

# Local Search

## CS 121

### Lecture 4B

Chris Archibald

July 6, 2009

## 1 Local Search

### 1.1 Why?

Sometimes in search the path doesn't matter, just the goal. Examples

- 8-queens
- integrated circuit design
- factory-floor layout
- job-shop scheduling
- automatic programming
- telecommunications network optimization
- vehicle routing
- portfolio management

If we don't need to remember the path, we can consider other search methods.

### 1.2 Local search

Local search algorithms operate using a single current state and generally only move to neighbors of that state, paths are not retained advantages : little memory, usually a constant amount, also work in large state spaces

Can also work for pure optimization problems

## 2 state space landscape

A landscape has a location and an elevation (value of heuristic function or objective function) try to find the lowest valley, maximization then highest peak (negate to get the other)

Complete always finds goal. Optimal always finds global min/max/.

## 3 Hill-climbing search

Very simple:

```
start at initial state
get highest valued successor of current
if this is lower than current, return
else loop
```

### 3.1 Comments

Sometimes called greedy-local search

Often gets stuck (in 8-queens 86% of the time) (only 4 steps for success and 3 for failure)

- Local maxima
- Ridges

How to solve these problems?

Sideways moves? Need to be careful not to get stuck in a loop.

If we allow 100 sideways moves, now we can solve 94% of random 8-queens instances (21 success, 64 failure)

### 3.2 Variants of hill-climbing

#### 3.2.1 Stochastic hill climbing

Randomly choose successor with probability proportional to quality of that move

#### 3.2.2 First-choice hill climbing

Randomly generate successors, take first one that is better than current (good for thousands of successors)

All variants so far are incomplete.

## 4 Random-restart hill-climbing

Complete with probability approaching 1, since we will eventually generate the goal as an initial state

- Can solve 3-million queens in under a minute

- Effectiveness depends on the shape of the state space (porcupine state space is hard to do this in)

## 5 Simulated annealing search

Take random move if better, otherwise with some probability proportional to the change in value of the new state, and the “temperature” which decreases over time (analogy of ping pong ball bouncing around on a bumpy surface)

- With a schedule that lowers the temperature slowly enough, algorithm will find global optimum with probability approaching 1.

## 6 Local beam search

Let’s keep more than one node

- Keep  $k$  best, at each step keep  $k$  best out of all of them, until we reach a goal

- Differences to random restart:

- Useful information shared between searches)

### 6.1 Stochastic beam search

Choose  $k$  successors at random, with probability proportional to value

## 7 genetic algorithms

Variante of stochastic beam search in which successor states are generated by combining two parent states.

- Population =  $k$  states, individuals = state, represented as a string

- fitness function

- randomly choose some for reproduction, according to fitness (probabilistically)

- crossover point, mutation

### 7.1 Comments

Advantage in crossover, works if string representation is meaningful in groups, require very careful engineering of the representation