

Local Search Techniques

Ruben Martins

**Carnegie
Mellon
University**

<http://www.cs.cmu.edu/~rubenm/15816-f25/>

Automated Reasoning and Satisfiability

September 22, 2025

Random k -SAT

Stochastic Local Search

WalkSAT and ProbSAT

Weight Transfer

Random k -SAT

Stochastic Local Search

WalkSAT and ProbSAT

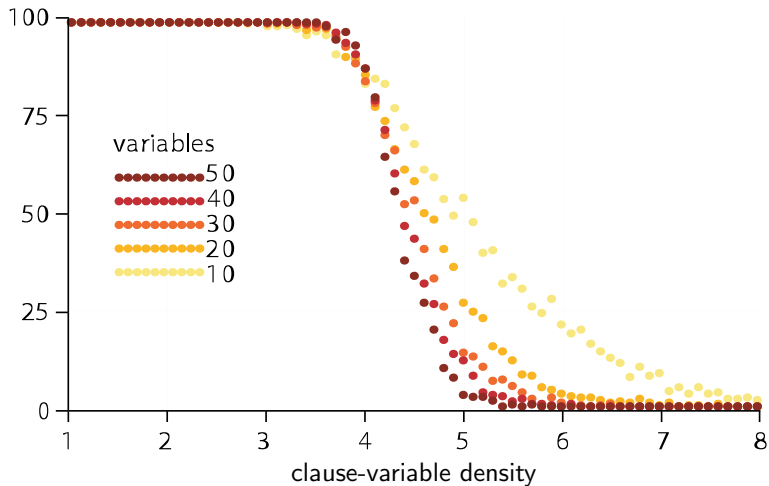
Weight Transfer

Random k-SAT

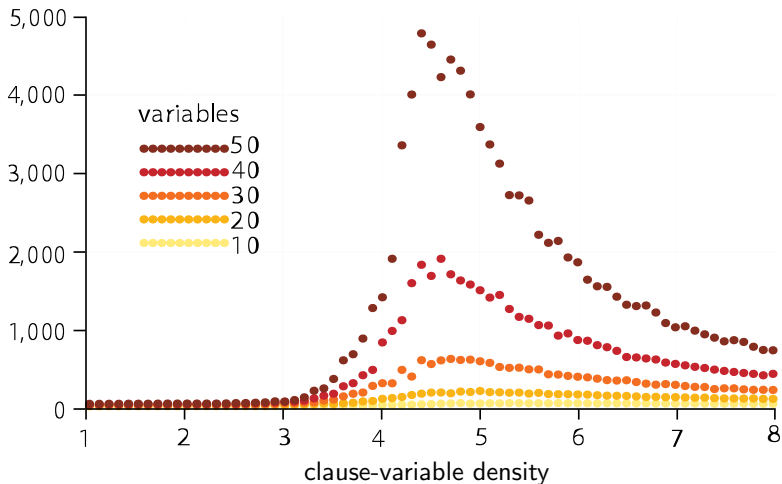
Local search solvers are particularly effective (and optimized) on hard uniform random (satisfiable) k-SAT problems

- All clauses have length k
- Variables have the same probability to occur
- Each literal is negated with probability of 50%
- Density is ratio Clauses to Variables

Random 3-SAT: % satisfiable, the phase transition



Random 3-SAT: exponential runtime, the threshold



Random k -SAT

Stochastic Local Search

WalkSAT and ProbSAT

Weight Transfer

Local Search: Generic structure

Generic structure of local search SAT solvers

```
1: for  $i$  in 1 to MAX_TRIES do  
2:    $\alpha :=$  random initial assignment  
3:   for  $j$  in 1 to MAX_STEPS do  
4:     if  $\alpha$  satisfies  $\Gamma$  then  
5:       return satisfiable  
6:      $\alpha :=$  Flip ( $\alpha$ )  
7: return unknown
```

Local Search: Global vs Local flips

Global flips

- Pro: Big improvements
- Neg: Probabilistic incomplete

Local flips

- Neg: Small improvements
- Pos: Probabilistic complete

Random k -SAT

Stochastic Local Search

WalkSAT and ProbSAT

Weight Transfer

Local Search: Types of Flips

Select a random unsatisfied clause C

- Free flip
- Random flip
- Heuristic flip

Local Search: WalkSAT Code [Selman, Kautz, and Cohen '93]

FLIP_WALKSAT (α)

- 1: $C :=$ **random** clause in Γ that is falsified by α
- 2: **if** a variable in C can be flipped for **free** **then**
- 3: flip in α **that** variable
- 4: **else**
- 5: flip in α with p a **random** $x_i \in C$
- 6: flip in α with $1 - p$ the **optimal** $x_i \in C$
- 7: **return** α

Local Search: ProbSAT [Balint and Schöning '12]

ProbSAT generalizes the WalkSAT code.

Let $\text{break}(x, \alpha)$ denote the number of clauses that are **only satisfied** by x or $\bar{x}$ under the assignment α

- $C :=$ random clause in Γ that is falsified by α
- randomly pick a variable x in C using **weights** $c^{-\text{break}(x, \alpha)}$
- an effective constant for random 3-SAT: $c = 2.5$
- update α by flipping x

Random k-SAT

Stochastic Local Search

WalkSAT and ProbSAT

Weight Transfer

Local Search: Weight Transfer

- All clauses have a **weight**
- Only do **global** flips
- Pick the variable that reduces the falsified weight the **most**
- If there is no weight-reducing variable, **modify the weights**

Local Search: Weight Transfer Pseudo-code

```
1: for  $i$  in 1 to MAX_TRIES do
2:    $\alpha :=$  random initial assignment
3:   for  $j$  in 1 to MAX_STEPS do
4:     if  $\alpha$  satisfies  $\Gamma$  then
5:       return satisfiable
6:     else if there exists a weight-reducing variable then
7:       flip the most weight-reducing variable in  $\alpha$ 
8:     else
9:       increase the weight of clauses falsified by  $\alpha$ 
10: return unknown
```

Weight Transfer Heuristics

Key heuristic: transfer weight from **neighboring** clauses

- Clauses are neighboring if they **share a literal**
- Transfer weight from **satisfied** to falsified clauses
- Transfer from **highest weight** satisfied neighboring clause

Weight Transfer Heuristics

Key heuristic: transfer weight from **neighboring** clauses

- Clauses are neighboring if they **share a literal**
- Transfer weight from **satisfied** to falsified clauses
- Transfer from **highest weight** satisfied neighboring clause

Divide and Distribute Fixed Weights (DDFW) heuristics

- Weight initialization $W(C) = w_0 = 8$
- Transfer weights if no weight-reducing variable to flip
- Transfer a weight of 1 if $W(C_{\text{satisfied}}) = w_0$
- Transfer a weight of 2 if $W(C_{\text{satisfied}}) > w_0$