An Introduction to Simulated Annealing

Kristian Guillaumier 2006, 07

Introduction

- Annealing process: heating of metals then cool them in a controlled way to form a minimum energy crystalline structure.
- Analogy with optimisation – to find the global minimum of some generic optimisation problem.

References

- Slide content from:
  - http://www.autonlab.org/tutorials/hillclimb02.pdf
  - http://www.geocities.com/francorbusetti/saweb.pdf
  - http://en.wikipedia.org/wiki/Simulated_annealing

An Analogy

by Franco Busetti [1]

- Imagine a ball that can bounce over mountains from valley to valley.
- Initially we have a high temperature which allows the ball to bounce higher – at this temperature we can reach any valley with enough bounces.
- As temperature declines, the ball cannot bounce so high so it is easier to be trapped in a valley (settle).
- A generating distribution (moveset) defines the next state/direction to go to from the current one.
- An acceptance distribution takes the optimality of the previous valley and the optimality of the suggested valley to probabilistically decide if the ball has to bounce to the suggested valley.

Standard Hill-Climbing

From Andrew W. Moore, Carnegie Mellon

- Let the MoveSet be the list of all moves possible from any current configuration.
- Attempt to Maximise Eval(X), where X is a solution and Eval is the fitness of that solution.
- Simple hill-climbing only requires us to move to the best solution from those possible from the MoveSet.
  - If all solutions are lower, then we are caught in a local maximum.

Standard Hill-Climbing

From Andrew W. Moore, Carnegie Mellon

1. X = Initial (possibly random) solution.
2. E = Eval(X).
3. For each possible move M from X in the MoveSet:
    1. X’ = MoveFromTo(X, M).
    2. E’ = Eval(X’).
4. If all E’ computed before < E:
    1. Local maximum – get out.
5. Else:
    1. X = Highest X’ encountered in For loop.
    2. E = Highest E’ encountered in For loop.
    3. Goto 3.

Standard Hill-Climbing

From Andrew W. Moore, Carnegie Mellon [2]

- Trivial to program.
- No need to understand the problem.
- Design of the MoveSet is critical (premature/never-ending convergence).
- Number of moves is very very large.

Randomised Hill-Climbing

From Andrew W. Moore, Carnegie Mellon [2]

1. X = Initial (possibly random) solution.
2. E = Eval(X).
3. M = Random Move From MoveSet.
4. X’ = MoveFromTo(X, M).
5. E’ = Eval(X’).
6. If E’ > E:
    1. X = X’
    2. E = E’
7. If Bored:
    1. Return.
8. Else:
    1. Goto 3.

Example

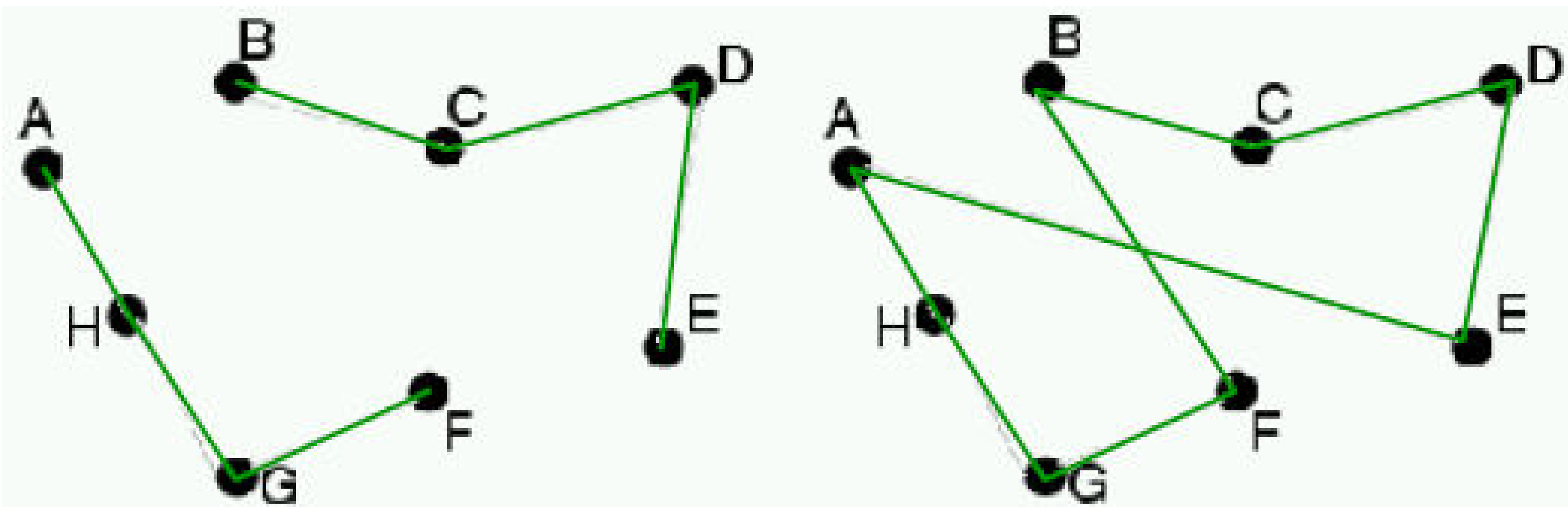
- Consider this constraint satisfaction problem from [2]:
  - `A V ~B V C`
  - `~A V C V D`
  - `B V D V ~E`
  - `~C V ~D V ~E`
  - `~A V ~C V E`
- Eval = number of satisfied clauses.
- MoveSet = Flip truth of any one variable.
- Example configuration (for the 5 vars) = {1,0,1,0,1}.
- Example result:
  - `A V ~B V C    : 1`
  - `~A V C V D    : 1`
  - `B V D V ~E    : 0`
  - `~C V ~D V ~E  : 1`
  - `~A V ~C V E   : 1`
- Eval = 4.

WalkSat

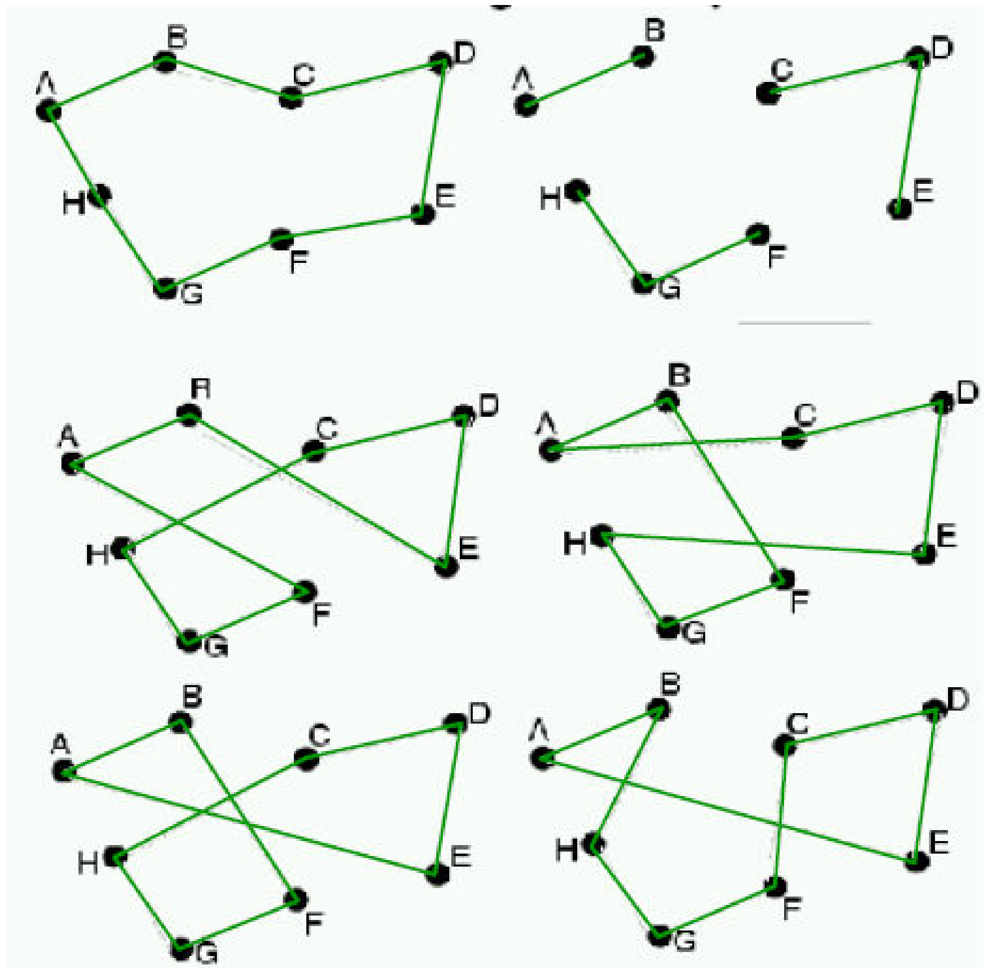
- Pick a random unsatisfied clause.
- Consider 3 moves: by flipping variables.
- If Eval improves:
  - Accept the best.
- Else:
  - 50% of time – pick least best.
  - 50% of time – pick random.

- This is actually the best known algorithm for satisfying Boolean formulae [2].

MoveSet for TSP: 2-Change



MoveSet for TSP: 3-Change



Simulated Annealing

1. X = Initial (possibly random) solution.
2. E = Eval(X).
3. M = Random move from MoveSet.
4. X’ = MoveFromTo(X, M).
5. E’ = Eval(X’).
6. If E’ > E Then:
    1. X = X’.
    2. E = E’
7. Else:
    1. With some probability (even though solution will be worse):
        1. X = X’.
        2. E = E’
8. Goto 3 unless bored.

Considerations

- Probability of choosing worse solution allows us to escape local maxima/minima.
- But which probability to choose?
  - A small one 0.1?
  - Decrease probability with time?
  - Decrease probability with time and as (E-E’) increases?

In Simulated Annealing

- If E’ > E then definitely accept the change.
- Otherwise accept with probability: exp(-(E-E’)/T_i), where
  - Exp is the exponential function (see Wikipedia).
  - T is time at instant i which is gradually decreasing.

Advantages and Disadvantages

- Can handle many constraints.
- General algorithm.
- Ability to approach the global optimum.
- It is a meta-heuristic – a lot of design choices are required.
- Tailoring.