Simulated Annealing

Algorithm:

1. Pick a neighbor at random
   if it improves situation, switch to neighbor
   else accept with probability
   $\exp(-K(\Delta E) / \tau)$
   K = fudge factor
   (ΔE) is how much worse is neighbor than current
   τ = Temperature
2. Lower τ after doing above "several" times

Repeat (1 & 2 above) until you get a fixed answer.

We can show that if you choose the right schedule for lowering temperatures,

Local Search via Genetic Algorithms
(motivated by natural selection in biology)

To use, pick a metric for how good our solution.

Example: in 8 queens, we can use # of non-attacking pairs of queens.
Start with an initial population of 8 queens on a board.

2	4	7	4	5	8	8	2

(24 non attacking queen pairs)

(This box means that in row 1, there is 1 queen at column 2, at row 2 there is queen at column 7, etc…)
For each row where queen is (we have number in table above)
Choose at random among remainder to get pairs to breed.
Choose a random cross-over point.

3	2	7	5	2	4	1	1

2	4	7	4	5	8	8	2

This creates two new boxes with the crossover boxes interchanged (put the 45882 sequence with the 327 from the other, and vice-versa).

3	2	7	4	5	8	8	2
2	4	7	5	2	4	1	1

Then mutate locations with a certain probability

3	2	7	4	1	8	8	2

Adversarial Search

Will consider multiagent environment where agents can compete or cooperate on goals. If the agents compete on conflicting goals, get an adversarial search problem.

Adversarial search problems are usually called games.

We will consider games that involve turn taking, have two players, and are zero sum with perfect information.

Perfect information = deterministic & environment visible to all playing.

Zero sum = if Player 1 gets payoff (+a), then player 2 gets payoff (-a).

Optimal decisions in games

Call our two players Max and Min.
A game consists of

- An initial State which includes board positions and identifies player who moves next.
- A successor function which returns a list of (move, state) pairs indicating legal next moves + boards. (gives you the move, and the board you get to after doing that move)
- A terminal test that determines when the game is over.
- A utility test that gives a numerical value to terminal positions.