Improvements to Core-Guided Binary Search for MaxSAT

A.Morgado F.Heras J.Marques-Silva

CASL/CSI, University College Dublin
Dublin, Ireland

SAT, June 2012

Outline

MaxSAT

Maximum Satisfiability (MaxSAT) Given a propositional formula in Conjunctive Normal Form (CNF), compute an assignment to the variables that maximizes the number of satisfied clauses

$$(x_1) \quad \land \quad (\neg x_1 \lor \neg x_2) \quad \land \quad (x_2)$$

The instance is unsatisfiable, and the MaxSAT solution is 2

MaxSAT

Maximum Satisfiability (MaxSAT) Given a propositional formula in Conjunctive Normal Form (CNF), compute an assignment to the variables that maximizes the number of satisfied clauses

$$(x_1) \quad \wedge \quad (\neg x_1 \vee \neg x_2) \quad \wedge \quad (x_2)$$

The instance is unsatisfiable, and the MaxSAT solution is 2

MaxSAT as MinUNSAT - MaxSAT solvers compute the minimum number of falsified clauses (1 in the example)

MaxSAT

Maximum Satisfiability (MaxSAT) Given a propositional formula in Conjunctive Normal Form (CNF), compute an assignment to the variables that maximizes the number of satisfied clauses

$$(x_1) \quad \wedge \quad (\neg x_1 \vee \neg x_2) \quad \wedge \quad (x_2)$$

The instance is unsatisfiable, and the MaxSAT solution is 2

MaxSAT as MinUNSAT - MaxSAT solvers compute the minimum number of falsified clauses (1 in the example)

There are 3 variants of the MaxSAT problem

MaxSAT Variants

- Partial MaxSAT
  - Set of clauses divided in two:
    - Hard clauses - have to be satisfied
    - Soft clauses - satisfied or not
  - Objective: Minimize the number of falsified soft clauses while satisfying all hard clauses
- Weighted MaxSAT
  - Clauses have weights associated that represent the cost of falsifying the clause
  - Objective: Minimize the sum of the weights of the falsified clauses
- Weighted Partial MaxSAT
  - Combines the previous two
  - Objective: Minimize the sum of the weights of the falsified soft clauses while satisfying all the hard clauses

MaxSAT Variants

- Partial MaxSAT
  - Set of clauses divided in two:
    - Hard clauses - have to be satisfied
    - Soft clauses - satisfied or not
  - Objective: Minimize the number of falsified soft clauses while satisfying all hard clauses
- Weighted MaxSAT
  - Clauses have weights associated that represent the cost of falsifying the clause
  - Objective: Minimize the sum of the weights of the falsified clauses
- Weighted Partial MaxSAT
  - Combines the previous two
  - Objective: Minimize the sum of the weights of the falsified soft clauses while satisfying all the hard clauses

MaxSAT Variants

- Partial MaxSAT
  - Set of clauses divided in two:
    - Hard clauses - have to be satisfied
    - Soft clauses - satisfied or not
  - Objective: Minimize the number of falsified soft clauses while satisfying all hard clauses
- Weighted MaxSAT
  - Clauses have weights associated that represent the cost of falsifying the clause
  - Objective: Minimize the sum of the weights of the falsified clauses
- Weighted Partial MaxSAT
  - Combines the previous two
  - Objective: Minimize the sum of the weights of the falsified soft clauses while satisfying all the hard clauses

MaxSAT Variants

- Partial MaxSAT
  - Set of clauses divided in two:
    - Hard clauses - have to be satisfied
    - Soft clauses - satisfied or not
  - Objective: Minimize the number of falsified soft clauses while satisfying all hard clauses
- Weighted MaxSAT
  - Clauses have weights associated that represent the cost of falsifying the clause
  - Objective: Minimize the sum of the weights of the falsified clauses
- Weighted Partial MaxSAT
  - Combines the previous two
  - Objective: Minimize the sum of the weights of the falsified soft clauses while satisfying all the hard clauses

MaxSAT Problem

- Complexity:
  - Decision version of MaxSAT is NP-Complete
  - MaxSAT is Δ_2^p-complete

- Many optimization problems can be modeled as MaxSAT: Bayesian Networks, Physics (Spinglass), BioInformatics (haplotyping, protein alignment), Circuit Debugging, Fault Localization, Graph problems (Max-Cut, Max-Clique), etc.

Motivation

- Study MaxSAT algorithms based on iteratively calling a SAT solver
  - most of them require an exponential number of calls to the SAT solver

Motivation

- Study MaxSAT algorithms based on iteratively calling a SAT solver
  - most of them require an exponential number of calls to the SAT solver
  - Binary Search requires a linear number of calls to the SAT solver

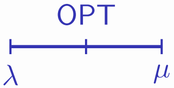
    [fm-SAT06]

Motivation

- Study MaxSAT algorithms based on iteratively calling a SAT solver
  - most of them require an exponential number of calls to the SAT solver
  - Binary Search requires a linear number of calls to the SAT solver [fm-SAT06]
- Extend binary search for MaxSAT:
  - BC - Core-guided binary search: Relax soft clauses on demand as dictated by the unsatisfiable cores found so far [hmms-AAAI11]
  - BCD -Core-guided binary search with disjoint cores: Maintaining disjoint cores allows adding smaller constraints [hmms-AAAI11]
  - **BCD2** - Improvement of BCD: New bounds, better maintenance of the bounds [mhms-SAT12]

Computing a MaxSAT Solution - Notation

- φ - Unsatisfiable Weighted CNF Formula:
  (c_i, w_i) weighted clause:
  - c_i is a clause
  - w_i is a non-negative integer or $\top$;
    cost of falsifying c_i
- $\mathcal{A}$ - Complete Assignment: $\mathcal{A} = \{x_1 = 1, x_2 = 0, \ldots\}$
  Cost of assignment: Sum of weights of falsified clauses
- OPT - Optimum cost: Minimum cost
- μ - Upper Bound: Cost not lower than OPT
- λ - Lower Bound: Value lower than OPT



Outline

Binary Search (1/2)

Proposed by Fu&Malik SAT2006

(c_i, w_i)

$$((x_1), 2) \quad ((x_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

Binary Search (1/2)

Proposed by Fu&Malik SAT2006

$$(c_i, w_i)$$

$$\Downarrow$$

$$(c_i \vee r_i,\ w_i)$$

$$((x_1), 2) \quad ((x_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

$$\Downarrow$$

$$((x_1 \vee r_1),\ 2) \quad ((x_2 \vee r_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

Binary Search (1/2)

Proposed by Fu&Malik SAT2006

$$(c_i, w_i)$$

$$\Downarrow$$

$$(c_i \vee r_i,\ w_i)$$

$$((x_1), 2) \quad ((x_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

$$\Downarrow$$

$$((x_1 \vee r_1),\ 2) \quad ((x_2 \vee r_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

If $\mathcal{A} \not\models c_i$ then $\mathcal{A} \models r_i$

Binary Search (1/2)

Proposed by Fu&Malik SAT2006

$$(c_i, w_i)$$

$$\Downarrow$$

$$(c_i \vee r_i,\ w_i)$$

$$((x_1), 2) \quad ((x_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

$$\Downarrow$$

$$((x_1 \vee r_1),\ 2) \quad ((x_2 \vee r_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

If $\mathcal{A} \not\models c_i$ then $\mathcal{A} \models r_i$

Minimize sum of weights of true relaxation variables

$\Leftrightarrow$

Minimize sum of weights of falsified soft clauses

Binary Search (2/2)

In each iteration:

- Consider τ - maximum allowed sum of weights of true r.v.

Binary Search (2/2)

In each iteration:

- Consider τ - maximum allowed sum of weights of true r.v.

  $\tau = \lfloor \frac{\mu+\lambda}{2} \rfloor$

Binary Search (2/2)

In each iteration:

- Consider τ - maximum allowed sum of weights of true r.v.

  $$\tau = \lfloor \frac{\mu+\lambda}{2} \rfloor$$

- Add PB-constraint $CNF(\sum w_i\ r_i \leq \tau)$

Binary Search (2/2)

In each iteration:

- Consider τ - maximum allowed sum of weights of true r.v.

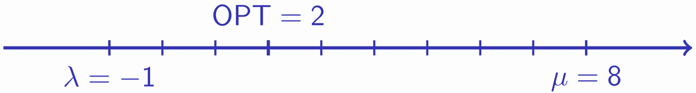
  $\tau = \lfloor \frac{\mu+\lambda}{2} \rfloor$

- Add PB-constraint $CNF(\sum w_i \; r_i \leq \tau)$

$((x_1 \vee r_1),\ 2) \quad ((x_2 \vee r_2), 5)$

$((\neg x_1 \vee \neg x_2),\ \top)$

$\mu_0 = 8 \quad \lambda_0 = -1$



Binary Search (2/2)

In each iteration:

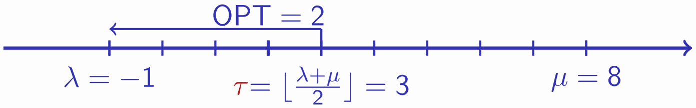
- Consider τ - maximum allowed sum of weights of true r.v.

  $$\tau = \lfloor \frac{\mu+\lambda}{2} \rfloor$$

- Add PB-constraint $CNF(\sum w_i \; r_i \leq \tau)$

$$((x_1 \vee r_1),\ 2) \quad ((x_2 \vee r_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

$$\mu_0 = 8 \quad \lambda_0 = -1$$



Binary Search (2/2)

In each iteration:

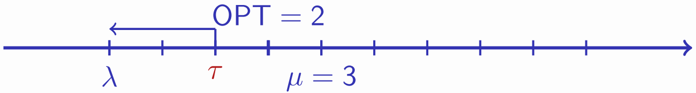
- Consider τ - maximum allowed sum of weights of true r.v.

  $$\tau = \lfloor \frac{\mu+\lambda}{2} \rfloor$$

- Add PB-constraint $CNF(\sum w_i \; r_i \leq \tau)$

$$((x_1 \vee r_1),\; 2) \quad ((x_2 \vee r_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\; \top)$$

$$\mu_1 = 3 \quad \lambda_1 = -1$$



Binary Search (2/2)

In each iteration:

- Consider τ - maximum allowed sum of weights of true r.v.

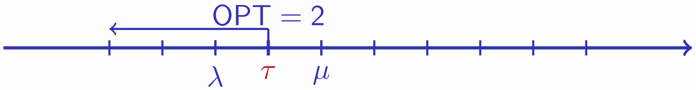
  $$\tau = \lfloor \frac{\mu+\lambda}{2} \rfloor$$

- Add PB-constraint $CNF(\sum w_i \; r_i \leq \tau)$

$((x_1 \vee r_1),\ 2) \quad ((x_2 \vee r_2), 5)$

$((\neg x_1 \vee \neg x_2),\ \top)$

$\mu_2 = 3 \quad \lambda_2 = 1$



Binary Search (2/2)

In each iteration:

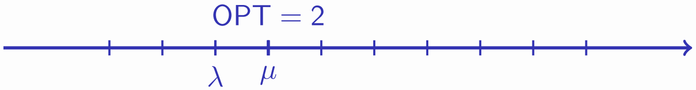
- Consider τ - maximum allowed sum of weights of true r.v.

$$\tau = \lfloor \frac{\mu+\lambda}{2} \rfloor$$

- Add PB-constraint $CNF(\sum w_i \; r_i \leq \tau)$

$$((x_1 \vee r_1),\ 2) \quad ((x_2 \vee r_2), 5)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

$$\mu_3 = 2 \quad \lambda_3 = 1$$



BC - Core-Guided Binary Search

- Binary Search relaxes all soft clauses
- State of the art SAT solvers report Unsatisfiable Cores on unsatisfiable formulas
  - φ_C - Unsatisfiable Core: subset of unsatisfiable clauses

BC - Core-Guided Binary Search

- Binary Search relaxes all soft clauses
- State of the art SAT solvers report Unsatisfiable Cores on unsatisfiable formulas
  - φ_C - Unsatisfiable Core: subset of unsatisfiable clauses

- BC only relaxes a clause when it belongs to an unsatisfiable core
  - Initially no clause is relaxed
  - On every unsatisfiable iteration obtain the core φ_C
  - **If** φ_C contains unrelaxed soft clauses clauses
    then relax unrelaxed soft clauses
    else proceed as Binary Search

BC - Core-Guided Binary Search

- Binary Search relaxes all soft clauses
- State of the art SAT solvers report Unsatisfiable Cores on unsatisfiable formulas
  - φ_C - Unsatisfiable Core: subset of unsatisfiable clauses
- BC only relaxes a clause when it belongs to an unsatisfiable core
  - Initially no clause is relaxed
  - On every unsatisfiable iteration obtain the core φ_C
  - **If** φ_C contains unrelaxed soft clauses clauses
    then relax unrelaxed soft clauses
    else proceed as Binary Search

$$((x_1,\ 2) \quad ((x_2), 5) \quad ((x_6), 1)$$
$$((\neg x_1 \vee \neg x_2),\ \top)$$

Outline

BCD/BCD2 - Core-Guided Binary Search with Disjoint Cores

- BC maintains one "large" PB-constraint
- Some cores do not intersect other cores throughout the iterations

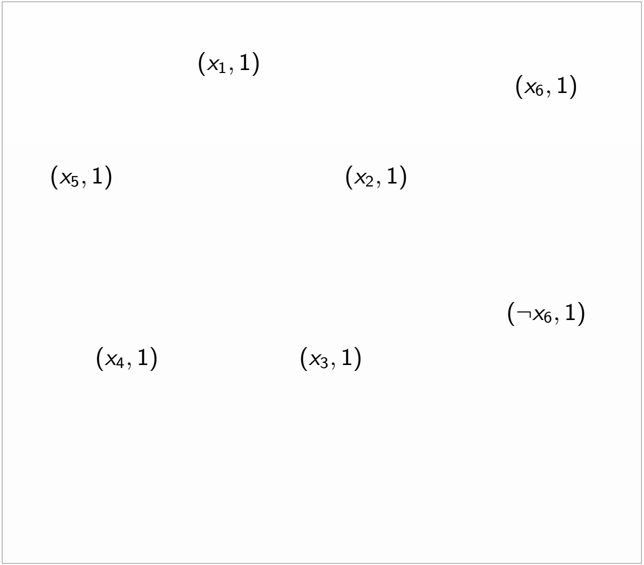
BCD/BCD2 - Core-Guided Binary Search with Disjoint Cores

- BC maintains one "large" PB-constraint
- Some cores do not intersect other cores throughout the iterations

- BCD/BCD2 maintain disjoint cores separate:
  - Each disjoint core C_i is a core with it own:
    - μ_i - upper bound
    - λ_i - lower bound
    - R_i - set of relaxation variables
    - τ_i - middle value
  - Each disjoint core C_i adds a PB-constraint to the formula

$$CNF\left(\sum_{r_j \in R_i} w_j r_j \leq \tau_i\right)$$

BCD/BCD2 on Example



$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1, 1)$

$(x_6, 1)$

$(x_5, 1)$ $(x_2, 1)$

$(\neg x_6, 1)$

$(x_4, 1)$ $(x_3, 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1, 1)$

$(x_6 \vee r_6, 1)$

$(x_5, 1)$ $(x_2, 1)$

$(\neg x_6 \vee r_7, 1)$

$(x_4, 1)$ $(x_3, 1)$

$\mu = 3$

$\lambda = 0$

$\tau = 1$

$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$

$(\neg x_2 \vee \neg x_3, \top)$

$(\neg x_3 \vee \neg x_4, \top)$

$(\neg x_4 \vee \neg x_5, \top)$

$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1, 1)$

$(x_6 \vee r_6, 1)$

$(x_5, 1)$ $(x_2, 1)$

$(\neg x_6 \vee r_7, 1)$

$(x_4, 1)$ $(x_3, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_6 \vee r_6, 1)$

$(x_5, 1)$

$(x_2 \vee r_2, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_1 + r_2 \leq 1)$

$(\neg x_6 \vee r_7, 1)$

$(x_4, 1)$

$(x_3, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_6 \vee r_6, 1)$

$(x_5, 1)$

$(x_2 \vee r_2, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_1 + r_2 \leq 1)$

$(\neg x_6 \vee r_7, 1)$

$(x_4, 1)$ $(x_3, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_6 \vee r_6, 1)$

$(x_5, 1)$

$(x_2 \vee r_2, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_1 + r_2 \leq 1)$

$(\neg x_6 \vee r_7, 1)$

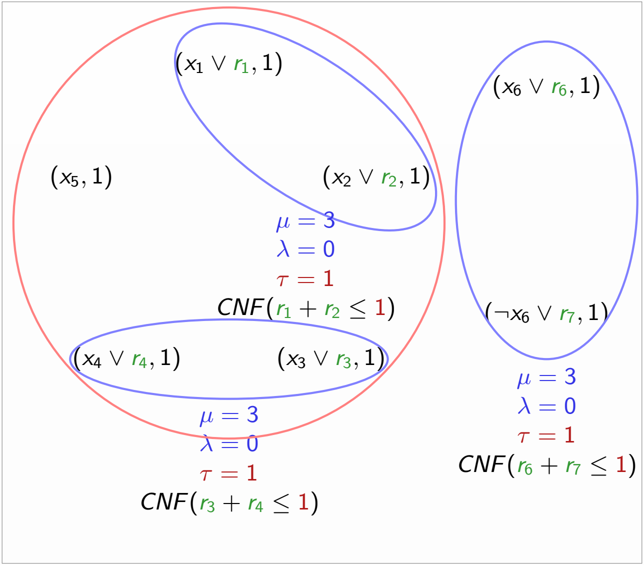
$(x_4 \vee r_4, 1)$ $(x_3 \vee r_3, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_3 + r_4 \leq 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example



$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_5 \vee r_5, 1)$ $(x_2 \vee r_2, 1)$

$(x_4 \vee r_4, 1)$ $(x_3 \vee r_3, 1)$

$\mu = 6$
$\lambda = 0$
$\tau = 3$
$CNF(r_1 + r_2 + r_3 + r_4 + r_5 \leq 3)$

$(x_6 \vee r_6, 1)$

$(\neg x_6 \vee r_7, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_5 \vee r_5, 1)$ $(x_2 \vee r_2, 1)$

$(x_4 \vee r_4, 1)$ $(x_3 \vee r_3, 1)$

$\mu = 6$
$\lambda = 0$
$\tau = 3$
$CNF(r_1 + r_2 + r_3 + r_4 + r_5 \leq 3)$

$(x_6 \vee r_6, 1)$

$(\neg x_6 \vee r_7, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_5 \vee r_5, 1)$ $(x_2 \vee r_2, 1)$

$(x_4 \vee r_4, 1)$ $(x_3 \vee r_3, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_1 + r_2 + r_3 + r_4 + r_5 \leq 1)$

$(x_6 \vee r_6, 1)$

$(\neg x_6 \vee r_7, 1)$

$\mu = 1$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_5 \vee r_5, 1)$ $(x_2 \vee r_2, 1)$

$(x_4 \vee r_4, 1)$ $(x_3 \vee r_3, 1)$

$\mu = 3$
$\lambda = 0$
$\tau = 1$
$CNF(r_1 + r_2 + r_3 + r_4 + r_5 \leq 1)$

$(x_6 \vee r_6, 1)$

$(\neg x_6 \vee r_7, 1)$

$\mu = 1$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example

$(x_1 \vee r_1, 1)$

$(x_5 \vee r_5, 1)$ $(x_2 \vee r_2, 1)$

$(x_4 \vee r_4, 1)$ $(x_3 \vee r_3, 1)$

$\mu = 3$
$\lambda = 1$
$\tau = 2$
$CNF(r_1 + r_2 + r_3 + r_4 + r_5 \leq 2)$

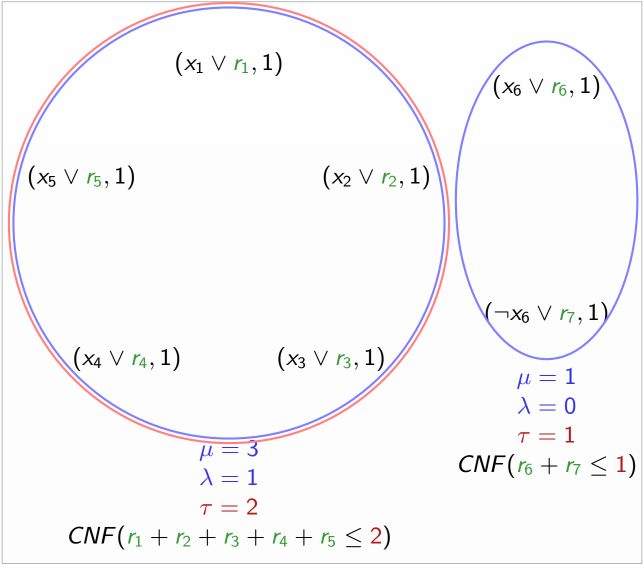
$(x_6 \vee r_6, 1)$

$(\neg x_6 \vee r_7, 1)$

$\mu = 1$
$\lambda = 0$
$\tau = 1$
$CNF(r_6 + r_7 \leq 1)$

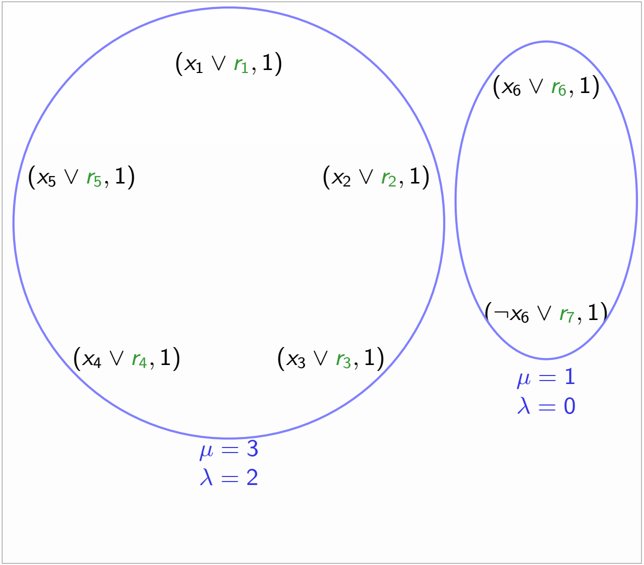
$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD/BCD2 on Example



$$(\neg x_1 \vee \neg x_2, \top)$$
$$(\neg x_2 \vee \neg x_3, \top)$$
$$(\neg x_3 \vee \neg x_4, \top)$$
$$(\neg x_4 \vee \neg x_5, \top)$$
$$(\neg x_5 \vee \neg x_1, \top)$$

BCD/BCD2 on Example



$(\neg x_1 \vee \neg x_2, \top)$
$(\neg x_2 \vee \neg x_3, \top)$
$(\neg x_3 \vee \neg x_4, \top)$
$(\neg x_4 \vee \neg x_5, \top)$
$(\neg x_5 \vee \neg x_1, \top)$

BCD2 - Improvements

- Improvements over BCD
  - On the maintenance of upper bound
    - Cost of Clause
    - Global Upper Bound
  - On the maintenance of lower bound
    - New Lower Bound

BCD2 - Improvements on the upper bound

- Cost of Clause
  - In BCD the cost of a clause consider the relaxation variable
    - $w_i \cdot \mathcal{A}(r_i)$
  - In BCD2 the cost of a clause disregards the relaxation variable
    - $w_i \cdot (1 - \mathcal{A}(c_i \setminus \{r_i\}))$

BCD2 - Improvements on the upper bound

- Cost of Clause
  - In BCD the cost of a clause consider the relaxation variable
    - $w_i \cdot \mathcal{A}(r_i)$
  - In BCD2 the cost of a clause disregards the relaxation variable
    - $w_i \cdot (1 - \mathcal{A}(c_i \setminus \{r_i\}))$
- Global Upper Bound
  - BCD only maintains local upper bounds μ_i
  - $\sum_i \mu_i$ can increase in some iterations
    - conservative update of μ_i over unrelaxed soft clauses
    - the weight of the clause
  - BCD2 maintains μ as global upper bound (and its assignment $\mathcal{A}_\mu$)
    - μ_i updated on unrelaxed soft clauses with the cost of the clause on $\mathcal{A}_\mu$

BCD2 - Improvements on the lower bound 1/2

- New Lower Bound when merging disjoint cores with unrelaxed clauses
  - Merge $C_1, \ldots, C_k$
  - Add $(c_1, w_1), \ldots, (c_m, w_m)$
- BCD: $\lambda_{new} = \lambda_1 + \ldots + \lambda_k$

BCD2 - Improvements on the lower bound 1/2

- New Lower Bound when merging disjoint cores with unrelaxed clauses
  - Merge $C_1, \ldots, C_k$
  - Add $(c_1, w_1), \ldots, (c_m, w_m)$
- BCD: $\lambda_{new} = \lambda_1 + \ldots + \lambda_k$
- BCD2: $\lambda_{new} = \lambda_1 + \ldots + \lambda_k + \Delta$
  - $\Delta = \ldots$ (next slide)
  - Why the current unsatisfiable core?
    Because:
    - Unrelaxed Soft Clauses
    - Disjoint Core needs update of bounds
    - A mix of both

BCD2 - Improvements on the lower bound 2/2

- Why the current unsatisfiable core?
  Because:
  - Unrelaxed Soft Clauses
    - One of the unrelaxed clauses unsatisfied: (c_x, w_x)
    - Increment by w_x

BCD2 - Improvements on the lower bound 2/2

- Why the current unsatisfiable core?
  Because:
  - Unrelaxed Soft Clauses
    - One of the unrelaxed clauses unsatisfied: (c_x, w_x)
    - Increment by w_x
    - Safe minimum: $\delta_1 = \min\{w_1, \ldots, w_m\}$

BCD2 - Improvements on the lower bound 2/2

- Why the current unsatisfiable core?
  Because:
  - Unrelaxed Soft Clauses
    - One of the unrelaxed clauses unsatisfied: (c_x, w_x)
    - Increment by w_x
    - Safe minimum: $\delta_1 = \min\{w_1, \ldots, w_m\}$
  - Disjoint Core needs update of bounds
    - Update lower bound of disjoint core: $\lambda_x = \tau_x$
    - Increment by $\tau_x - \lambda_x$

BCD2 - Improvements on the lower bound 2/2

- Why the current unsatisfiable core?
  Because:
  - Unrelaxed Soft Clauses
    - One of the unrelaxed clauses unsatisfied: (c_x, w_x)
    - Increment by w_x
    - Safe minimum: $\delta_1 = \min\{w_1, \ldots, w_m\}$
  - Disjoint Core needs update of bounds
    - Update lower bound of disjoint core: $\lambda_x = \tau_x$
    - Increment by $\tau_x - \lambda_x$
    - Safe minimum: $\delta_2 = \min\{\tau_1 - \lambda_1, \ldots, \tau_k - \lambda_k\}$

BCD2 - Improvements on the lower bound 2/2

- Why the current unsatisfiable core?
  Because:
  - Unrelaxed Soft Clauses
    - One of the unrelaxed clauses unsatisfied: (c_x, w_x)
    - Increment by w_x
    - Safe minimum: $\delta_1 = \min\{w_1, \ldots, w_m\}$
  - Disjoint Core needs update of bounds
    - Update lower bound of disjoint core: $\lambda_x = \tau_x$
    - Increment by $\tau_x - \lambda_x$
    - Safe minimum: $\delta_2 = \min\{\tau_1 - \lambda_1, \ldots, \tau_k - \lambda_k\}$
  - A mix of both
    - Safe minimum: $\delta_1 + \delta_2$

BCD2 - Improvements on the lower bound 2/2

- Why the current unsatisfiable core?
  Because:
  - Unrelaxed Soft Clauses
    - One of the unrelaxed clauses unsatisfied: (c_x, w_x)
    - Increment by w_x
    - Safe minimum: $\delta_1 = \min\{w_1, \ldots, w_m\}$
  - Disjoint Core needs update of bounds
    - Update lower bound of disjoint core: $\lambda_x = \tau_x$
    - Increment by $\tau_x - \lambda_x$
    - Safe minimum: $\delta_2 = \min\{\tau_1 - \lambda_1, \ldots, \tau_k - \lambda_k\}$
  - A mix of both
    - Safe minimum: $\delta_1 + \delta_2$
- In BCD2 $\Delta = \min\{\delta_1, \delta_2, \delta_1 + \delta_2\} = \min\{\delta_1, \delta_2\}$

Outline

Additional Techniques

- **Hardening**
  - Adapted the Hardening Rule from B&B MaxSAT algorithms
  - Use global μ and $\sum \lambda_i$ as bounds
  - Whenever bounds are updated then test hardening rule on remaining soft clauses
  - May declare soft clauses as hard
  - Details in the paper

Additional Techniques

- **Hardening**
  - Adapted the Hardening Rule from B&B MaxSAT algorithms
  - Use global μ and $\sum \lambda_i$ as bounds
  - Whenever bounds are updated then test hardening rule on remaining soft clauses
  - May declare soft clauses as hard
  - Details in the paper

- **Biased Search**
  - Decide τ depending on the outcomes of previous iterations
  - Not necessarily the middle point
  - Details in the paper

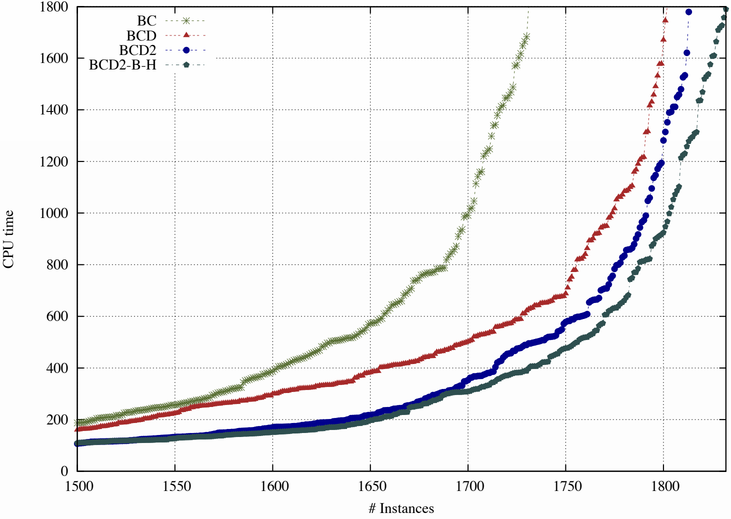
Outline

Experimental Results

- Linux cluster with 50 nodes, each node 3Ghz and 32 GB
- Time limit 1800 seconds and memory limit 4GB

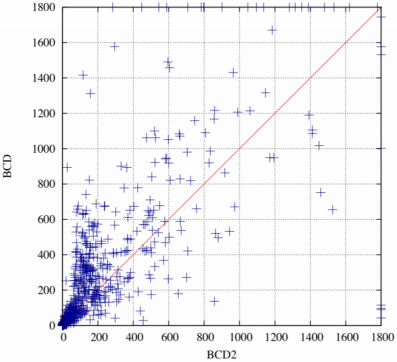
- Instances from 2009-2011 MaxSAT Evaluation instances
  - All Non-Random instances - crafted and industrial
  - All categories: (Plain) MaxSAT, Partial MaxSAT, Weighted MaxSAT, Weighted Partial MaxSAT
  - $>$ 2600 instances

BC, BCD and BCD2 (and extensions)

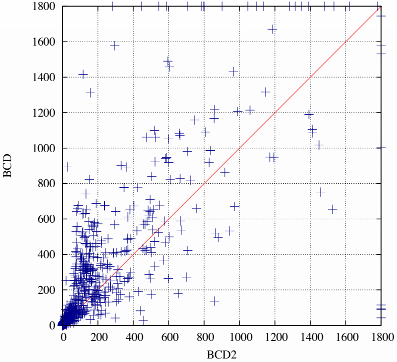


BC 1730; BCD 1801; BCD2 1813; BCD2-B-H 1832

BCD vs BCD2



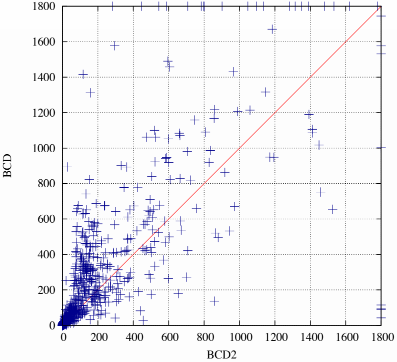
BCD vs BCD2



1305 instances with $\geq$ 20% runtime difference:

- 918 BCD2 wins
- 387 BCD wins

BCD vs BCD2



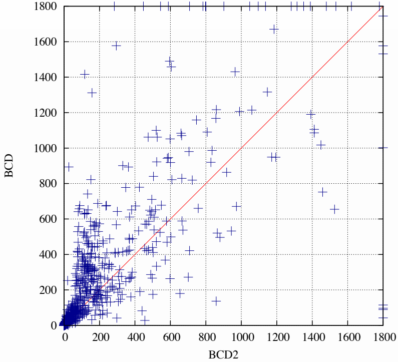
1305 instances with $\geq 20\%$ runtime difference:

- 918 BCD2 wins
- 387 BCD wins

1793 instances solved by both:

- BCD: 124 907 SAT calls
- BCD2: 68 690 SAT calls

BCD vs BCD2



1305 instances with $\geq$ 20% runtime difference:

- 918 BCD2 wins
- 387 BCD wins

1793 instances solved by both:

- BCD: 124 907 SAT calls
- BCD2: 68 690 SAT calls
- close 50% less calls

Outline

Contributions

- Proposed improvements to BCD
  - Improvement on the organization of the BCD algorithm:
    - Maintain a global upper bound μ and associated assignment $\mathcal{A}_\mu$
    - Maintain the contribution of each soft clause to the cost
    - New more aggressive lower bound
    - Reduced the number of iterations (fewer SAT calls)
  - Proposed two techniques - Hardening & Biased Search
    - Can be implemented on any core-guided MaxSAT algorithm (that uses upper and lower bounds)
- BCD2 (with extensions) shown to improved BCD on extensive set of benchmarks

Obrigado. Thank you. Grazie.