Incomplete Search Methods and SAT

CS410/510CS Lecture 5

Bart Massey <bart@cs.pdx.edu>
PSU CS Department
February 8, 2001

Overview

- Review: Good Complete Search
- More On SAT
- Gradient Descent SAT Search
- Stochastic SAT Search
- Stochastic Search For Generic CSPs

Review: Good Complete Search

Want to recover some material from last class

- Admissible heuristics
- Good complete search techniques

and give suggestions for HW 1c

Admissible Heuristics

Recall *Admissible Heuristic* always (=provably) "optimistic"

In OPTIMAL TENT ASSIGNMENT, admissible is

- 0? 100? 10000?
- Everyone in one big tent?

Note $f(n) = g(n) + h(n)$, i.e. given partial tent packing scoring $g(n)$, need $h(n)$ packing remainder

- Remainder in 1 big tent?
- Perfect scores for remainder?
- Note easy 1-way $\longrightarrow$ 2-way transformation

Depth-First Branch and Bound

DFBnB is *prune* using admissible heuristic

If cannot optimistically finish partial assignment well, do not finish at all

Note: BnB pruning works with *iterated* DFS just fine

- ID
- IB

Limited Discrepancy Search

LDS idea: iterate on number of ignored heuristic bests (= "discrepancies")

- LDS 0: Prune DFS children with any discrepancy
- LDS 1: DFS with initial disc. limit of 1
- LDS 2: DFS with initial disc. limit of 2
  ...
- LDS d: DFS

Why Limited Discrepancy Search?

LDS

- Helps with early mistakes
- Is easy to implement
- Is complete

LDS works with dynamic var + val order, but order must be *consistent* over iterations

Combining DFBnB + LDS

BnB pruning OK on each LDS iteration!

- Iteration is DFS
- Reasoning behind BnB prune still applies

Synergy between LDS and BnB:

- LDS gets good score quicker
- Good score allows BnB to prune more

Homework 1c

HW 1c1: Apply LDS + DFBnB to OPTIMAL TENT ASSIGNMENT

Three heuristics

1. variable-ordering
2. admissible goal-value
3. value-ordering

Can use 2 for 3, but ties should be broken sensibly

More On SAT

"Side-track" to Boolean CSPs: SAT. Easy to

- think about space
- write down problems/instances
- generate problems

Already saw complete SAT algorithms

Conjunctive Normal Form

- Consider a proposition in Boolean variables

$$\neg(A \wedge B \wedge C \wedge D) \wedge (A \vee B)$$

- All such can be transformed to "Conjunctive Normal Form" (CNF)

$$(\neg A \vee \neg B \vee \neg C \vee \neg D) \wedge (A \vee B)$$

- Result: SAT instance

N-ary Boolean Constraint Satisfaction

Leave SAT formula in full clausal form (no 3-SAT)

- Every var domain binary
- Constraints n-ary

Solve using CSP methods

Davis-Putnam

Recall DP: DFS with “fixed” var order, dynamic val order, *unit propagation*

Should now think about

- Dynamic var order heuristics
- Dynamic val order heuristics
- Pruning heuristics

Heuristics For DP

Obvious var ideas

- Var in shortest clause first (shorten clauses by assignments)
- Var in shortest clause > 2 first? (works!)
- Var in most clauses first

Obvious val ideas

- Val eliminating most clauses
- Val producing most binary clauses?

DP Preprocessing

Try to prune space?

- Dynamically: difficult, but do unit prop
- Statically: detect more complicated relationships
  - Eliminate singleton clauses
  - Notice short contradictions
  - Other cleverness

Good Complete SAT Search?

Why not use good complete methods?

- CSP: ID makes no sense
- Binary: IB makes no sense
- DFBnB: No good admissible heuristic
- LDS: works ok! (see Gent+Walsh for ref)

Gradient Descent SAT Search

- Without heuristics: give up completeness for performance = ISAMP
- With simple value heuristics: give up completeness for performance = Weighted ISAMP
- With good var + val + dist heuristics: give up completeness for performance?

Answer: try "gradient descent" search = greedy improvement of non-solution

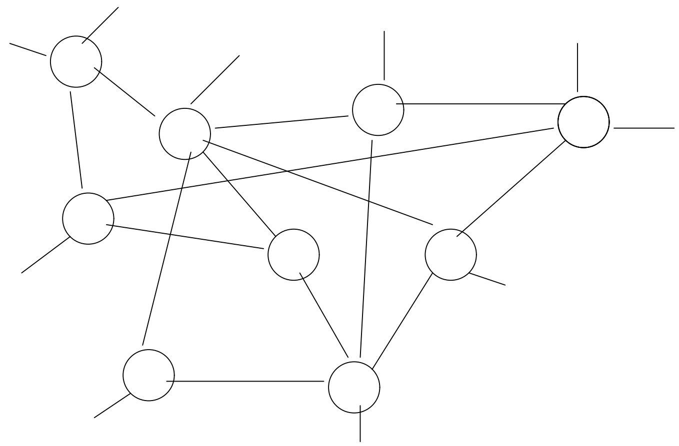
Greedy Heuristic Search

Idea: suppose given total assignment to SAT problem

- Not soln, because 1 or more clauses false
- "Flip" any var in broken clause: fixes it
- Call move that fixes more than breaks "downhill move"
- Greedy heuristic search: pick best downhill move until all clauses SAT.

Local Search

What's "local" about greedy search? Consider search space diagram:



Note that only local moves are considered: no global context

Weighted ISAMP vs. Local Search

Weighted ISAMP is a kind of local search:

- Heuristic is "probabilistic greedy"
- Works on partial assignment
- Only consider *extending* partial assignment

Biggest problem: Weighted ISAMP *gives up too easily*— "near miss" leaf as good as a mile

(Can fix by e.g. bounded backtracking. Better to move toward good leaves?)

Starting Local Search

Local search defined over total assignments. Bootstrap problem?

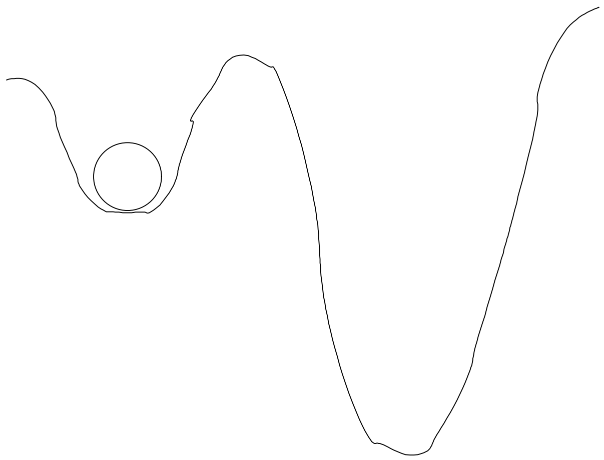
Obvious ideas:

- Use heuristic 1-SAMP for initial assignment
- Assign each var heuristically
- Assign each var randomly

Random strategy is recommended practice!

The Local Optima Problem

Gradient descent is incomplete. Problem? Can get caught in "local minima": soln exists, but all local moves lose



Cope with local minima to be at least *statistically* complete

Stochastic SAT Search

Weighted ISAMP: make *weighted random* choices to avoid repeats

Similar idea: make randomized choices in greedy local search!
Stochastic = from “random variable”

Noise Moves

Idea: when "stuck" in local minimum, move randomly. Problems:

- May bounce right back
- Hard to tell when *effectively* stuck

Solution: Occasionally move randomly *regardless*! (SAT: make random flip)

Simulated Annealing

Early stochastic local search idea, based on physics. Start randomly, become increasingly greedy with time.

- Works OK for a few things, but mostly not.
- Physics is wrong
- Theorem says perfect memory + constant noise better

Restarts

Problem with uniform noise:

- Large noise = no convergence
- Small noise = local minima

Want to explore many good local minima...

Restart: stop and pick a new random assignment

- Free parameter: must be tuned
- Careful tuning gives *rapid random restarts* (Selman + Gomes)

GSAT (Selman)

forever $\{$

    A := random total assignment to vars of F

    r times $\{$

        If A satisfies F, return A

        Choose most improving $v \in \mathbf{vars}(F)$

        $A := A \cdot [v \mapsto \mathbf{flip}(v)]$

    $\}$

$\}$

WalkSAT (Selman)

forever $\{$
    $A :=$ random total assignment to vars of F
    r times $\{$
        If A satisfies F, return A
        Select first of $\{$
            perfect $v \in \mathbf{vars}(F)$
            on coin flip, random v
            least damaging v
        $\}$
        $A := A \cdot [v \mapsto \mathbf{flip}(v)]$
    $\}$
$\}$

Tabu Search

Problem: even WalkSAT fails to "move far away"; sticks in short loops

Partial Solution: Use "Tabu" list to recall, avoid recently visited states

- Uses memory interestingly (unlike prev!)
- Only effective on certain problem classes

GENET (Tsang)

Learning architecture for control of stochastic search. Decides

- restarts
- noise

(sort of) by connectionist (simple neural) network

Tsang reports recent good results. I am largely unfamiliar with this work

Stochastic Search For Generic CSPs

Ideas are not just for SAT!

Choices:

- Reduce CSP to SAT problem
- Lift SAT algorithm to discrete vars

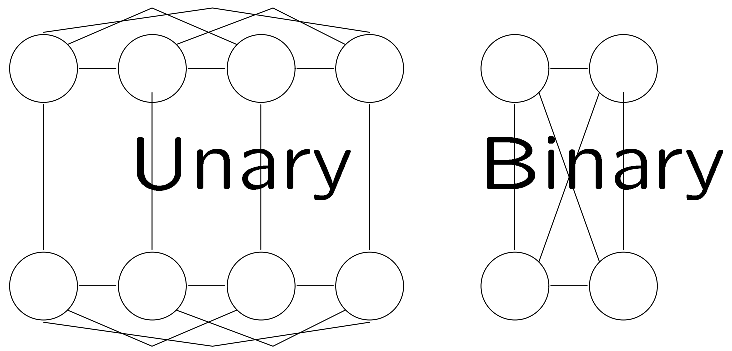
Both have been pursued extensively

CSP Reduction To SAT

CSP instance can be reduced to SAT instance by encoding values as bits (duh)

- Unary encoding
- Binary encoding

Consider graph 3-coloring:



Min-Conflicts (Johnston + Minton)

Direct local search on CSPs: like GSAT, but minimize number of violated constraints

Works well for scheduling, similar. Will examine later on. Good choice for OPTIMAL TENT ASSIGNMENT HW1c2!