$: E.g. $\{c_1, c_9\}$
- SAT($\mathcal{F} \setminus \{c_1, c_9\}$)? No
- Core of $\mathcal{F}$: $\{c_3, c_4, c_7, c_8, c_{11}, c_{12}\}$. Update $\mathcal{K}$
- Find MHS of $\mathcal{K}$: E.g. $\{c_4, c_9\}$
- Terminate & return 2

- Assign the activation literals at a special decision level (-1)
- CDCL fails when finding a contradiction at level 0
  - The implication graph must involve some activation literals
- Do clause resolution until the cut contains only activation literals
- The resulting clause is a MUS of the original formula



MAXHS: Pseudocode

Algorithm: MAXHS

$\mathcal{K}\emptyset$// The MUSs
$\sigma \leftarrow \emptyset$// The optimal model
while satisfiability $\neq$ **SAT do**
  $hs \leftarrow$ Find-MinCost-HittingSet($\mathcal{K}$);
  $(sat, \kappa, \sigma) \leftarrow$ CDCL($\mathcal{F} \setminus hs$);
  add κ to $\mathcal{K}$;
end
return σ;

- CDCL returns the tuple (sat, κ, σ) where:
  - *sat* is in $\{$**SAT**, **UNSAT**, **UNKNOWN**$\}$
  - κ is a MUS
  - σ is a solution if $=$(**SAT**) $=$ **true**

- **Je recrute un postdoc!**
  - Planification des prises de vue et vidages d'une constellation de satellites d'observation (Projet JAPETUS – PROMETHEE, CNES, CNRS, LEANSPACE)