Function Optimization with Local Search

Sven Koenig, USC

Russell and Norvig, 3rd Edition, Sections 4.1 and 4.2

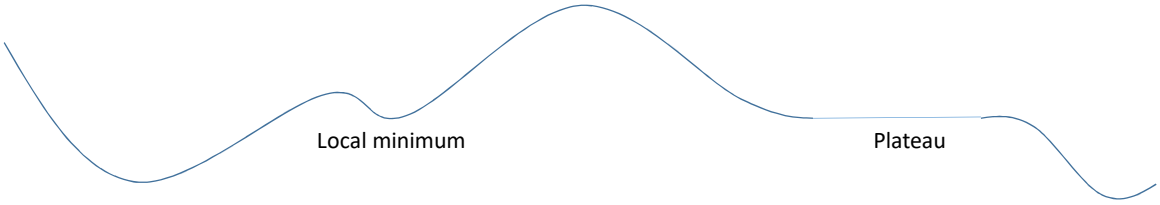
These slides are new and can contain mistakes and typos.
Please report them to Sven (skoenig@usc.edu).

Gradient Descent

- Finding a local minimum of a differentiable function $f(x_1, x_2, \ldots, x_n)$ with gradient descent (for a small positive learning rate α)
- Initialize $x_1, x_2, \ldots, x_n$ with random values
- Repeat until local minimum reached
  - For all x_i in parallel
    - $x_i := x_i - \alpha \, d f(x_1, x_2, \ldots, x_n) / d x_i$

Gradient Descent

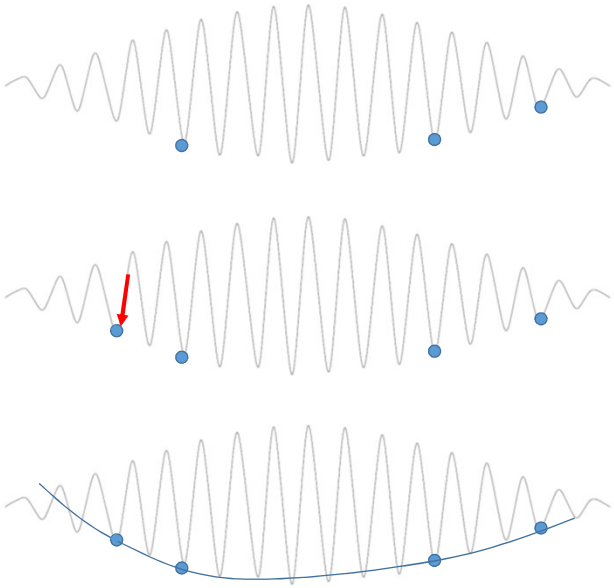
- Problems and solution approaches
  - Overshooting the local minimum: momentum term
  - Local minima: random restarts, simulated annealing, STAGE
  - Plateaus (one of the issues of threshold activation functions): random restarts
  - Ridges: momentum term



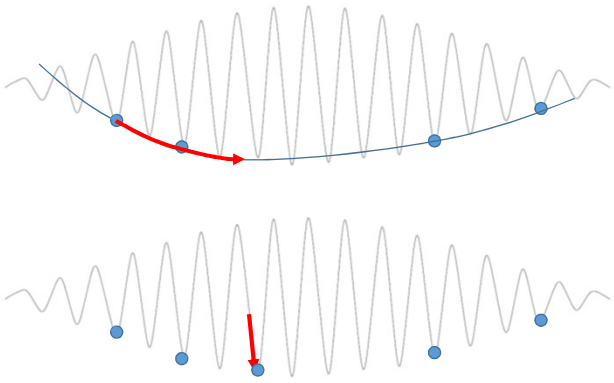
STAGE (by Boyan and Moore)

1. Use gradient descent with random restarts and remember all local minima
2. Estimate a function of the local minima
3. Stage 1: use the ending point of gradient descent on the given function as starting point for gradient descent on the function of the local minima
4. Stage 2: use the ending point of gradient descent on the function of the local minima as starting point for gradient descent on the given function
5. Go to 2.

STAGE



STAGE



Local Search for Function Optimization

- From now on
  - Function maximization instead of minimization (called gradient ascent or hillclimbing)
  - discrete rather than continuous functions

Hillclimbing

- Combine with random restarts

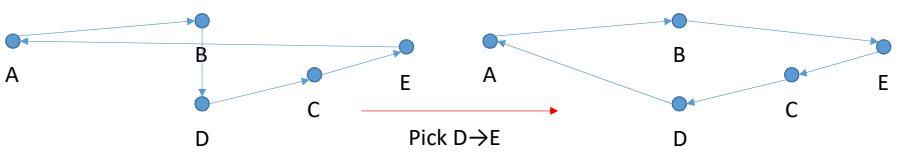
```
function HILL-CLIMBING(problem) returns a solution state
  inputs: problem, a problem
  static: current, a node
          next, a node

  current ← MAKE-NODE(INITIAL-STATE[problem])
  loop do
     next ← a highest-valued successor of current
     if VALUE[next] < VALUE[current] then return current
     current ← next
  end
```

Hillclimbing

- Example applications (typically NP-hard problems)
  - Map coloring: color all states of a given country with 4 colors so that no neighboring states have the same color
  - Boolean SATisfiability: find an interpretation that makes a given propositional sentence true
  - Traveling salesperson problem: visit all given cities in the plane with a shortest tour (= with the smallest round-trip distance)

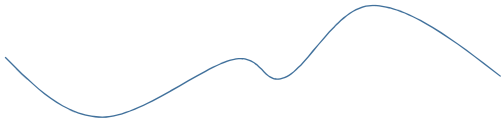
Hillclimbing

- Example applications
  - Map coloring: assign random colors to states, then repeatedly change the color of some state to decrease the number of neighboring states that have the same color
  - Boolean SATisfiability: transform the propositional sentence into conjunctive normal form, assign random truth values to all propositional symbols, then repeatedly switch the truth value of some symbol to decrease the number of clauses that evaluate to false
  - Traveling salesperson problem: pick a random tour, then repeatedly perform a "path reversal" for some pair of cities to shorten the tour (called two-opt algorithm)



Simulated Annealing

= Hillclimbing with going downhill from time to time



- Annealing: the process of gradually cooling a liquid until it freezes
  If the temperature is lowered sufficiently slowly, the material attains a lowest-energy (= perfect ordered) configuration.

```
function SIMULATED-ANNEALING( problem, schedule) returns a solution state
  inputs: problem, a problem
          schedule, a mapping from time to "temperature"
  static: current, a node
          next, a node
          T, a "temperature" controlling the probability of downward steps

  current ← MAKE-NODE(INITIAL-STATE[problem])
  for t ← 1 to ∞ do
      T ← schedule[t]
      if T=0 then return current
      next ← a randomly selected successor of current
      ΔE ← VALUE[next] – VALUE[current]
      if ΔE > 0 then current ← next
      else current ← next only with probability e^(ΔE/T)
```

VALUE[x] = total energy of the atoms in the material

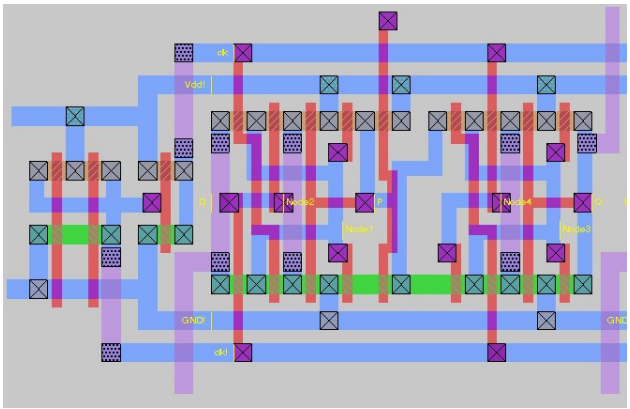
decrease the temperature (T) over time

Go downhill with a probability that is the higher the

- less one goes downhill (ΔE) and
- the fewer iterations (t) simulated annealing has run = the higher the temperature (T) is

Simulated Annealing

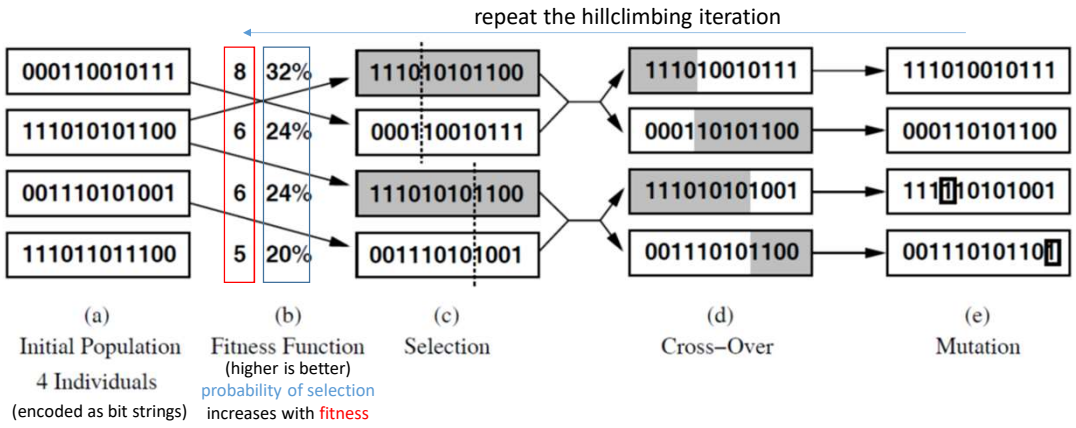
- Example application: VLSI layout



[www.eg.bucknell.edu/~ee342/recognition/flipflop.gif]

Genetic Algorithms

= Hillclimbing with parallel search and going downhill from time to time



Individuals are solutions

Individuals get to reproduce the more, the "fitter" they are (= the higher their solution quality is)

Children are "genetic mixtures" of their parents

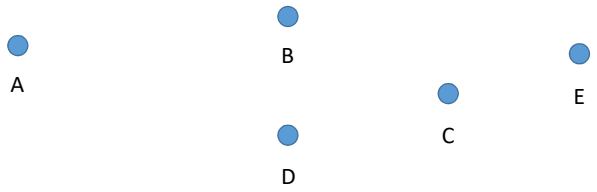
A small number of random mutations occur (to maintain diversity within the population)

Genetic Algorithms

- Problems and solution approaches
  - Forgetting good solutions: retain the best solutions
  - Good encodings are crucial: cross-over of solutions should lead to solutions most of the time, cross-over of good solutions should have a chance to lead to even better solutions

Genetic Algorithms

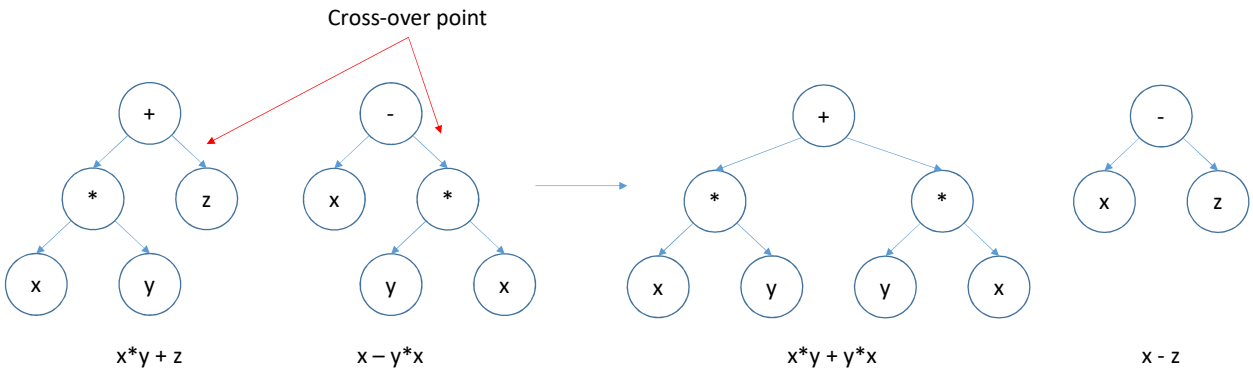
- "Genetic algorithms are the 3rd best way of doing just about anything"
- Example application: the traveling salesperson problem



- Is ABDCEA is good encoding for a tour? For example, what happens when one recombines ABD|CEA with BED|CAB?
- What is a better encoding for a tour?

Genetic Algorithms

- Example application: evolutionary programming



Local Search

- Want to play around with local search algorithms for constraint satisfaction?
- Go here: http://aispace.org/hill/