Local Search (Ch. 4.1-4.3)
Announcements

Office hours change:

Shneka - NOW = Thursday 2:30 - 3:30
Before = Thursday 1:30 - 3:30

Jangyoon - NOW = Friday 2:30 - 3:30
(Johnny) Before = Friday 2:00 - 3:00
Review: Genetic algorithms

Genetic algorithms have two parts:
1. Fitness rating for survival
2. Mix-and-match reproduction (with random changes occasionally)

Genetic algorithms are slow but tend to find optimal solution to fitness rating (goal)

Can get stuck if pool's diversity becomes too little (depend on random changes)
Review: Genetic algorithms

Let's make a small (fake) example with the 4-queens problem

Adults:

Child pool (fitness):

(20) = (30)

(10) = (20)

(15) = (30)
Let's make a small (fake) example with the 4-queens problem.

Child pool (fitness):

- (20)
- (10)
- (15)

Weighted random selection:

- (30)
- (20)
- (35)
Newton's method

Newton's method finds roots by pretending that the function is linear and looking where it would reach zero.

Idea:
1. Find $m =$ slope (derivative) at current $x_0$
2. Solve $x_1 = x_0 - y_0 / m$
3. Repeat 1 with $x_1, x_2, x_3...$ until converge
Newton's method

Despite the linear approximation, the method converges very quickly on (most) functions (see: newton.cpp)
Newton's method

Let \( f(x) = 3 \cdot x^2 + x \cdot \sin(x) + e^x - 10 \)

Then \( f'(x) = 6 \cdot x + \sin(x) + x \cdot \cos(x) + e^x \)

Start at... \( x_0 = 5 \) (random choice)

\( x_1 = 5 - \frac{208.619}{178.873} = 3.8337 \)
\( x_2 = 3.8337 - \frac{77.8787}{65.6459} = 2.64736 \)
\( x_3 = 2.64736 - \frac{26.398}{28.1446} = 1.70942 \)
\( x_4 = 1.70942 - \frac{5.98508}{16.5364} = 1.34748 \)
\( x_5 = 1.34748 - \frac{0.608903}{13.2062} = 1.30138 \)
\( x_6 = 1.30138 - \frac{0.00953223}{12.7929} = 1.30063 \)
\( x_7 = 1.30063 - \frac{2.48511e-06}{12.7863} = 1.30063 \)
Newton's method
You do Newton's method on:
\[ f(x) = x^3 - 3x + 1 \]
\[ f'(x) = 3x^2 - 3 \]

You pick \( x_0 \) (please not a root!)
Newton's method

Newton's method issues:
Sometimes does not find root

\[ f(x) = x^{1/3} \]

(Sometimes divide by zero)

\[ f(x) = x^2/(x+1) \]
Planning & uncertainty

Let's go back to the simple vacuum:

Except we will change the rules as such: When the agent sucks, there is a 20% chance the adjacent square will be cleaned too, also if no dirt on current, suck makes dirt 30% of time.
Planning & uncertainty

The outcome of sucking action is now stochastic (how do we build a search tree?)

To represent this we use **AND-OR trees** and there will be two types of actions:
1. Stochastic (AND nodes), goal must be in all sub-branches
2. Deterministic (OR nodes) goal needs only on one of the sub-branches
Planning & uncertainty

AND Node

OR Node

20% 80% 70% 30%
If we again modify the rules (suck reverted): Floor are slippery and agent only moves successfully 60% of the time (failure to move does not change state).

This situation is slightly more difficult as there is a chance the agent is never able to get off the starting state (constantly slips and fails to move).
Planning & uncertainty
Planning & uncertainty

Draw the search space for this agent

Agent movement:
80% forward
10% left
10% right
Planning & uncertainty

If there is a pit “P”, that deducts points when you land in it, what is the optimal path...
P = 2?
P = 20000?

Agent movement:
(all cost 1)
80% forward
10% left
10% right