Community-based Partitioning for MaxSAT Solving

Ruben Martins Vasco Manquinho Inês Lynce

IST/INESC-ID, Technical University of Lisbon, Portugal

July 10, 2013

What is Maximum Satisfiability?

CNF Formula:

$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	x_1
x_3	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- Formula is unsatisfiable

What is Maximum Satisfiability?

CNF Formula:

$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	x_1
x_3	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- Formula is unsatisfiable
- Maximum Satisfiability (MaxSAT):
  - Find an assignment that maximizes (minimizes) number of satisfied (unsatisfied) clauses

What is Maximum Satisfiability?

CNF Formula:

$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	x_1
x_3	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- An optimal solution would be:
  - $\nu = \{x_1 = 1, x_2 = 1, x_3 = 1\}$
- This assignment unsatisfies only 1 clause

MaxSAT Problems

- MaxSAT:
  - All clauses are soft
  - Minimize number of unsatisfied soft clauses

MaxSAT Problems

- MaxSAT:
  - All clauses are soft
  - Minimize number of unsatisfied soft clauses
- Partial MaxSAT:
  - Clauses are soft or hard
  - Hard clauses must be satisfied
  - Minimize number of unsatisfied soft clauses

MaxSAT Problems

- MaxSAT:
  - All clauses are soft
  - Minimize number of unsatisfied soft clauses
- Partial MaxSAT:
  - Clauses are soft or hard
  - Hard clauses must be satisfied
  - Minimize number of unsatisfied soft clauses
- Weighted Partial MaxSAT:
  - Clauses are soft or hard
  - Weights associated with soft clauses
  - Minimize sum of weights of unsatisfied soft clauses

MaxSAT Algorithms

- Branch and Bound:
  - Extensive use of lower bounding procedures
  - Restrictive use of MaxSAT inference rules
- Linear search on the number of unsatisfied clauses:
  - Each time a new solution is found, a new constraint is added that excludes solutions with higher cost
- Unsatisfiability-based solvers:
  - Iterative identification and relaxation of unsatisfiable subformulas

MaxSAT Algorithms

- Branch and Bound:
  - Extensive use of lower bounding procedures
  - Restrictive use of MaxSAT inference rules
- Linear search on the number of unsatisfied clauses:
  - Each time a new solution is found, a new constraint is added that excludes solutions with higher cost
- **Unsatisfiability-based solvers:**
  - Iterative identification and relaxation of unsatisfiable subformulas

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h (Hard):		$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	
φ_s (Soft):	x_1	x_3	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h:		$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	
φ_s:	x_1	x_3	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- Formula is unsatisfiable

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h:		$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	
φ_s:	x_1	x_3	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- Formula is unsatisfiable
- Identify an unsatisfiable core

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h: $\bar{x}_2 \vee \bar{x}_1$ $\quad x_2 \vee \bar{x}_3$ $\quad \text{CNF}(r_1 + r_2 \leq 1)$

φ_s: $x_1 \vee r_1$ $\quad x_3 \vee r_2$ $\quad x_2 \vee \bar{x}_1$ $\quad \bar{x}_3 \vee x_1$

- Relax unsatisfiable core:
  - Add relaxation variables
  - Add at-most-one constraint

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h:	$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	$\text{CNF}(r_1 + r_2 \leq 1)$	
φ_s:	$x_1 \vee r_1$	$x_3 \vee r_2$	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- Formula is unsatisfiable

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h:	$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	$\text{CNF}(r_1 + r_2 \leq 1)$	
φ_s:	$x_1 \vee r_1$	$x_3 \vee r_2$	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- Formula is unsatisfiable
- Identify an unsatisfiable core

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h: $\bar{x}_2 \vee \bar{x}_1$ $\quad x_2 \vee \bar{x}_3$ $\quad \text{CNF}(r_1 + r_2 \leq 1)$ $\quad \text{CNF}(r_3 + \ldots + r_6 \leq 1)$

φ_s: $x_1 \vee r_1 \vee r_3$ $\quad x_3 \vee r_2 \vee r_4$ $\quad x_2 \vee \bar{x}_1 \vee r_5$ $\quad \bar{x}_3 \vee x_1 \vee r_6$

- Relax unsatisfiable core:
  - Add relaxation variables
  - Add at-most-one constraint

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h:	$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	$\text{CNF}(r_1 + r_2 \leq 1)$	$\text{CNF}(r_3 + \ldots + r_6 \leq 1)$
φ_s:	$x_1 \vee r_1 \vee r_3$	$x_3 \vee r_2 \vee r_4$	$x_2 \vee \bar{x}_1 \vee r_5$	$\bar{x}_3 \vee x_1 \vee r_6$

- Formula is satisfiable
- An optimal solution would be:
  - $\nu = \{x_1 = 1, x_2 = 0, x_3 = 0\}$

Unsatisfiability-based Algorithms (Fu&Malik [SAT'06])

Partial MaxSAT Formula:

φ_h:		$\bar{x}_2 \vee \bar{x}_1$	$x_2 \vee \bar{x}_3$	
φ_s:	x_1	x_3	$x_2 \vee \bar{x}_1$	$\bar{x}_3 \vee x_1$

- Formula is satisfiable
- An optimal solution would be:
  - $\nu = \{x_1 = 1, x_2 = 0, x_3 = 0\}$
- This assignment unsatisfies 2 soft clauses

Unsatisfiability-based Algorithms

- Fu&Malik algorithm can be generalized for weighted partial MaxSAT (Manquinho et al. [SAT'09], Ansótegui et al. [SAT'09])
- Unsatisfiability-based algorithms are very effective on industrial benchmarks

Unsatisfiability-based Algorithms

- Fu&Malik algorithm can be generalized for weighted partial MaxSAT (Manquinho et al. [SAT'09], Ansótegui et al. [SAT'09])
- Unsatisfiability-based algorithms are very effective on industrial benchmarks
- However, performance is related with the unsatisfiable cores given by the SAT solver:
  - Some unsatisfiable cores may be unnecessarily large

Unsatisfiability-based Algorithms

- Fu&Malik algorithm can be generalized for weighted partial MaxSAT (Manquinho et al. [SAT'09], Ansótegui et al. [SAT'09])
- Unsatisfiability-based algorithms are very effective on industrial benchmarks
- However, performance is related with the unsatisfiable cores given by the SAT solver:
  - Some unsatisfiable cores may be unnecessarily large
  - **Solution:** Partitioning of the soft clauses

Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

γ_1 γ_2 γ_3

Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

(2) Add a new partition to the formula

γ_1 γ_2 γ_3

Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:

- Relax unsatisfiable core

Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:

- Relax unsatisfiable core

(4) The formula is satisfiable:

- If there are no more partitions:
  ▷ Optimum found
- Otherwise, **go back to 2**

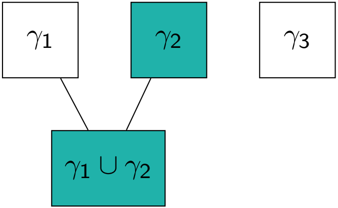
Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:
   - Relax unsatisfiable core

(4) The formula is satisfiable:
   - If there are no more partitions:
     ▷ Optimum found
   - Otherwise, go back to 2



Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

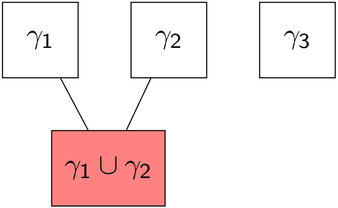
(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:

- Relax unsatisfiable core

(4) The formula is satisfiable:

- If there are no more partitions:
  ▷ Optimum found
- Otherwise, go back to 2



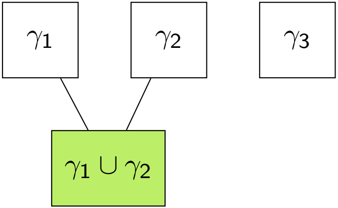
Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:
  - Relax unsatisfiable core

(4) The formula is satisfiable:
  - If there are no more partitions:
    ▷ Optimum found
  - Otherwise, **go back to 2**



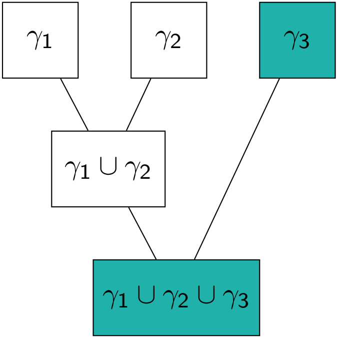
Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:
   - Relax unsatisfiable core

(4) The formula is satisfiable:
   - If there are no more partitions:
     ▷ Optimum found
   - Otherwise, go back to 2



Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

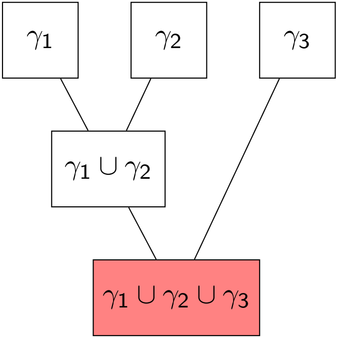
(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:

- Relax unsatisfiable core

(4) The formula is satisfiable:

- If there are no more partitions:
  ▷ Optimum found
- Otherwise, go back to 2



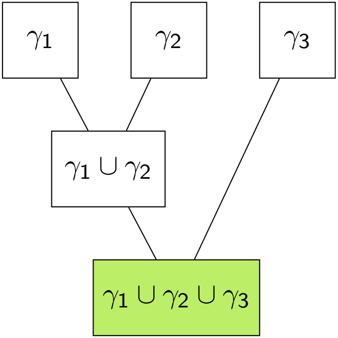
Unsatisfiability-based Algorithm w/ Partitioning (Martins et al. [ECAI'12])

(1) Partition the soft clauses

(2) Add a new partition to the formula

(3) While the formula is unsatisfiable:
   - Relax unsatisfiable core

(4) The formula is satisfiable:
   - If there are no more partitions:
     ▷ **Optimum found**
   - Otherwise, go back to 2



Unsatisfiability-based Algorithms w/ Partitioning

- How to partition the soft clauses?
  - For weighted partial MaxSAT, weight-based partitioning has shown to significantly improve the performance of the solver

    (Martins et al. [ECAI'12], Ansótegui et al. [CP'12])

  - However, for partial MaxSAT all soft clauses have weight 1

Unsatisfiability-based Algorithms w/ Partitioning

- How to partition the soft clauses?
  - For weighted partial MaxSAT, weight-based partitioning has shown to significantly improve the performance of the solver

    (Martins et al. [ECAI'12], Ansótegui et al. [CP'12])

  - However, for partial MaxSAT all soft clauses have weight 1
  - **Solution:** Graph-based partitioning

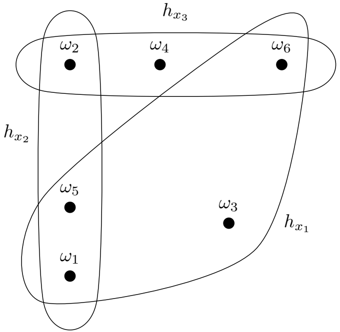
Hypergraph Partitioning

Hypergraph representation of a MaxSAT formula:

- Vertices: Represents each clause
- Hyperedge: For each variable, there is an hyperedge connecting all vertices that represent clauses that contain that variable

Hypergraph Partitioning

- $\omega_1 = [\bar{x}_1 \vee \bar{x}_2]$
- $\omega_2 = [x_2 \vee \bar{x}_3]$
- $\omega_3 = (x_1)$
- $\omega_4 = (x_3)$
- $\omega_5 = (x_2 \vee \bar{x}_1)$
- $\omega_6 = (\bar{x}_3 \vee x_1)$



Partitions given by hypergraph partitioning:

- Only soft clauses are considered in the partitions
- $\gamma_1 = \{\omega_3, \omega_6\}$, $\gamma_2 = \{\omega_4, \omega_5\}$

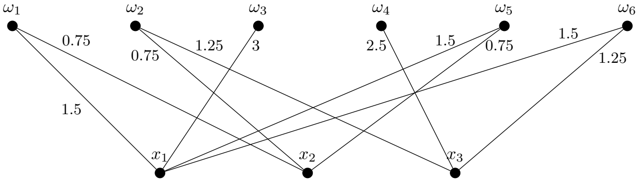
Community-based Partitioning (CVIG)

Clause-Variable Incidence Graph (CVIG) of a MaxSAT formula:

- Vertices: Represents each variable and each clause
- Edges: There is an edge between each variable and each clause where the variable occurs
- Each edge has a corresponding weight:
  - More weight is given to clauses that establish edges between variables that occur in soft clauses (details in the paper)

Community-based Partitioning (CVIG)

- $\omega_1 = [\bar{x}_1 \vee \bar{x}_2]$
- $\omega_2 = [x_2 \vee \bar{x}_3]$
- $\omega_3 = (x_1)$
- $\omega_4 = (x_3)$
- $\omega_5 = (x_2 \vee \bar{x}_1)$
- $\omega_6 = (\bar{x}_3 \vee x_1)$



Partitions given by the identification of communities:

- $\gamma_1 = \{\omega_3, \omega_5\}$, $\gamma_2 = \{\omega_4, \omega_6\}$

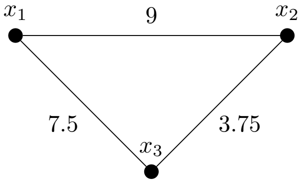
Community-based Partitioning (VIG)

Variable Incidence Graph (VIG) of a MaxSAT formula:

- Vertices: Represents each variable
- Edge: If two variables belong to the same clause, then there is an edge between them
- Each edge has a corresponding weight:
  - More weight is given to clauses that establish edges between variables that occur in soft clauses (details in the paper)

Community-based Partitioning (VIG)

- $\omega_1 = [\bar{x}_1 \vee \bar{x}_2]$
- $\omega_2 = [x_2 \vee \bar{x}_3]$
- $\omega_3 = (x_1)$
- $\omega_4 = (x_3)$
- $\omega_5 = (x_2 \vee \bar{x}_1)$
- $\omega_6 = (\bar{x}_3 \vee x_1)$



Partitions given by the identification of communities:

- Mapping from the partition of variables to clauses
- $\gamma_1 = \{\omega_3, \omega_4, \omega_5, \omega_6\}$

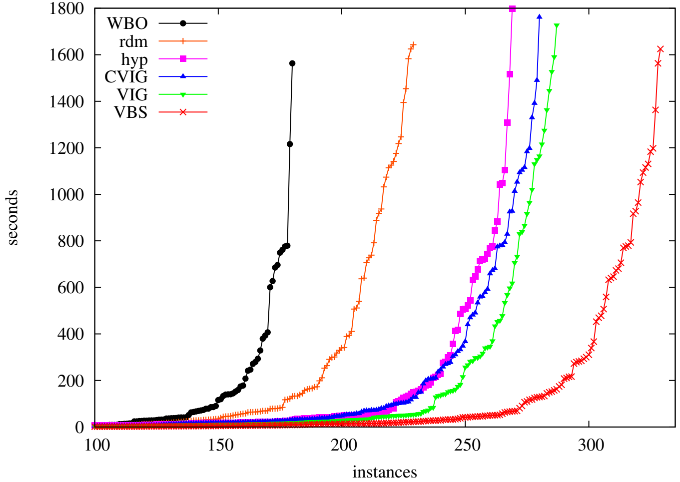
Experimental Results

- Benchmarks:
  - 504 industrial partial MaxSAT instances

- Solvers:
  - `WBO`
  - `rdm` (Random partitioning – 16 partitions)
  - `hyp` (Hypergraph partitioning – 16 partitions)
  - `VIG` (Community partitioning – Variable Incidence Graph)
  - `CVIG` (Community partitioning – Clause-Variable Incidence Graph)
  - `VBS` (Virtual Best Solver)

Experimental Results

- Running times of solvers for industrial partial MaxSAT instances



Conclusions

- Partitioning approaches outperform WBO on most instances:
  - Finds smaller unsatisfiable cores:
    e.g. WBO: avg. 110 soft clauses VS. VIG: avg. 66 soft clauses
- All algorithms contribute to the VBS:
  - Different graph-based partition methods solve different instances
  - Using the structure of the formula improves the partitioning
- Partitioning approaches are not limited to WBO:
  - The same idea can be applied to other unsatisfiability-based algorithms

Questions?