Optimization and Simulation
Simulation Exercise 5: Simulated Annealing

Bilal Farooq

May 14, 2012

The Exercise: Simulated Annealing

- 5.1: Simulated Annealing for smooth optimization
- 5.2: Simulated Annealing for combinatorial optimization problem [Travelling salesman problem]
  - 5.2.1: Solution proposal function
  - 5.2.2: Objective function
  - 5.2.3: Putting the pieces together

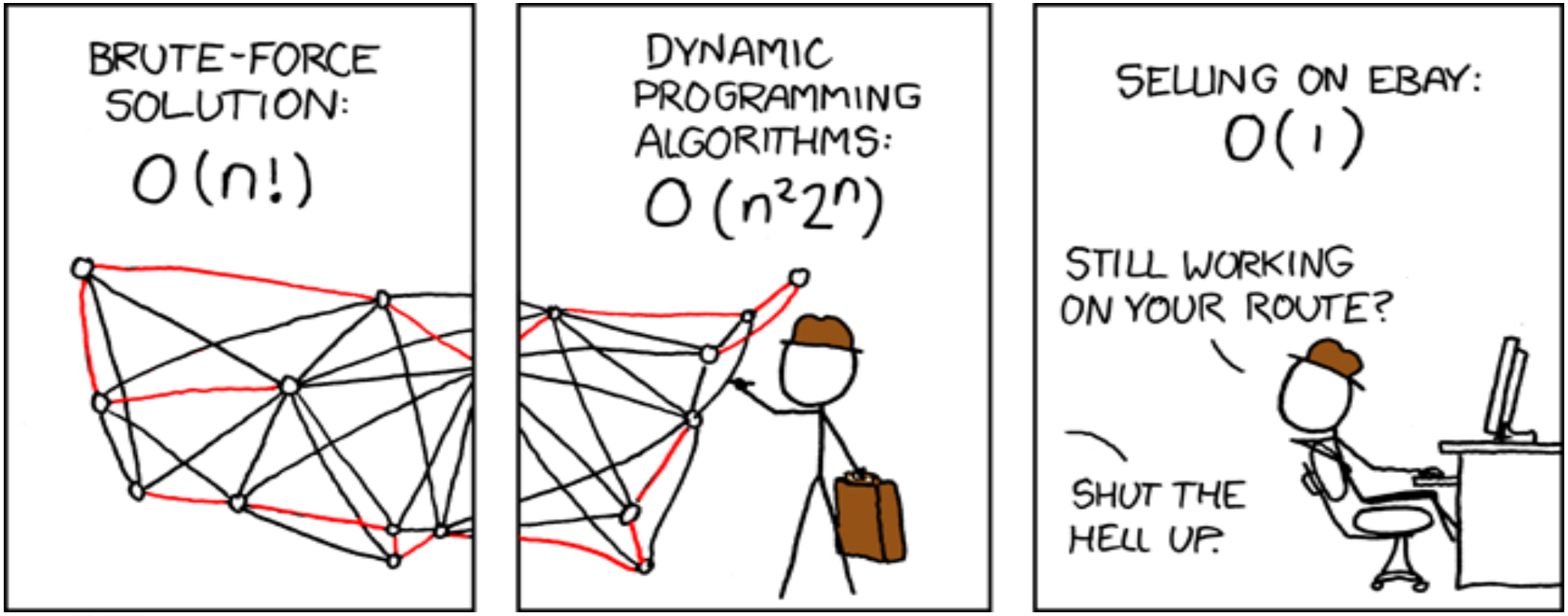
5.1 Simulated Annealing for Smooth Optimization

- Implement the function `SimulatedAnnealing.m` using the description
- Test your implementation with `SimulatedAnnealingTestSmooth.m`
  - Investigate the effect of different parameters
  - Try your own objective function

5.2 Travelling Salesman Problem

- A salesman has to visit *n* cities
  - He starts and ends his trip at his home city
  - Assume cost of travel to be total trip length
- What sequence of cities minimizes the travel cost?
  - Formulated in 1930s
  - Application: planning, logistics, manufacturing
  - NP-Hard problem
  - Approximate solutions

5.2 Travelling Salesman Problem



www.xkcd.com

5.2 Travelling Salesman Problem

- Cities are consecutively numbered: *1, 2,...,N*
- We encode solutions as x = $(x_1\ x_2\ \ldots\ x_N)$ where
  - x_i is the index of i^{th} city visited along the way
  - x_N is the last city visited before returning home
  - Every city must be visited exactly once

5.2.1 Solution Proposal Function

- Implement `RandomizeCitySequence.m`
- Test with `RandomizeCitySequenceTest.m`

5.2.2 Objective Function

- Implement `EvaluateCitySequence.m`
- Test with `EvaluateCitySequenceTest.m`

5.2.3 Putting the Pieces Together

- Run `SimulatedAnnealingTestTSP.m`
  - You already have the components ready
  - Try to reach the global optimum