Artificial Intelligence Programming

Local Search and Genetic Algorithms

Chris Brooks

Department of Computer Science

University of San Francisco

7-2: Overview

- Local Search - When is it useful?
- Hill-climbing search
- Simulated Annealing
- Genetic Algorithms

7-3: Local Search

- So far, the algorithms we've looked at store the entire path from initial state to goal state.
- This leads to memory/space issues on large problems.
- For some problems, path information is essential
  - Route finding
  - Water Jugs
  - 8-puzzle
  - The solution is the sequence of actions to take.
  - We know what the goal state is, but not how to reach it.

7-4: Local Search

- For other sorts of problems, we may not care what the sequence of actions is.
- CSPs fit this description.
  - Finding the optimal (or satisfactory) solution is what's important.
  - Scheduling
  - VLSI layout
  - Cryptography
  - Function optimization
  - Protein folding, gene sequencing
- The solution is an assignment of values to variables that maximizes some objective function.
- In these cases, we can safely discard at least some of the path information.

7-5: Local Search

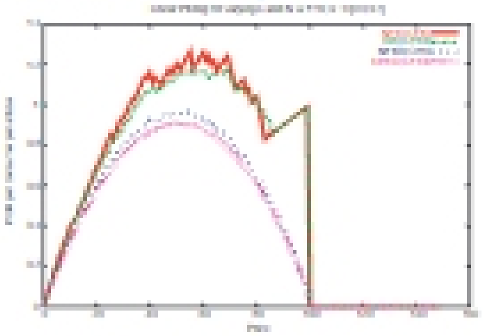
- A search algorithm that uses only the current state (as opposed to path information) is called a **local search** algorithm.
- Advantages:
  - Constant memory requirements
  - Can find solutions in incredibly large spaces.
- Disadvantages:
  - Hard to guarantee optimality; we might only find a local optimum
  - Lack of memory may cause us to revisit states or oscillate.

7-6: Search Landscape

- Local search is often useful for optimization problems
- "Find parameters such that o(x) is maximized/minimized"
- This is a search problem, where the state space is the combination of value assignments to parameters.
- If there are n parameters, we can imagine an $n + 1$ dimensional space, where the first n dimensions are the parameters of the function, and the $n + 1$th dimension is the **objective function**.
- We call this space a **search landscape**
  - Optima are on hills
  - Valleys are poor solutions.
  - (reverse this to minimize $o(x)$)

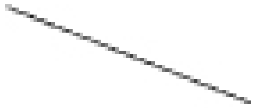
7-7: Search Landscape

- A one-dimensional landscape:



7-8: Search Landscape

- A two-dimensional landscape:



7-13: Improving hill-climbing

- Random-restart hill-climbing
- Run until an optimum is reached
- Randomly choose a new initial state
- Run again.
- After **n** iterations, keep best solution.
  - If we have a guess as to the number of optima, we can choose an **n**.

7-14: Simulated Annealing

- Hill-climbing's weakness is that it never moves 'downhill'
- Like greedy search, it can't "back up".
- Simulated annealing is an attempt to overcome this.
- "Bad" actions are occasionally chosen to move out of a local optimum.

7-15: Simulated Annealing

- Based on analogies to crystal formation.
- When a metal cools, lattices form as molecules fit into place.
- By reheating and recooling, a harder metal is formed
  - Small undoing leads to better solution.
  - Minimize the "energy" in the system
- Similarly, small steps away from the solution can help hill-climbing escape local optima.

7-16: Simulated Annealing