STOCHASTIC LOCAL SEARCH
FOUNDATION AND APPLICATION

MAX-SAT & MAX-CSP

Presented by:
Wei-Lwun Lu

1

The MAX-SAX Problems

- MAX-SAT is the optimization variant of SAT.
- Unweighted MAX-SAT:
  - Finds a variable assignment that maximizes the number of the satisfied clauses.
  - Standard simplification techniques (e.g. unit propagation, pure literal reduction) are not directly applicable to MAX-SAT.

2

- Weighted MAX-SAT:

  Finds a variable assignment that maximizes the total weights of satisfied clauses in the formula

- Hard Constraints: constraints that must be satisfied. (assign high weights)
- Soft Constraints: constraints whose violation does not preclude feasibility. (assign small weights)

3

- MAX-SAT is *NP-Hard*
  - because SAT can be reduced to MAX-SAT in polynomial time.
- Many polynomial time approximation algorithm

  (e.g. Asano and Williamson's 1.275-approximation)

- Empirically, approximation algorithms achieve better solution quality; however, SLS is much faster.

4

SLS Algorithms for MAX-SAT

- Any SLS algorithms for SAT can be applied to unweighted MAX-SAT by tracking of the incumbent candidate solution.
- SLS algorithms perform well in SAT

  ≠ perform well in MAX-SAT

  (e.g. WalkSAT, Novelty$^+$)

WalkSAT for MAX-SAT

- All WalkSAT algorithms can be extended to MAX-SAT in two different ways:
  1. "*we*" mechanism (second stage):

     Use the objective function for weighted MAX-SAT (total weights of satisfied clauses)
  2. "*wcs*" mechanism (first stage):

     Consider clause weights in the selection of an unsatisfied clause.

WalkSAT-JKS

- A variant of WalkSAT/SKC
- In the first stage, restricts the clause selection in the hard constraint clauses until all of them are satisfied. (hard constraint clauses are determined by a given threshold)
- In the second stage, it allows random walk even when 'zero damage' flips are available.

7

Novelty^{+}/*wcs+we*

- Novelty^{+} with with '*we*' and '*wcs*' mechanism.
- Novelty^{+}/*wcs+we* is the state-of-the-art SLS algorithm for MAX-SAT to find quasi-optimal solutions in significantly less CPU time than other high-performance algorithms.

8

DLM-SW

- DLM variant with evaluation function:

$$g'(a) = \sum_{c \in CU(a)} \left(clp(c) + w(c) \right)$$

- Use an iterative first improvement algorithm.
- Performs better than WalkSAT-JKS;

  worse than Novelty^{+}/*wcs* or Novelty^{+}/*wcs*+*we*

9

DLM-99-SAT

- DLM-99-SAT variant for MAX-SAT
- Initiate penalty clp(c) to w(c)
- The parameters $\delta^{+}, \delta^{-}, \delta_{s}$ is assigned individually to each clause proportional to its weights
- Performs better than DLM-SW;

  worse than Novelty^{+}/*wcs* or Novelty^{+}/*wcs*+*we*

10

GLSSAT

- GLSSAT variant for MAX-SAT
- Update the clauses with maximum utility. The utility is defined as:

$$util(a,c) = w(c)/(1+clp(c))$$

- Performs better than other DLM and WalkSAT in terms of solution quality reached after a fixed number of iterations
- Performs worse than Novelty^{+}/*wcs* or Novelty^{+}/*wcs+we* in terms of CPU times
- Tends to suffer from stagnation behavior.

11

SAMD

- *Steepest Ascent Mildest Descent* (*SAMD*) is the earliest Tabu search for MAX-SAT.
- It only declares *tabu* on variable flipped in non-improving steps; variables flipped in improving steps are not declared *tabu*.

12

TS-YI

- TS-YI is a Tabu search based on first improvement search strategy.
- In each search steps, variable assignment is scanned in random order.
- It always performs better than GSAT; but sometimes worth than WalkSAT/SKC

  (e.g. weighted subset covering instances)

RoTS

- Robust Tabu Search (RoTS) for MAX-SAT
  - Best iterated improvement
  - Robust Tabu Search

    (repeated choose *tt* randomly from interval [tt_{min}, tt_{max}])
  - Aspiration criterion: allows a variable to be flipped regardless of its *tabu* status if this leads to an improvement in the incumbent candidate solution.
  - Force any variable whose value has not been changed over the last 10 n search steps to be flipped.

RoTS (cont.)

- In wjnh instances, RoTS requires more search steps but less CPU time than GLS. But still worse than $Novelty^+$ variants.
- On Weighted Uniform Random 3-SAT instances, RoTS performs better than $Novelty^+$ variants and GLS.

15

ILS-YI

- Iterated Local Search variant for MAX-SAT.
- Subsidiary local search: iterative first improvement.
- Perturbation: fixed number of uninformed random walk steps.
- Performs better than GSAT; but worse than WalkSAT/SKC or TS-YI.

16

IRoTS

- Iterated Local Search variant for MAX-SAT.
- Subsidiary local search: RoTS with smaller *tabu* tenure.
- Perturbation: RoTS with substantial larger *tabu* tenure.
- All variables are declared non-tabu at the beginning of each local search and perterbation.

IRoTS (cont.)

- IRoTS will select new s with the following strategies:
  1. g(s') > g($\hat{s}$): choose s'.
  2. g(s') = g(s): choose one of them in random
  3. otherwise : choose the worse of s and s' with prob. 0.1; the better one with prob. 0.9.

IRoTS (cont.)

- On weighted and unweighted Uniform Random 3-SAT instances, IRoTS performs significantly better than GLS and Novelty+ variants in terms of CPU time.
- On the wjnh instances, IRoTS performs worse than Novelty+ variants.
- For MAX-SAT-encoded instances, IRoTS performs worse than GLS.

19

Non-Oblivious SLS Algorithms

- Non-oblivious evaluation functions reflect the degree of satisfaction of the clauses satisfied by a given variable assignment.

  e.g.

$$g_2(a) := 3/2 \cdot w(S_1(s)) + 2 \cdot w(S_2(s))$$

  For 4/3-approximation for MAX-2-SAT

- Non-oblivious version achieve better worse-case approximation ratio.

20

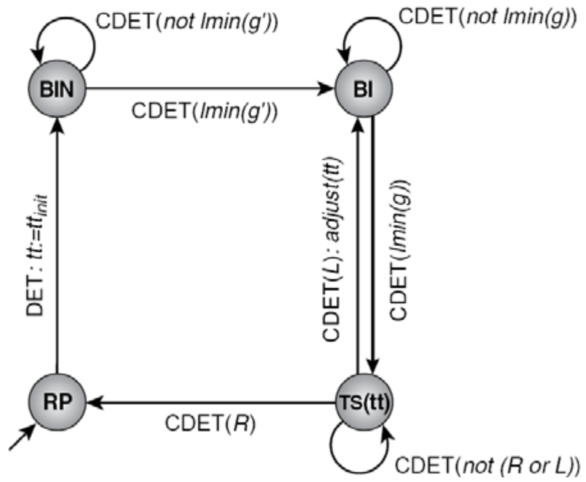
Non-Oblivious SLS Algorithm (con.)

- SLS with non-oblivious evaluation function performs worse than SLS with oblivious evaluation function (e.g. GSAT, GWSAT, GSAT/Tabu).
- Hybrid: first performs non-oblivious SLS then performs oblivious SLS.

H-RTS

- RP: Random Picking
- BIN: Non-oblivious Iterative Best Improvement
- BI: Oblivious Iterative Best Improvement
- TS: Tabu Search
- R: countm(10 n + 1)
- L: scount(2 (tt + 1))

H-RTS (cont.)



23

H-RTS (cont.)

- H-RTS performs significantly better than GSAT, GWSAT, GSAT/Tabu in terms of solution quality achieved after a fixed number of search steps.
- Interestingly, some evidences show that the performance of H-RTS does not significantly depend on non-oblivious local search but on oblivious local search.

24

MAX-CSP Problem

- Unweighted MAX-CSP
  - to find a variable assignment $a*$ that maximizes the number of satisfied constraints.
- Weighted MAX-CAP
  - every constraint is assigned a weight.
  - to find a variable assignment $a*$ that maximizes the total weight of the satisfied constraints.

25

Randomly Generated MAX-SAT Instances

- To generate Uniform Random Binary CSP instances, we have to setup four parameters:
  1. n: # of variables
  2. k: domain size
  3. α : constraint graph density
  4. β : constraint tightness

26

MCH

- SLS algorithms for CSP can be directly applied to MAX-CSP.
- Min-Conflicts Heuristic (MCH) variant
- Typically, WMCH performs better than basic MCH and basic MCH with random restart.

27

TS-GH

- TS-GH determines each search step by considering all (*v*, *y*) pairs for which *v* occurs in a currently violated constraints.
- TS-GH has to be implemented efficiently.
- TS-GH is state-of-the-art SLS algorithm for unweighted MAX-CSP.

28

SLS for Weighted MAX-CSP

- SLS algorithms for weighted MAX-CSP has remained largely unexplored.
- Current approaches:
  – Approximation algorithms
  – Approximation algorithms + Iterated Improvement (APII)
  – Greedy construction heuristic + iterated Improvement (GII)
  – MCH

29