What is AI?

Think like a human
Cognitive Modeling

Think rationally
Logic-based Systems

Act like a human
Turing Test

Act rationally
Rational Agents

Next couple of weeks

Solve the maze!
Solve the maze!

How did you figure it out?

One approach

What now?
One approach

Three choices

Pick one!
What now?

Still three options!
Which would you explore/pick?

Most people go down a single path until they realize that it’s wrong
One approach

Keep exploring

One approach

Keep exploring

One approach

What now?

One approach

Are we stuck?
No. Red positions are just possible options we haven’t explored
One approach

How do we know not to go left?

One approach

Have to be careful and keep track of where we’ve been if we can loop

One approach

Now what?

One approach

Now what?
One approach

Search problems

What information do we need to figure out a solution?

Search problems

Where to start

Where to finish (goal)

What the “world” (in this case a maze) looks like
- We’ll define the world as a collection of discrete states
- States are connected if we can get from one state to another by taking a particular action
- This is called the “state space”
For a given problem, still could have different state-spaces

How many more states are there?
Solving the maze

How what?

Could we have found it any other way?

Search algorithm

Keep track of a list of states that we could visit, we’ll call it “to_visit”

General idea:
- take a state off the to_visit list
- if it’s the goal state
  - we’re done!
- if it’s not the goal state
  - Add all of the successive states to the to_visit list
  - repeat

How do we start?
- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat

- take a state off the to_visit list
  - if it's the goal state
    we're done!
  - if it's not the goal state
    Add all of the successive states to the to_visit list
 repeat
- take a state off the to_visit list
- if it's the goal state
  we're done!
- if it's not the goal state
  Add all of the successive states to the
to_visit list
repeat

Which one?

- take a state off the to_visit list
- if it's the goal state
  we're done!
- if it's not the goal state
  Add all of the successive states to the
to_visit list
repeat

Is it a goal state?

- take a state off the to_visit list
- if it's the goal state
  we're done!
- if it's not the goal state
  Add all of the successive states to the
to_visit list
repeat

Where should we add
them in the list?

- take a state off the to_visit list
- if it's the goal state
  we're done!
- if it's not the goal state
  Add all of the successive states to the
to_visit list
repeat

Let's add them to the front
What do we do here?

to_visit

- take a state off the to_visit list
- if it's the goal state
  - we're done!
- if it's not the goal state
  Add all of the successive states to the to_visit list
repeat

//

List keeps track of where to go next, i.e. the states we know about but haven't explored.
- take a state off the to_visit list
- if it's the goal state
  we're done!
- if it's not the goal state
  Add all of the successive states to the to_visit list
  repeat

What type of structure/list is the to_visit list?
It's a stack!! (LIFO)
Search algorithms

add the start state to to_visit

Repeat
- take a state off the to_visit list
- if it's the goal state
  - we're done!
- if it's not the goal state
  - Add all of the successive states to the to_visit list
  - repeat

What would happen if it was a queue?

- Depth first search (DFS): to_visit is a stack
- Breadth first search (BFS): to_visit is a queue

N-queens problem

Place N queens on an N by N chess board such that none of the N queens are attacking any other queen.

Solution(s)?
N-queens problem
Place N queens on an N by N chess board such that none of the N queens are attacking any other queen.

Search algorithm
add the start state to to_visit

Repeat
- take a state off the to_visit list
  - if it's the goal state
    - we're done!
  - if it's not the goal state
    - What states can I get to from the current state?
      - Add all of the successive states to the to_visit list

Any problem that we can define these three things can be plugged into the search algorithm!
N queens problem

http://en.wikipedia.org/wiki/Eight_queens_puzzle