15-780: Graduate AI

Lecture 1. Intro & Search

Geoff Gordon (this lecture)
Ziv Bar-Joseph
TAs Geoff Hollinger, Henry Lin

Admin

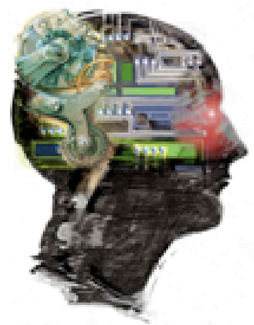


Image from USA Today

15-780: Graduate Artificial Intelligence, Fall 2007

Home

Schedule

Handouts

News

Links

Course Overview

Lectures	Tue. & Thu. 10:30 AM - 11:50 AM in **Wean Hall 5409**
IMPORTANT	Please periodically check the News page for important updates.

This course is targeted at graduate students who need to learn about current-day research, and about how to perform current-day research, in Artificial Intelligence---the discipline of designing intelligent decision-making machines.

Techniques from Probability, Statistics, Economics, Algorithms, Operations Research and Optimal Control are increasingly important tools for improving the intelligence and autonomy of machines, whether those machines are robots surveying Antarctica, schedulers moving billions of dollars of inventory, spacecraft deciding which experiments to perform, or vehicles negotiating for lanes on the freeway. This AI course is a review of a selected set of these tools. The course will cover the ideas underlying these tools, their implementation, and how to use them or extend them in your research.

Website highlights

- Book: Russell and Norvig. *Artificial Intelligence: A Modern Approach*, 2nd ed.
- Grading
- Final project
- Office hours

Background

- *No prerequisites*
- *But, suggest familiarity with at least some of the following:*
  - *Linear algebra*
  - *Calculus*
  - *Algorithms & data structures*
  - *Complexity theory*

Waitlist, Audits

- *If you need us to approve something, send us email*

Course email list

- *Send an email to thlin at cs to be included on course announcement list*

Matlab

- *Should all have access to Matlab via school computers*
  - *Those with access to CS license servers, please use if possible*
  - *Limited number of Andrew licenses*
- *Tutorial a week from today*

Intro

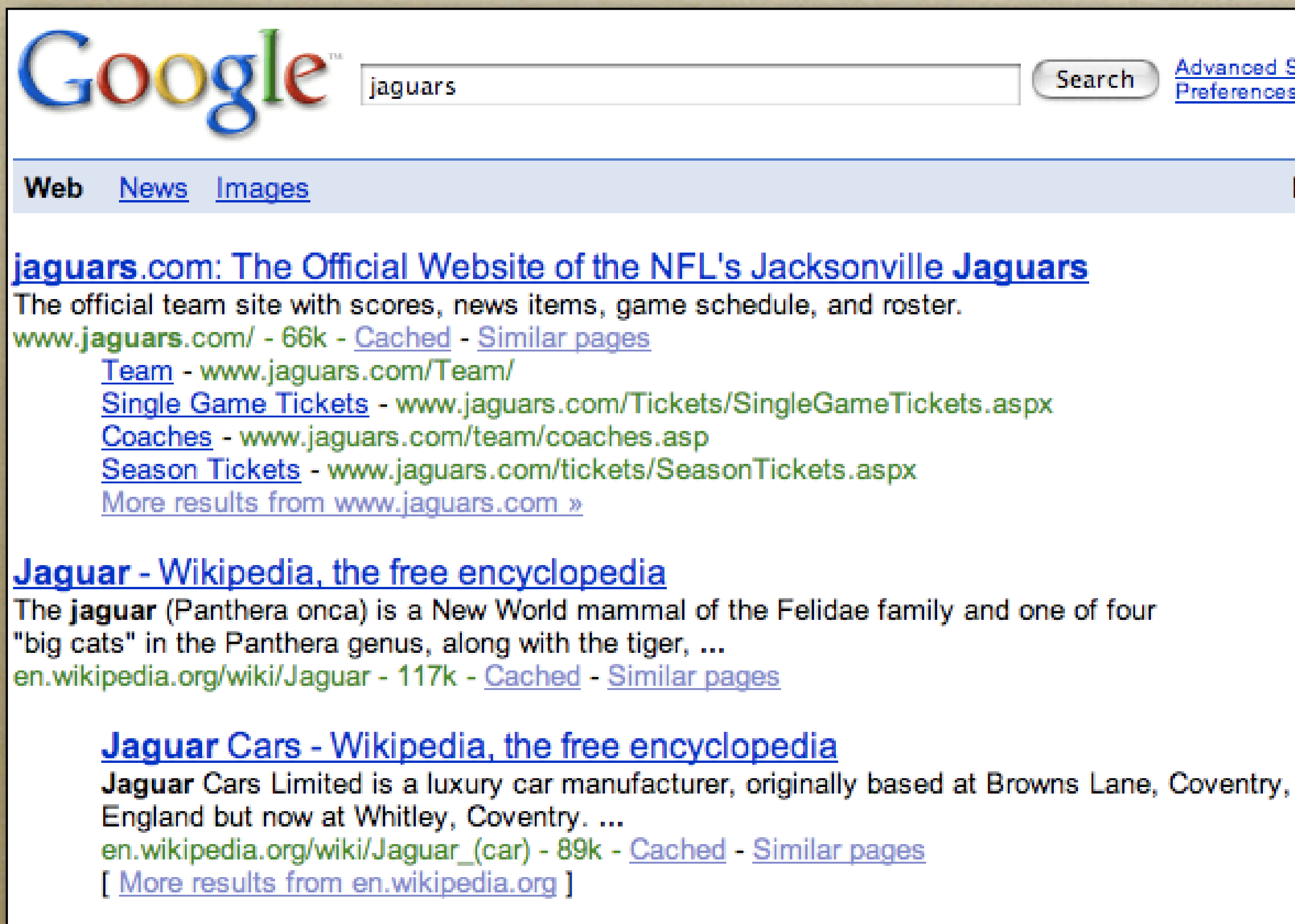
What is AI?

- *Easy part: A*
- *Hard part: I*
  - *Anything we don't know how to make a computer do yet*
  - *Corollary: once we do it, it isn't AI anymore :-)*

Definition by examples

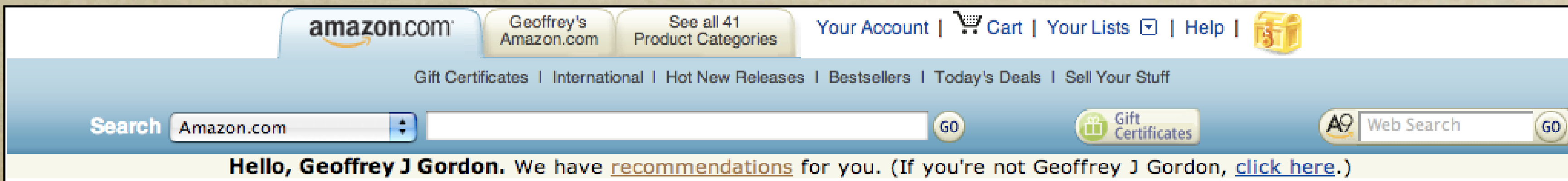
- *Board games*
  - *Deep Blue*
  - *TD-Gammon*
  - *Samuels's checkers player*
- *Card games*
  - *Poker*
  - *Bridge*

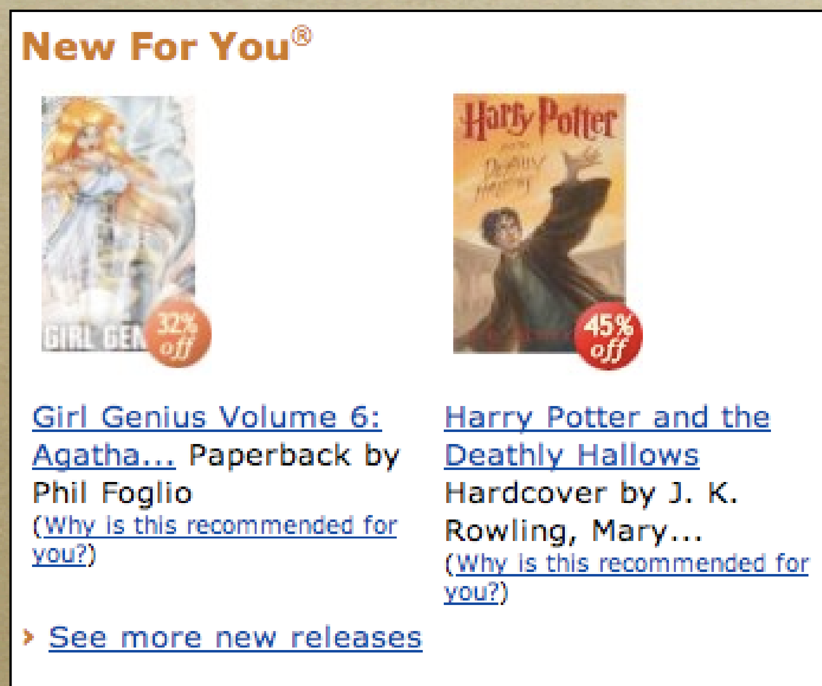
Web search


Google
jaguars
Search
Advanced S
Preferences
Web News Images
jaguars.com: The Official Website of the NFL's Jacksonville Jaguars
The official team site with scores, news items, game schedule, and roster.
www.jaguars.com/ - 66k - Cached - Similar pages
Team - www.jaguars.com/Team/
Single Game Tickets - www.jaguars.com/Tickets/SingleGameTickets.aspx
Coaches - www.jaguars.com/team/coaches.asp
Season Tickets - www.jaguars.com/tickets/SeasonTickets.aspx
More results from www.jaguars.com »
Jaguar - Wikipedia, the free encyclopedia
The jaguar (Panthera onca) is a New World mammal of the Felidae family and one of four "big cats" in the Panthera genus, along with the tiger, ...
en.wikipedia.org/wiki/Jaguar - 117k - Cached - Similar pages
Jaguar Cars - Wikipedia, the free encyclopedia
Jaguar Cars Limited is a luxury car manufacturer, originally based at Browns Lane, Coventry, England but now at Whitley, Coventry. ...
en.wikipedia.org/wiki/Jaguar_(car) - 89k - Cached - Similar pages
[More results from en.wikipedia.org]

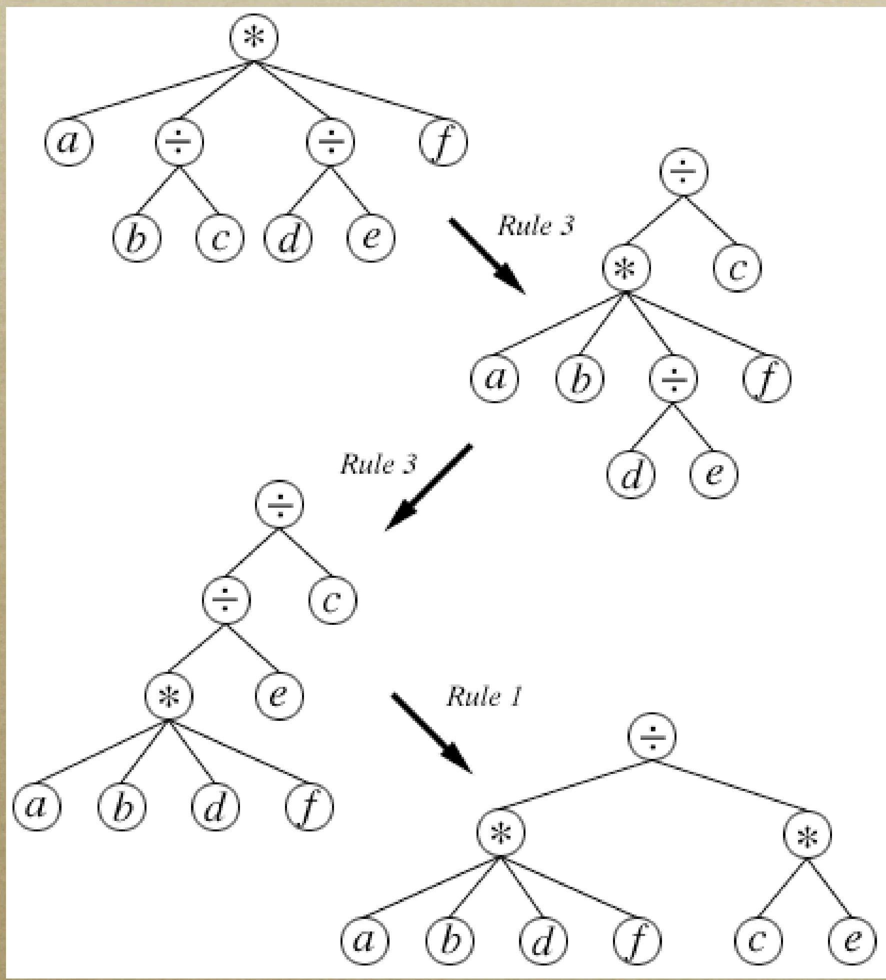
Web search, cont'd

Recommender systems





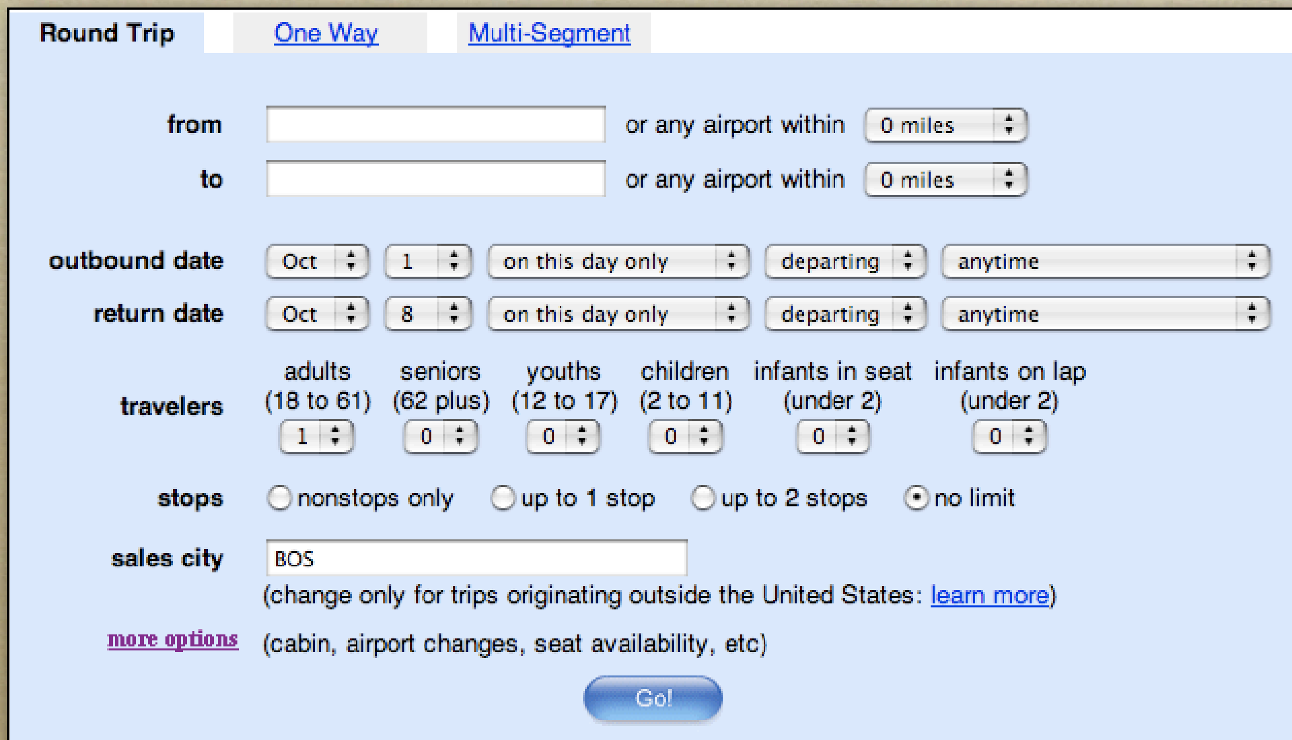
Computer algebra systems



from http://www.math.wpi.edu/IQP/BVCalcHist/calctoc.html

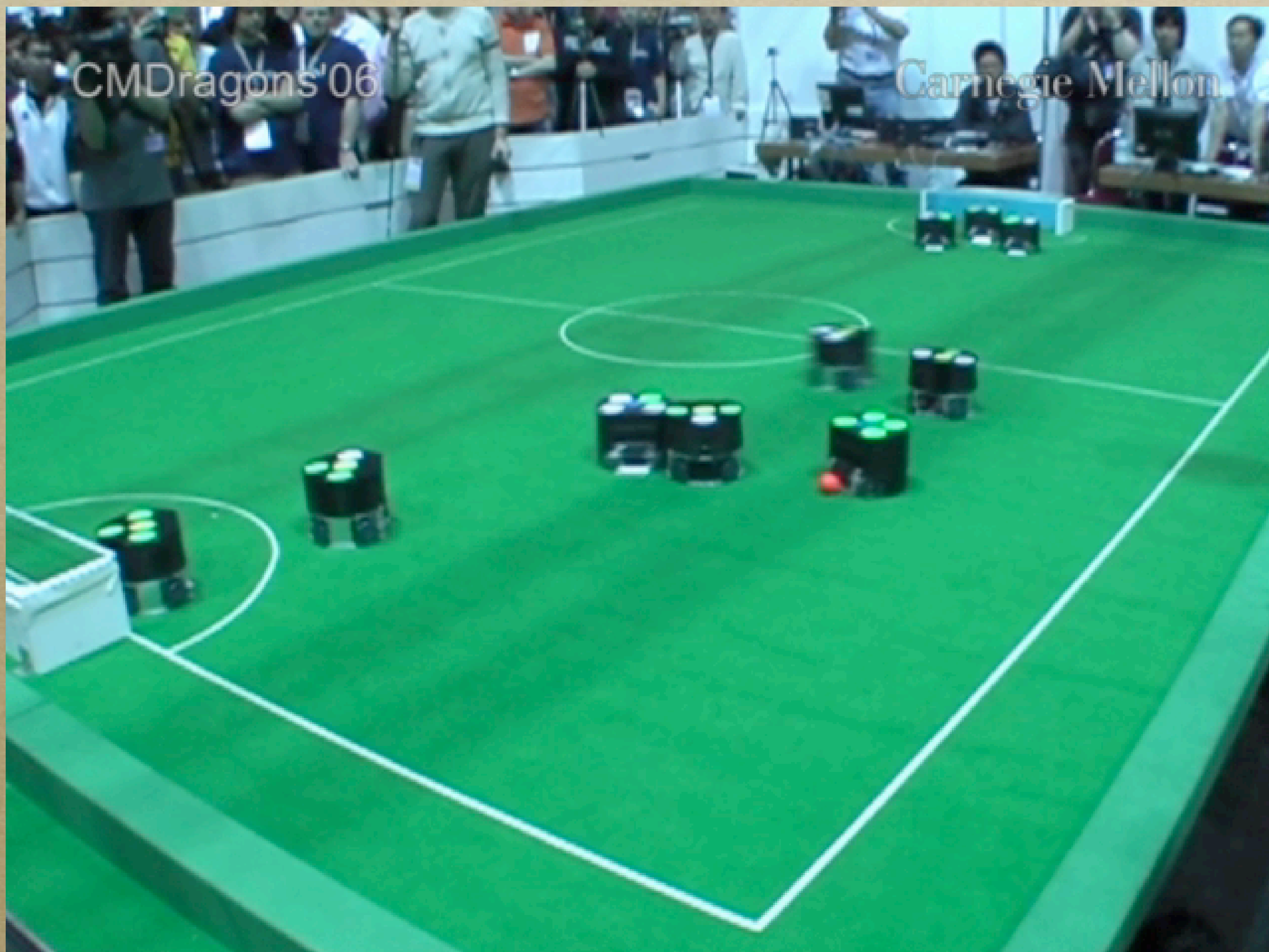
Grand Challenge road race

Getting from A to B



- *ITA software (http://beta.itasoftware.com)*

Robocup

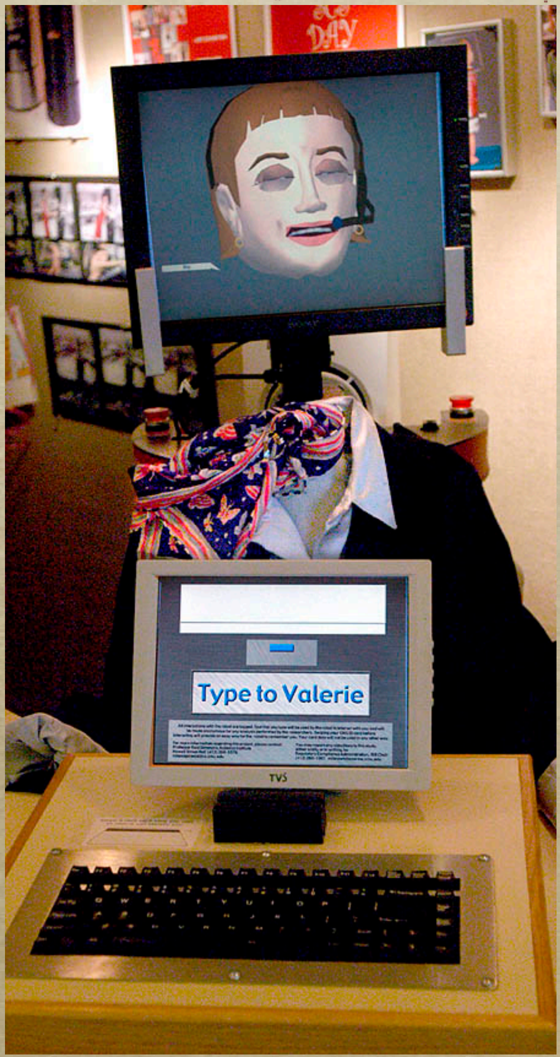


More examples

- *Motor skills: riding a bicycle, learning to walk, playing pool, …*
- *Vision*

More examples

- *Valerie and Tank, the Roboceptionists*
- *Social skills: attending a party, giving directions, …*



More examples

- *Natural language*
- *Speech recognition*

Common threads

- *Search and optimization*
    - *Set the problem up well (so that we can apply a standard algorithm)*
- *Managing uncertainty*
    - *The more different types of uncertainty, the harder the problem (and the slower the solution)*

Sources of uncertainty

- *Classic AI: no uncertainty, pure search*
  - *Mathematica*
  - *deterministic planning*
- *This is the topic of Part I of the course*

Opponents cause uncertainty

- *In chess, must guess what opponent will do; cannot directly control him/her*
- *Alternating moves: game trees (Part I)*
- *Simultaneous or hidden moves: game theory (Part III; computationally harder, especially if a sequence of moves)*

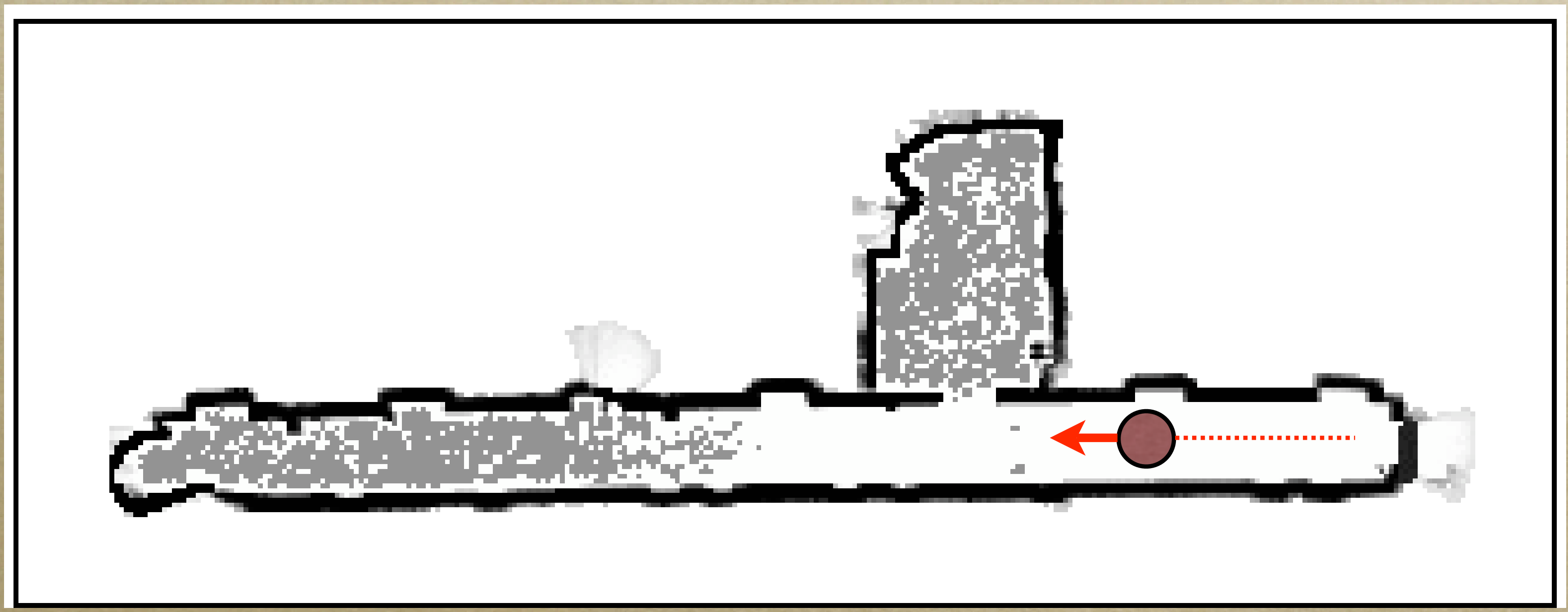
Outcome uncertainty

- *In backgammon, don't know ahead of time what the dice will show*
- *When driving down a corridor, wheel slippage causes unexpected deviations*
- *Open a door, find out what's behind it*
- *MDPs (Part II)*

Sensor uncertainty

- *Image of a handwritten digit → 0, 1, …, 9*
- *Image of room → person locations, identities*
- *Laser rangefinder scan of a corridor → map, robot location*

Sensor uncertainty example



Sensor uncertainty

- *For given set of immediate measurements, multiple states may be possible*
- *State = "everything we know about the world"*
- *More in Part II*

Combining sensor and outcome uncertainty

- *Build a robotic mouse*
- *Lives in cage with levers, blinky lights, etc.*
- *Pressing levers in the right sequence dispenses a snack of robo-cheese*
- *Move around, experiment w/ levers to turn on lights, get robo-cheese*
- *This is a POMDP (more in Part II)*

Other agents cause uncertainty

- *In many AI problems, there are other agents who aren’t (necessarily) opponents*
  - *Ignore them & pretend part of Nature*
  - *Assume they’re opponents (pessimistic)*
  - *Learn to cope with what they do*
  - *Try to convince them to cooperate (paradoxically, this is the hardest case)*
- *Part III*

Search

How to build a robotic grad student

- *Grad AI: progress for graduation 4, time 4*
- *Wine tasting: progress 1, time 2*
- *Nonlinear Frobnitz Dynamics: progress 5, time 9*

Constraints

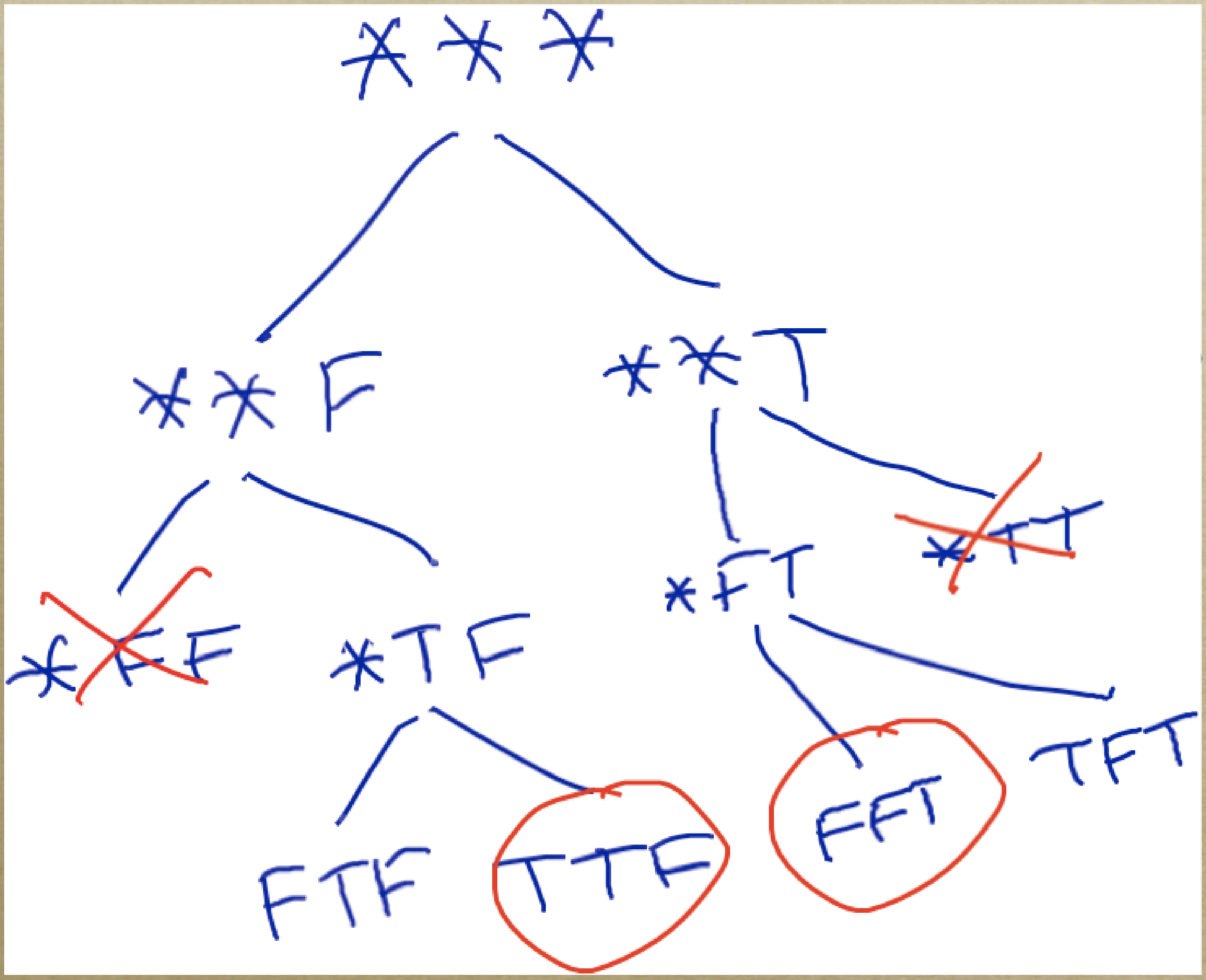
- *Must take courses w/ ttl progress* ≥ 5
- *Total time* ≤ 10

Solution by enumeration

Can we do better?

- *What about partial state (*, T, T)?*

Search graph

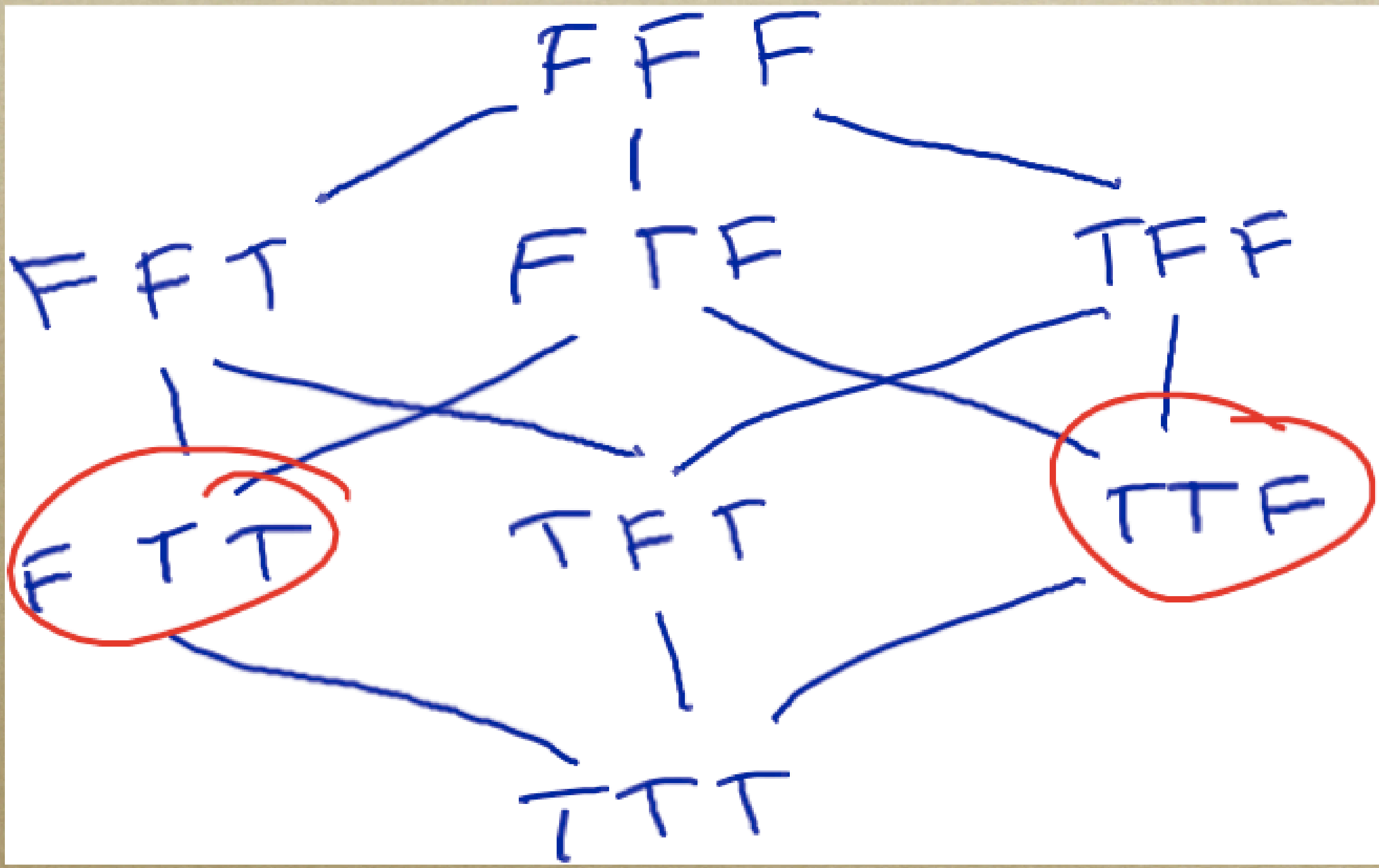


Node: ***, **F, **T, *F*, *FF, *FT, …

Search graph

- *Node: solution or partial solution*
- *Neighbor generating function*
- *Solution test = yes, no, maybe*

Alternate search graph



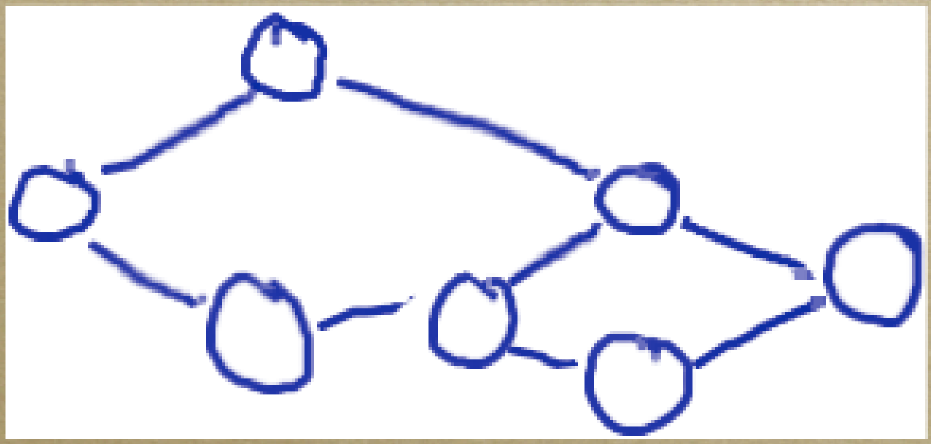
- *Nodes: FFF, FFT, FTF, FTT, …*

A node can be anything

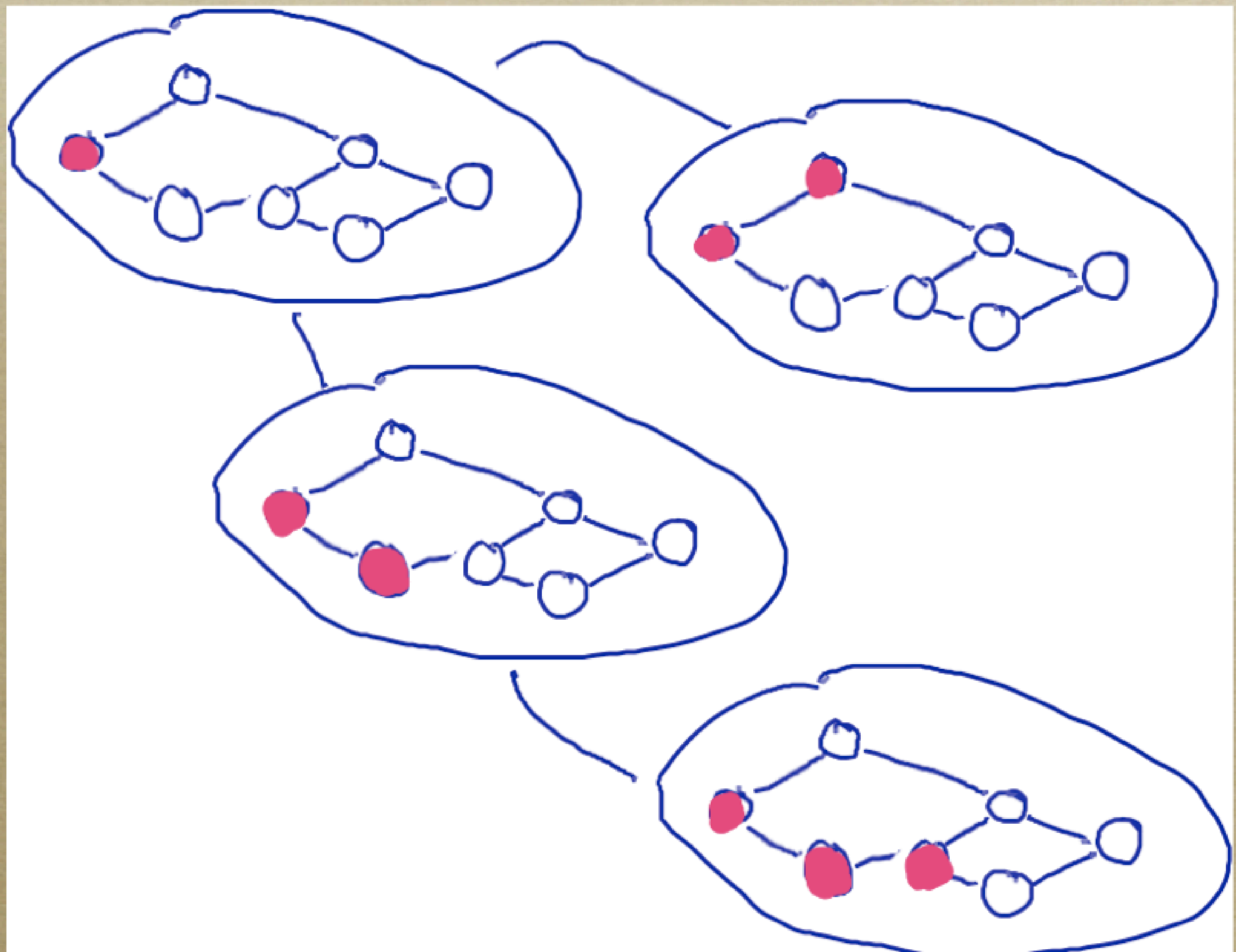
- *A list of variable settings*
- *A mathematical formula*
- *A set of flights that go from PIT to LAX*
- *A graph*

When a node is a graph

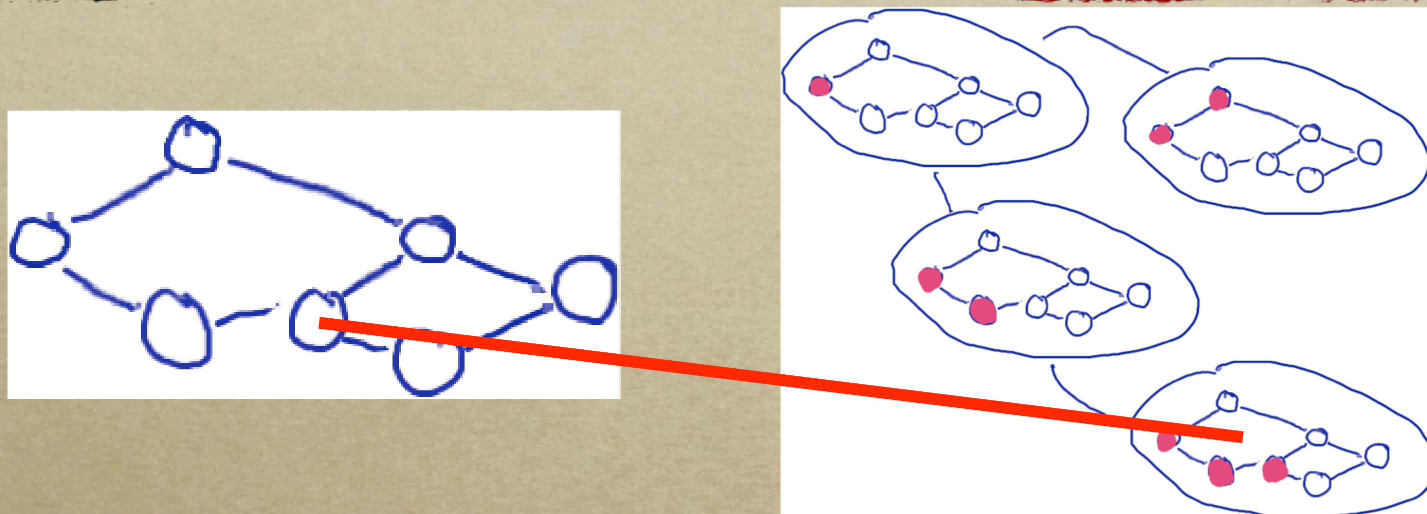
- *Not to be confused with search graph*
- *E.g., a (partial) matching, a (partial) spanning tree, or a (partial) path*



Search graph for shortest path



Isomorphism



- *For path planning, if we prune non-shortest paths, search graph is isomorphic to original graph*
- *Node X in original graph ≡ shortest path from start to X*

Generic search

$S = \{$ *some set of nodes* $\}$ $\quad M = \emptyset$

While $(S \neq \emptyset)$

$x \leftarrow$ *some element of* S, $\; S \leftarrow S \setminus x$

$M = M \cup \{x\}$

if $(solution(x) = Y)$ *return* x

if $(solution(x) = N)$ *continue*

$S = S \cup (neighbors(x) \setminus M)$

can be approximate

Choices

- *Where to start?*
- *Which element of S to visit next?*
- *How much effort to maintain S, M?*

Shortcuts for open, visited list

- *Open list:*
  - *Throw away some elements?*
  - *Sort key?*
- *Visited list:*
  - *Just don't return to parent*
  - *Keep nodes in path from start to X*
  - *Keep all nodes*

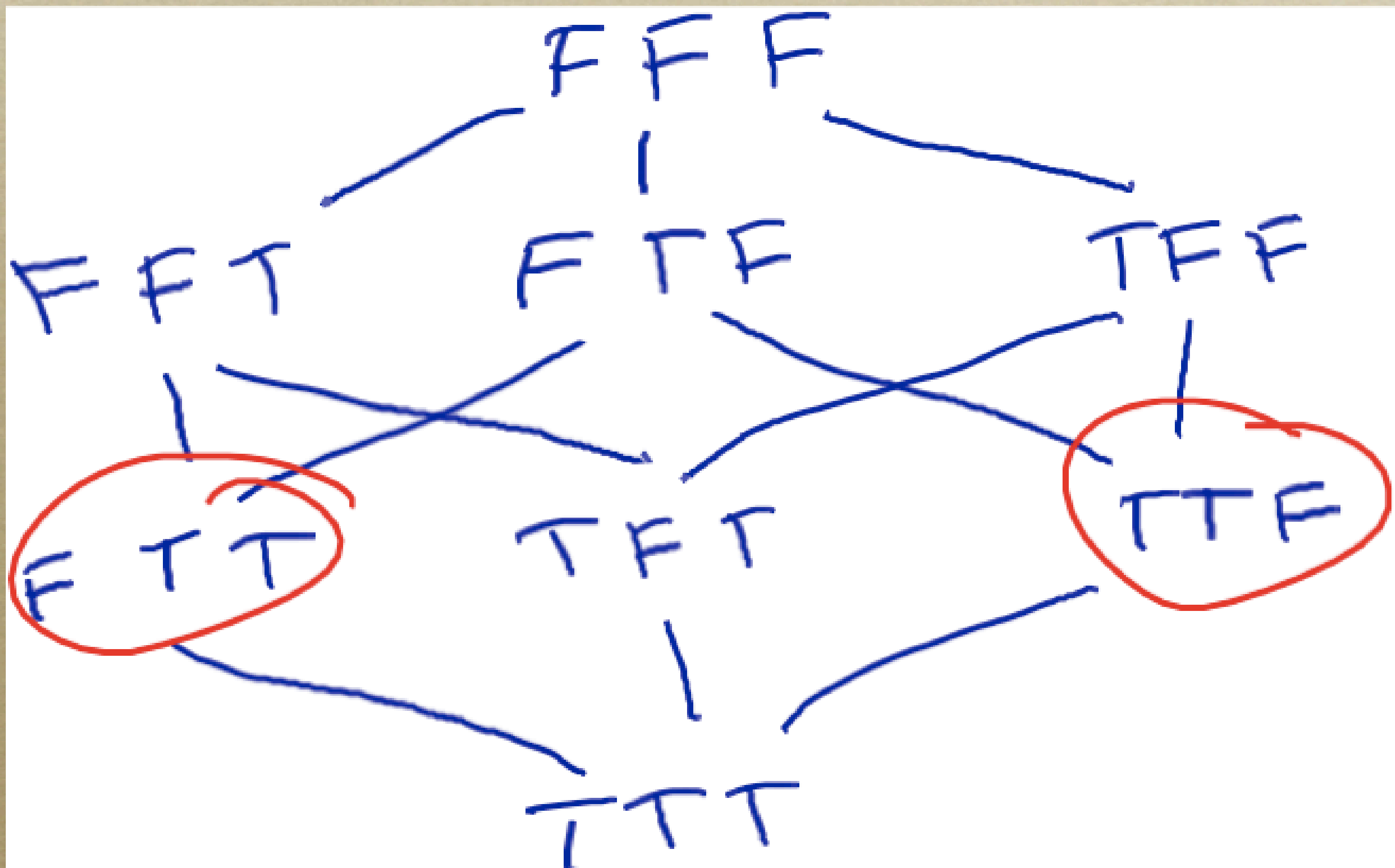
Data structures: M

- *Need insert, membership test*
  - *hash table (expense of equality test?)*
  - *avoid M altogether using node ordering*
    - *only insert neighbors y > x into S*

Data structures: S

- *For S: need insert, pop*
  - *LIFO (stack)*
  - *FIFO (queue)*
  - *priority queue (choice of sort key)*

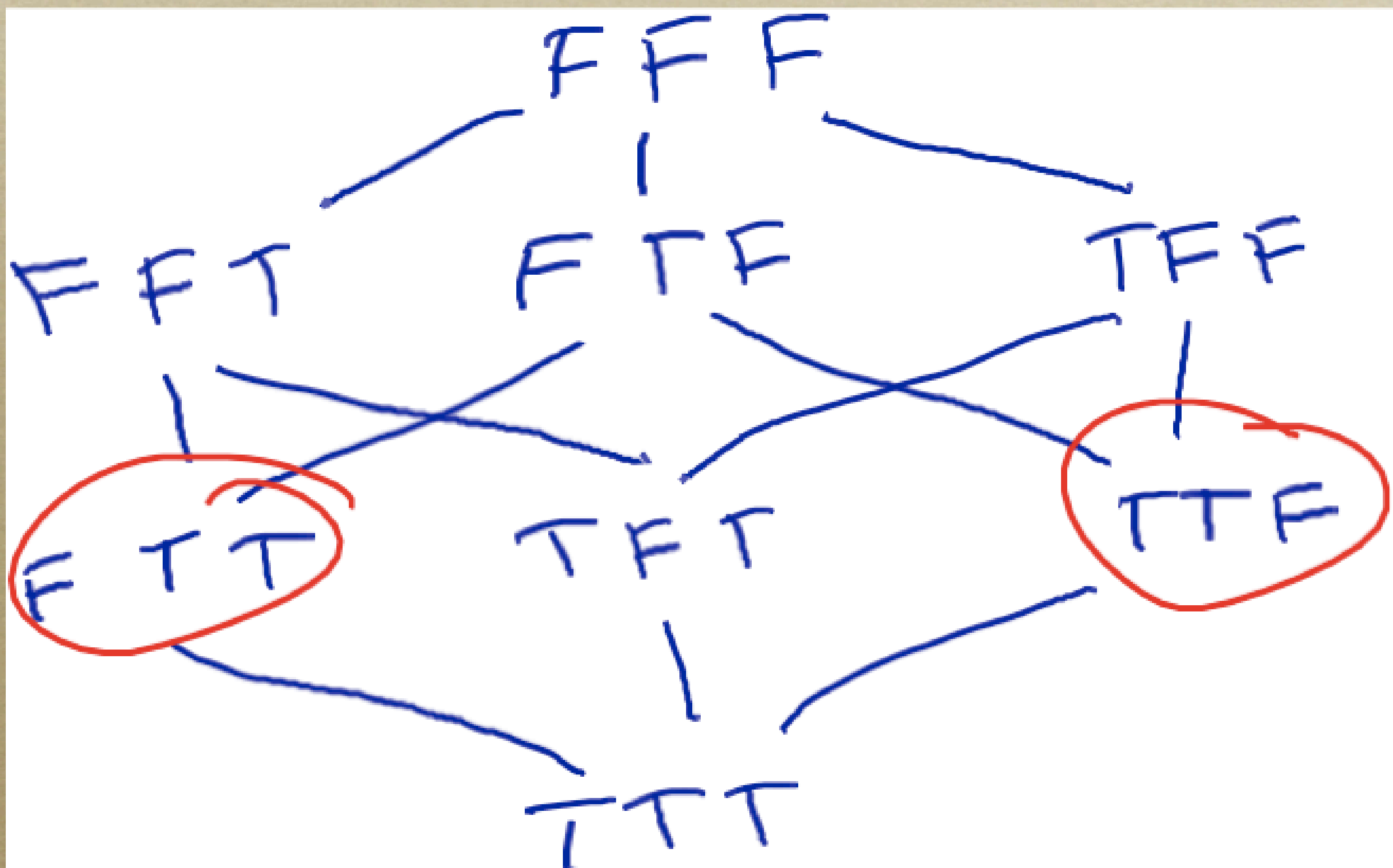
Stack: DFS



DFS discussion

- *Advantages*
  - *low memory if search graph is shallow*
- *Disadvantages*
  - *fails to terminate if graph has infinite depth*
  - *May not find shallowest solution*

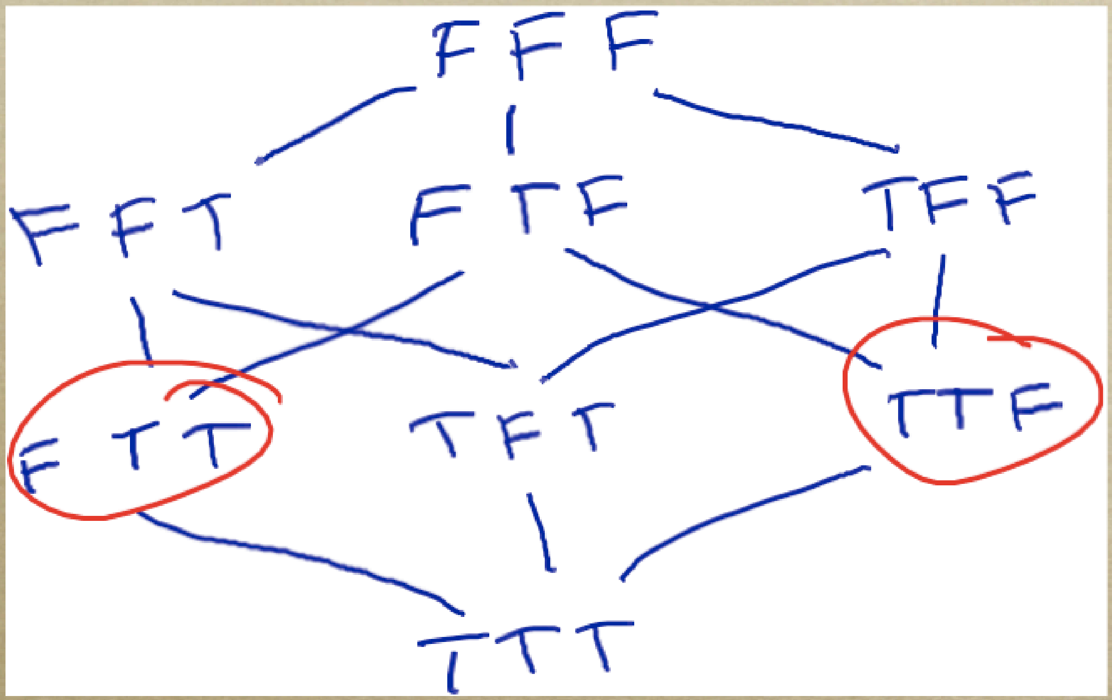
Queue: BFS



BFS discussion

- *Advantages*
  - *low memory if graph is narrow (rare)*
  - *always finds shallowest solution*
- *Disadvantages*
  - *common case: memory grows exponentially with search depth*

DFID



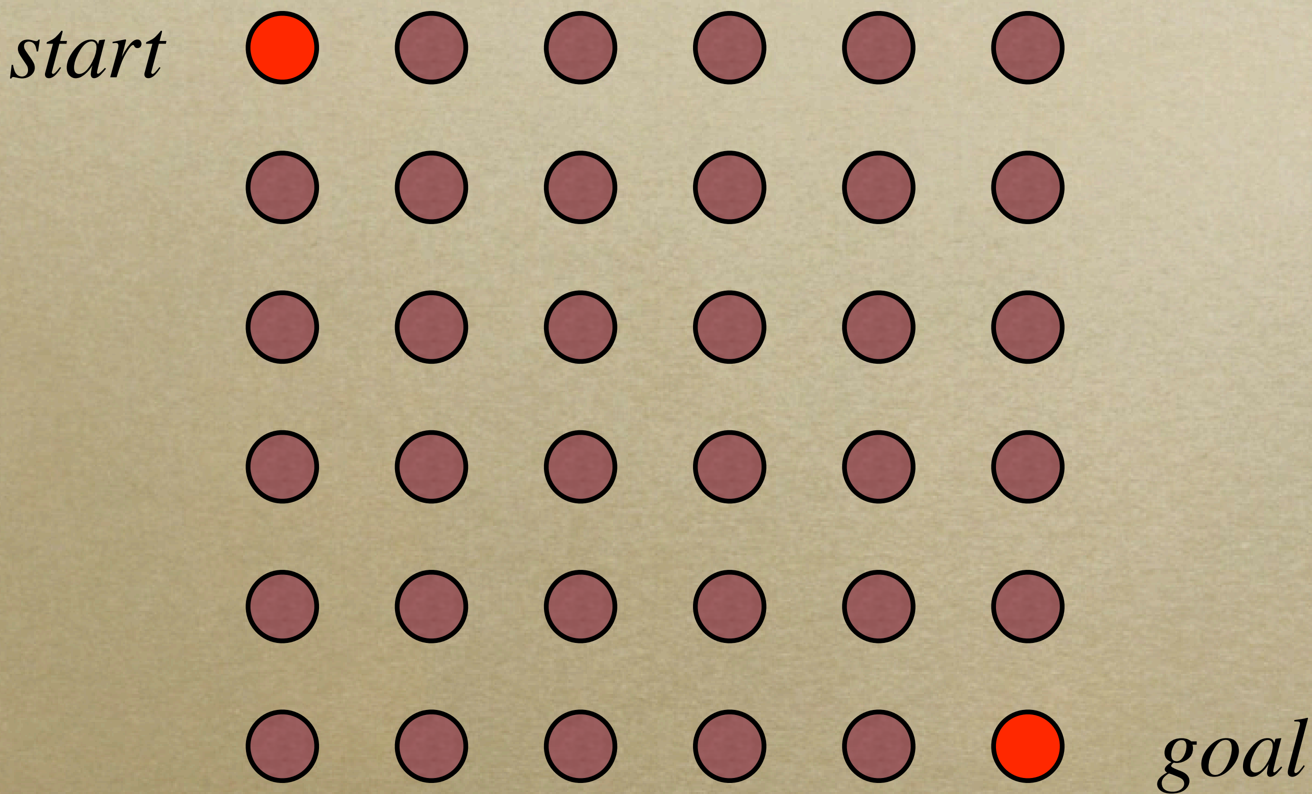
- *Run a DFS but limit depth to k*
- *If we fail to find a solution, increase k and try again*

DFID discussion

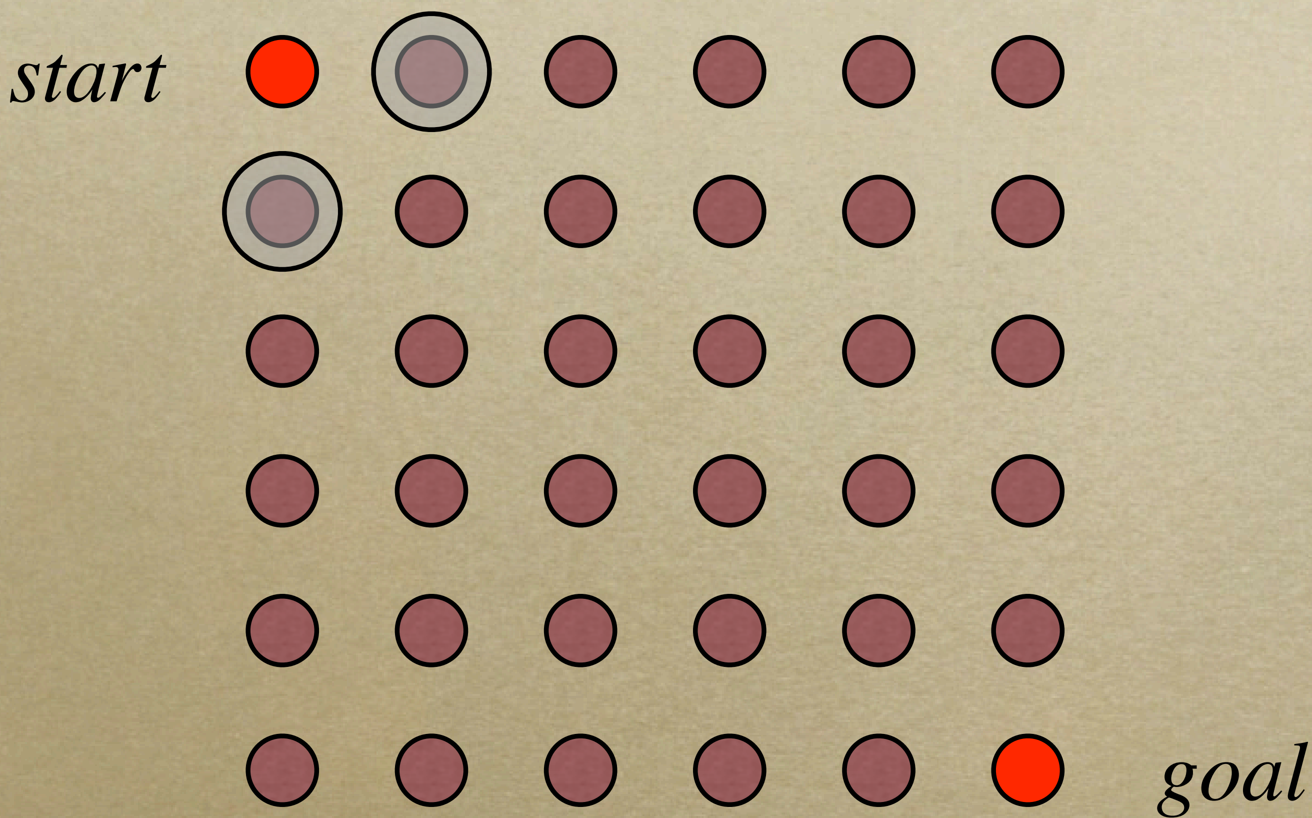
- *Combines advantages of BFS and DFS*
- *Always low memory*
- *Finds shallowest (or nearly shallowest) solution*
- *Also works for A* (described below)*

Heuristic Search

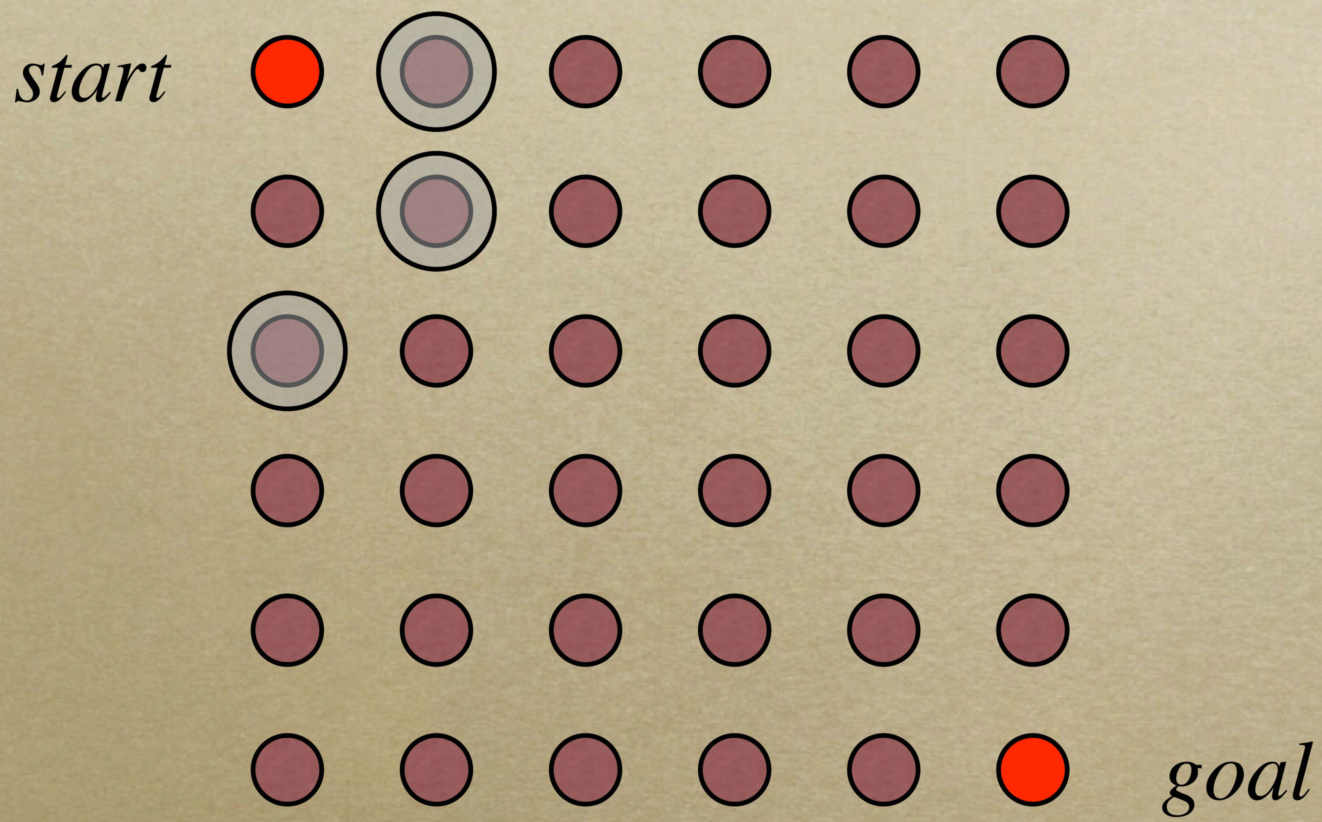
DFS looking stupid



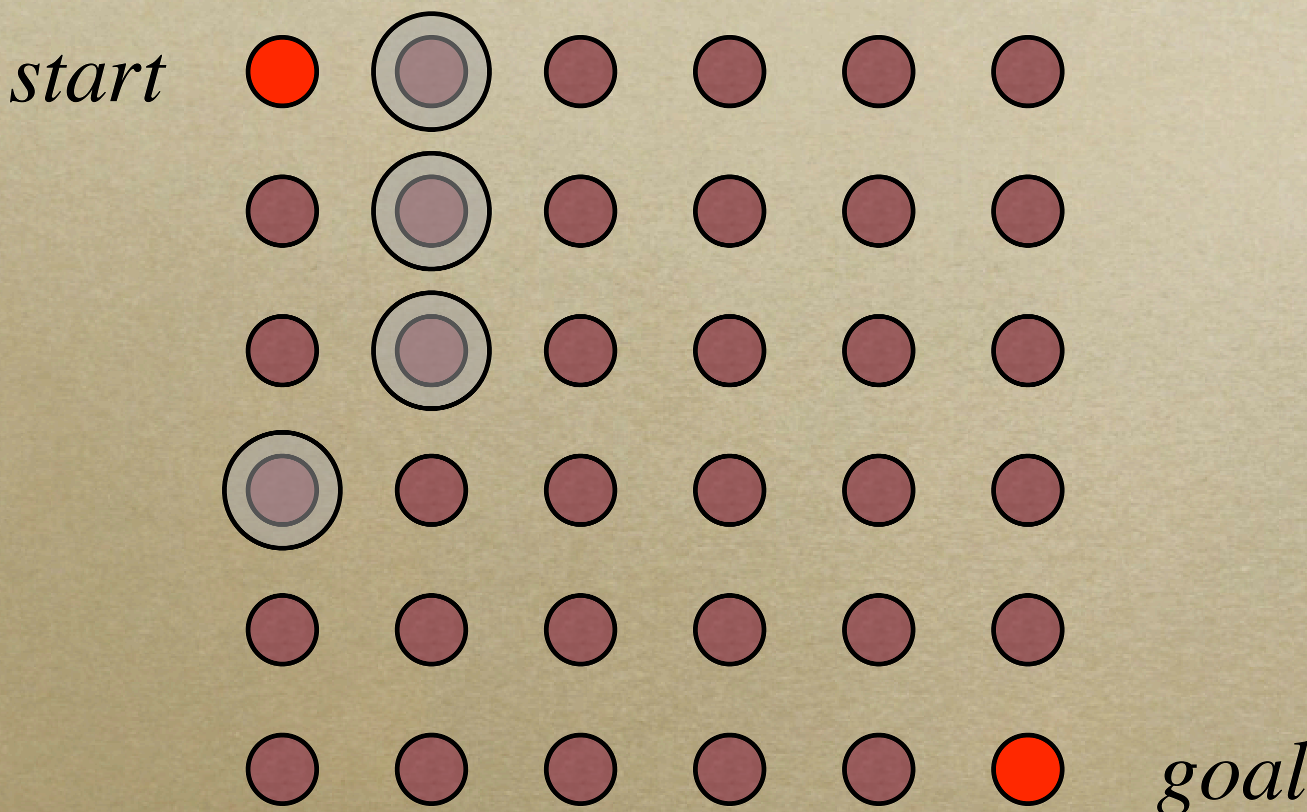
DFS looking stupid



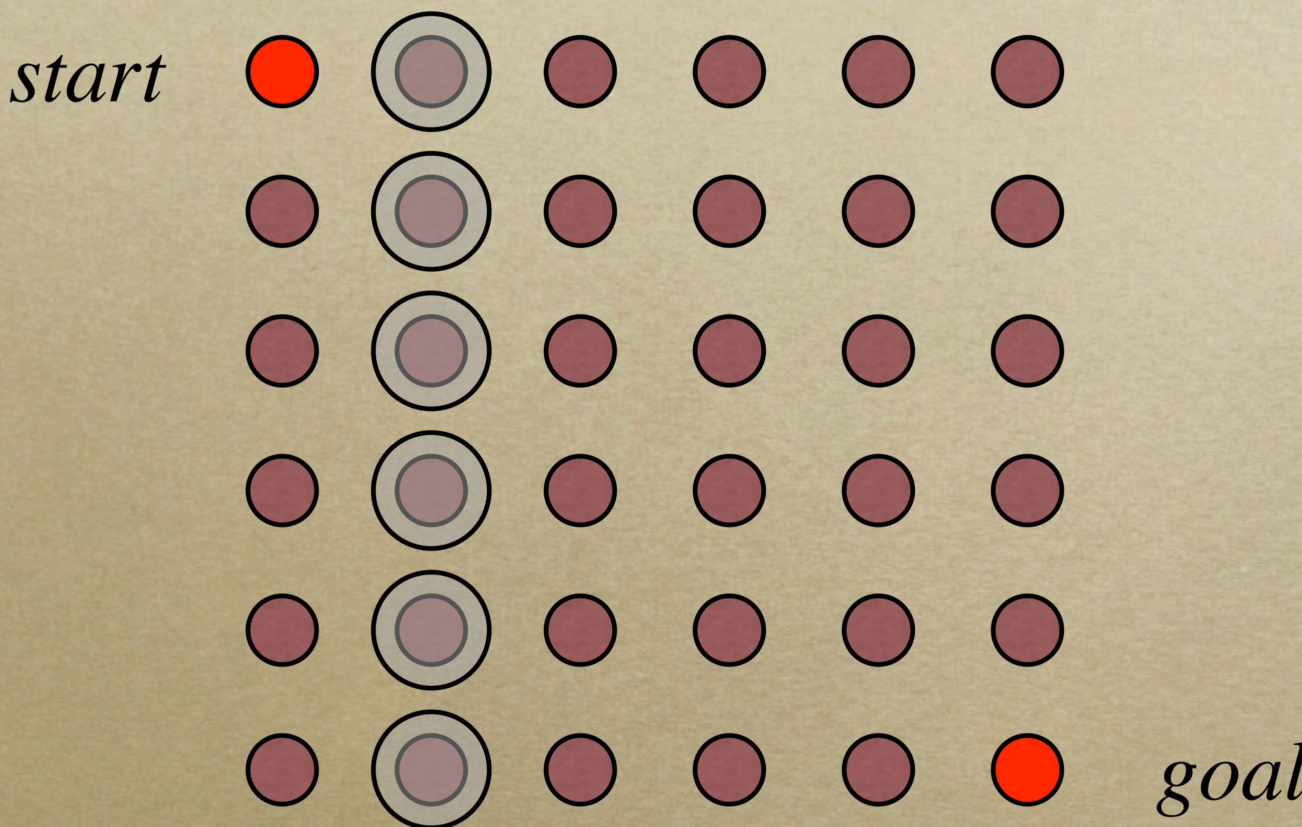
DFS looking stupid



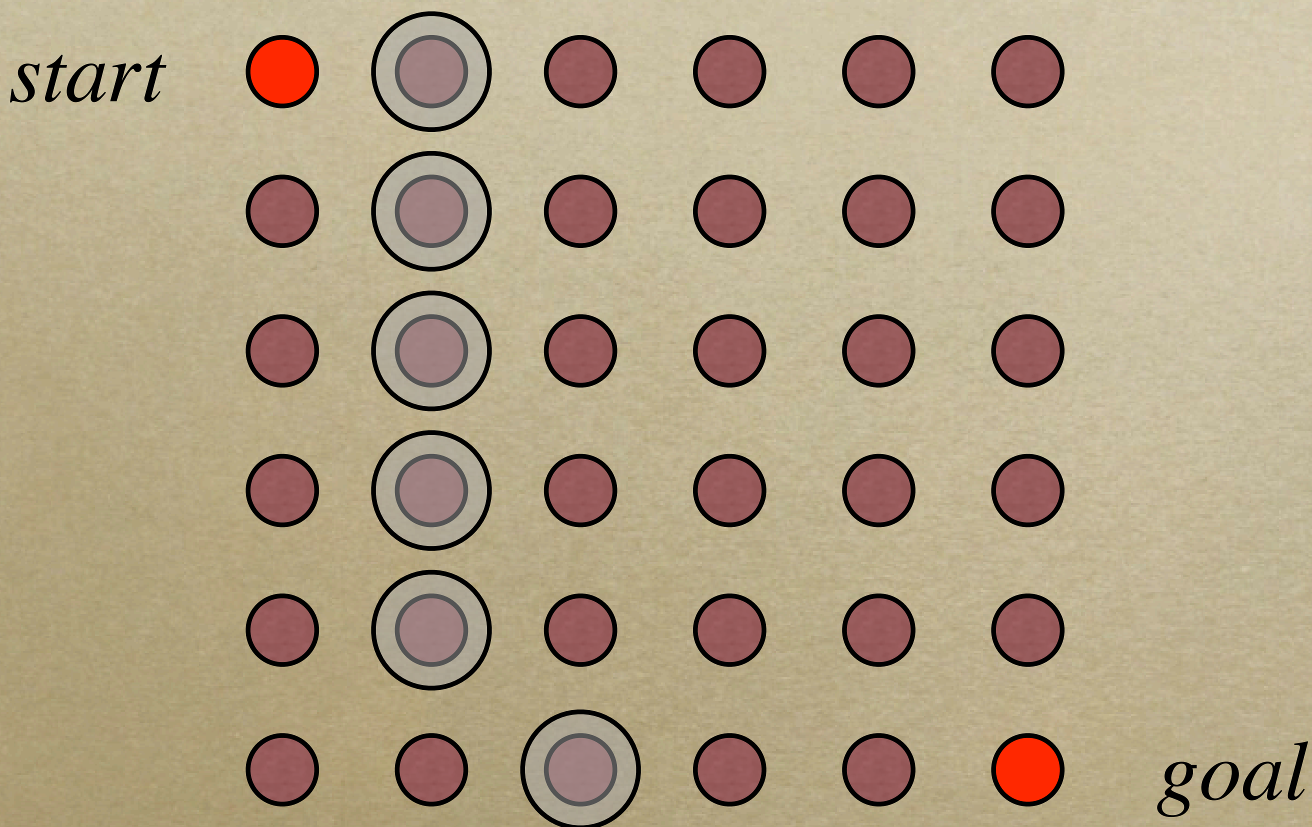
DFS looking stupid



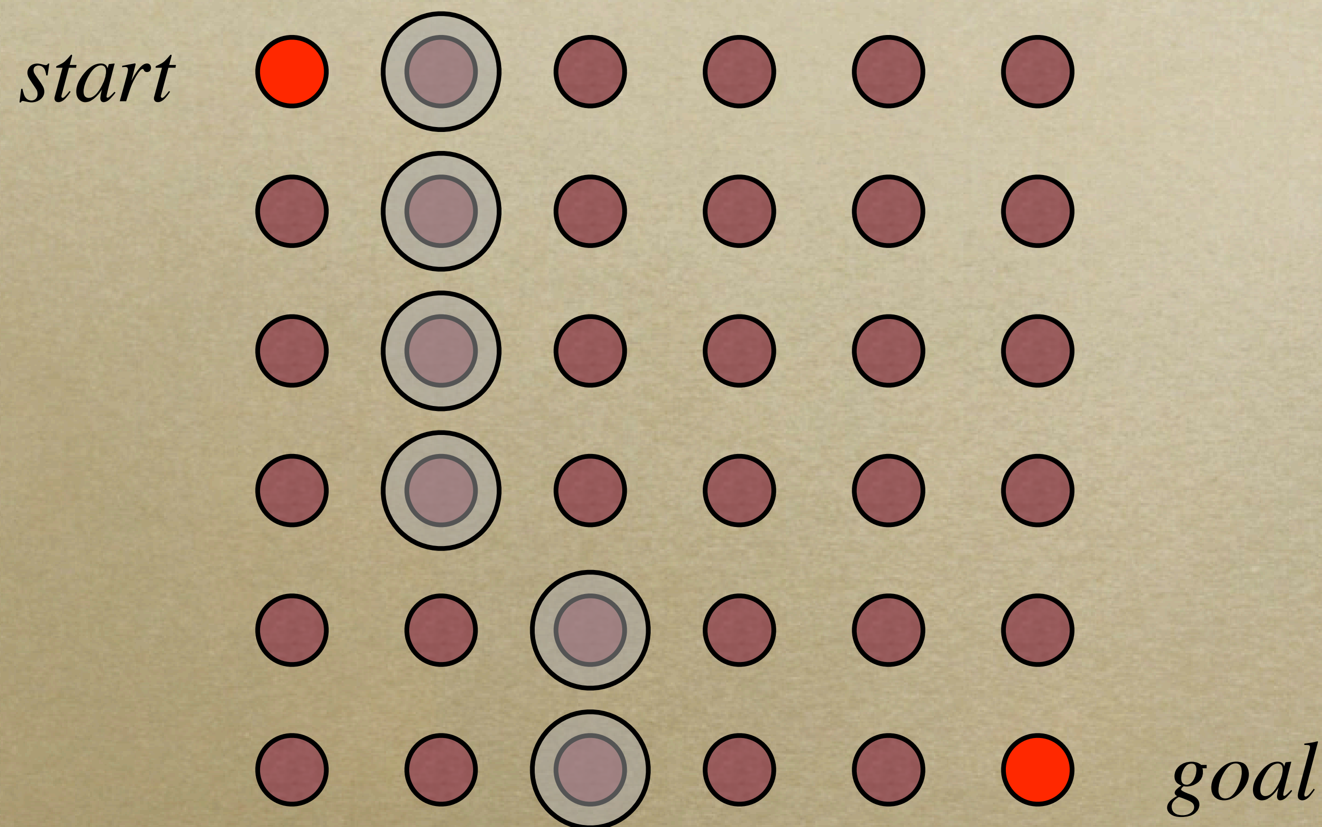
…skipping a few steps



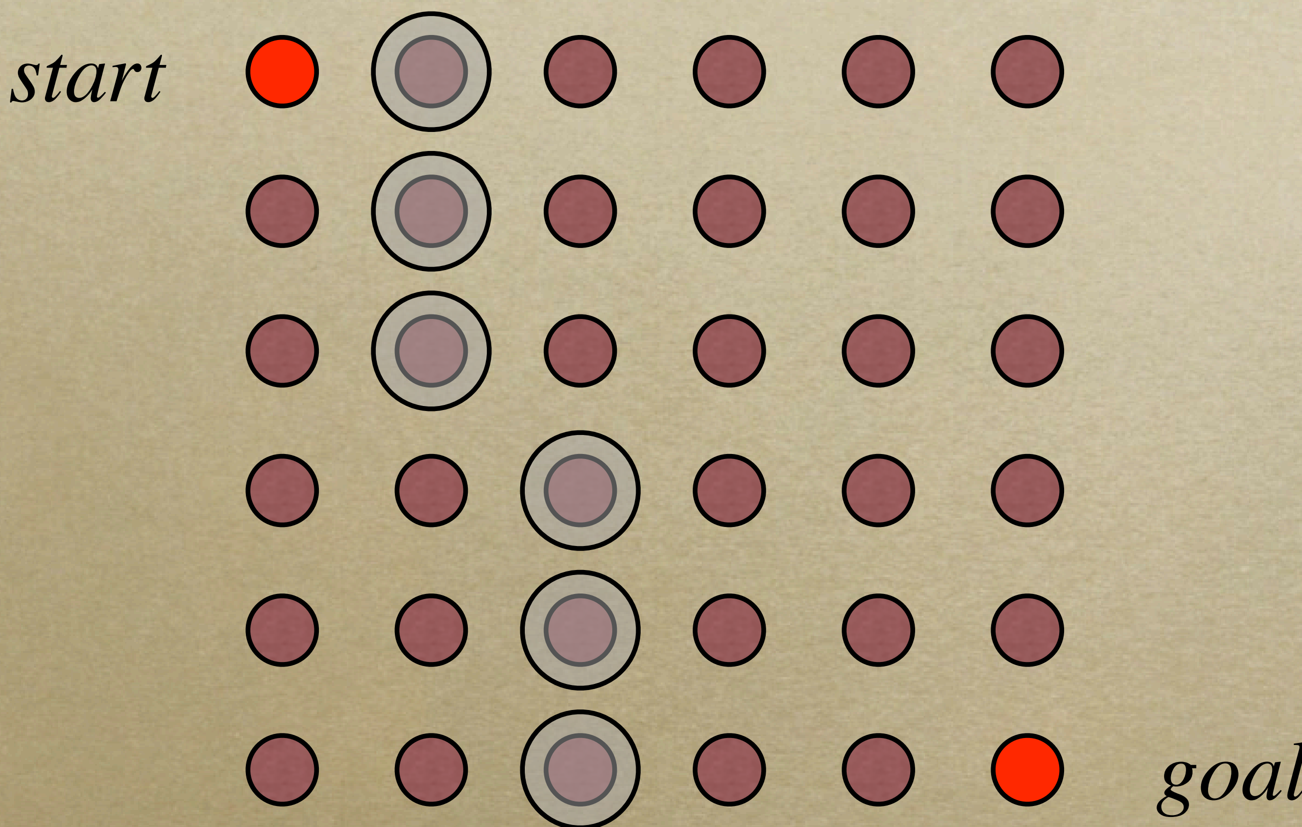
DFS looking stupid



DFS looking stupid



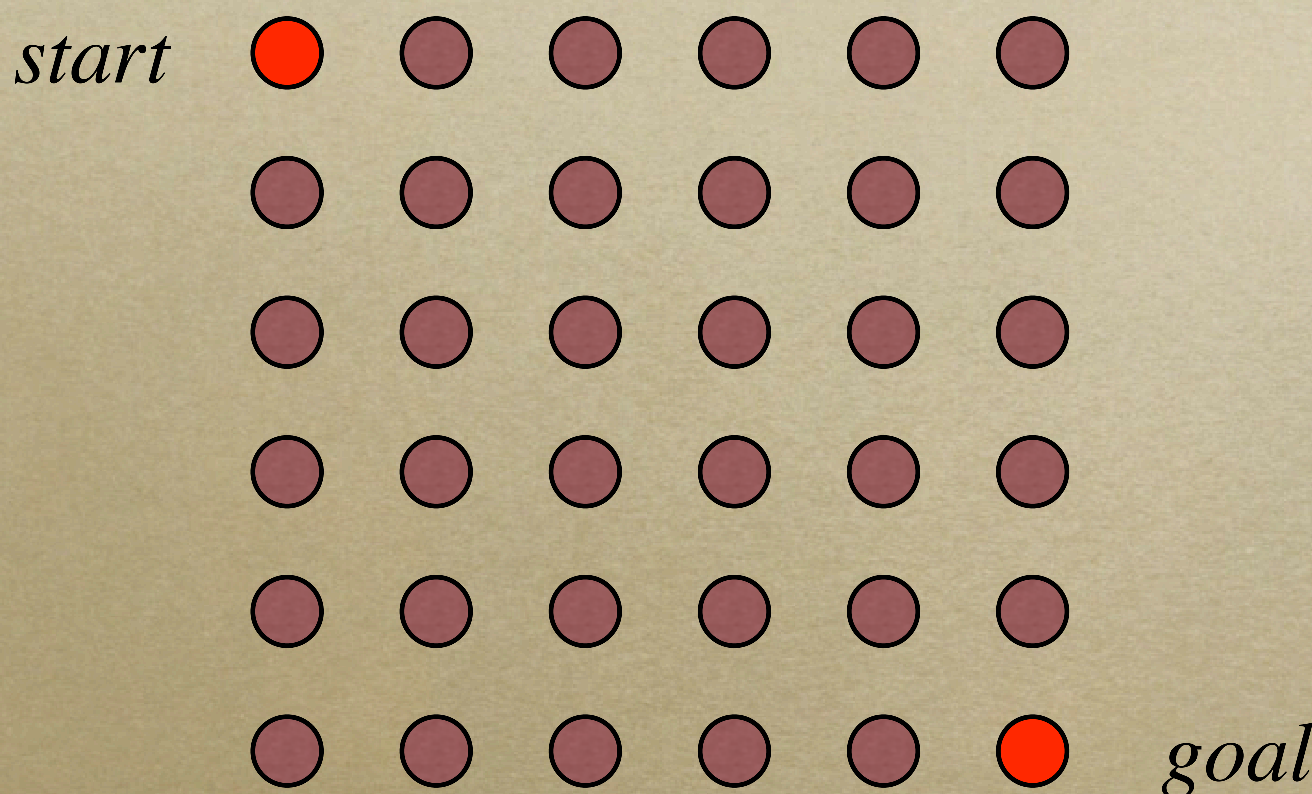
DFS looking stupid



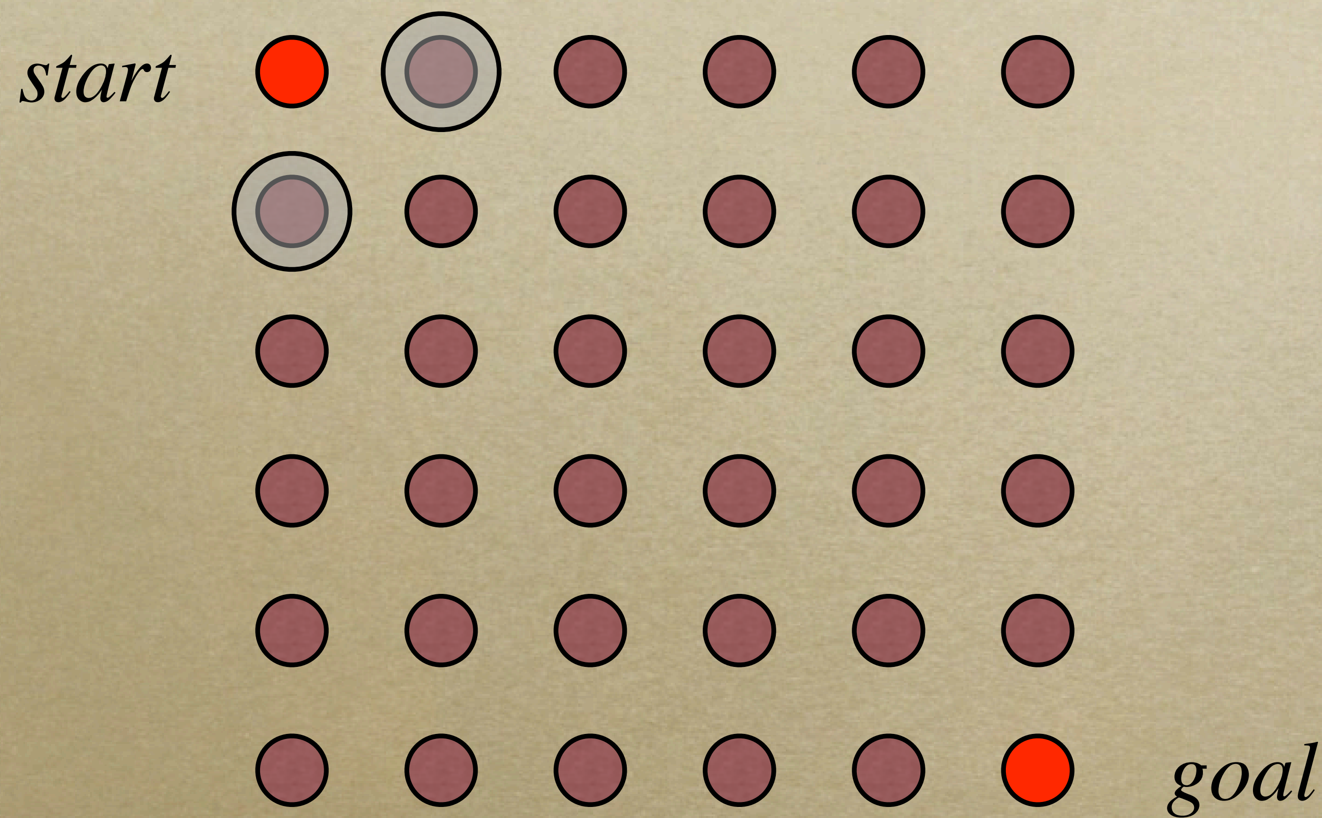
Heuristic search

- *Implement open set S with a priority queue*
  - *Ops: insert, update_priority, pop*
- *Pop always gives node w/ best priority*
- *Priority function = place to give the search algorithm additional info*
- *E.g., priority(x, y) =* |*gx-x*| + |*gy-y*|

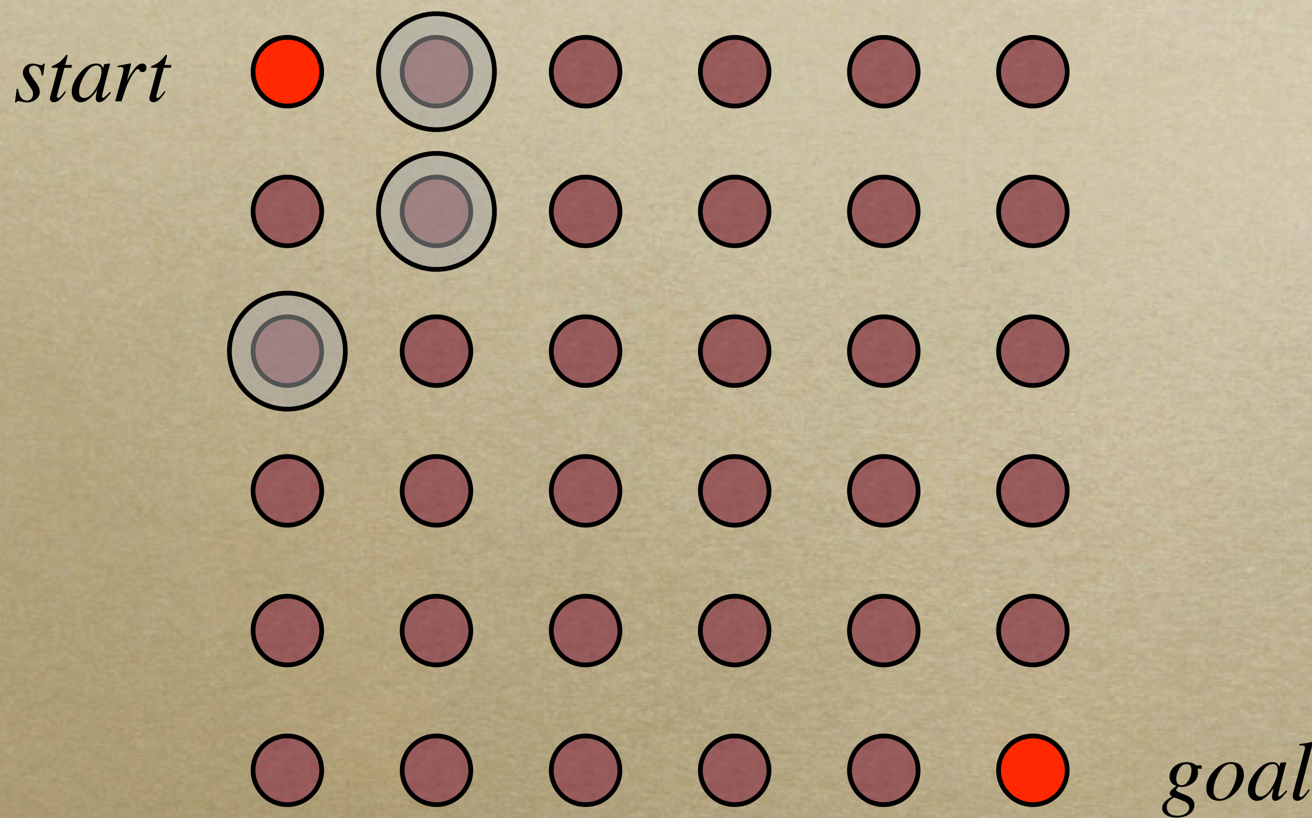
Heuristic search looking smart



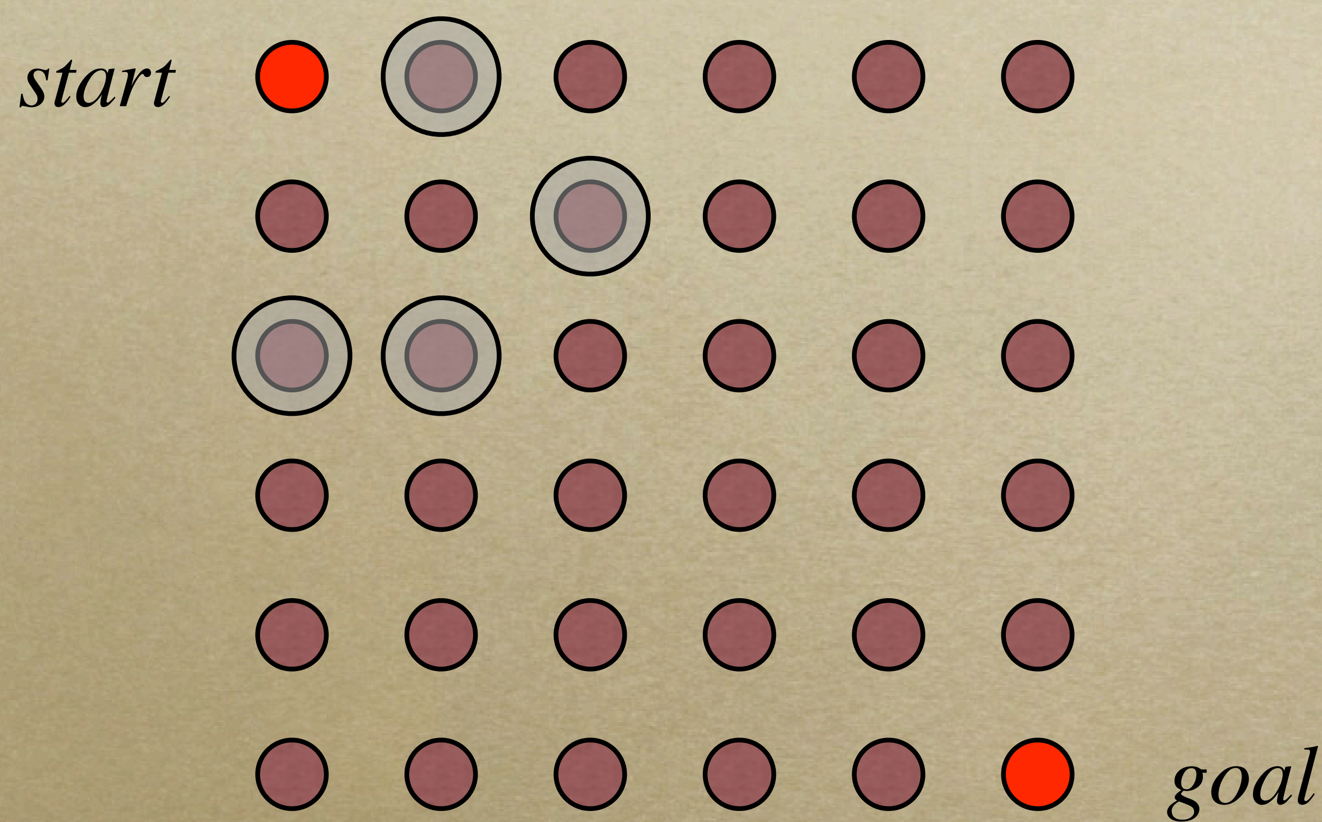
Heuristic search looking smart



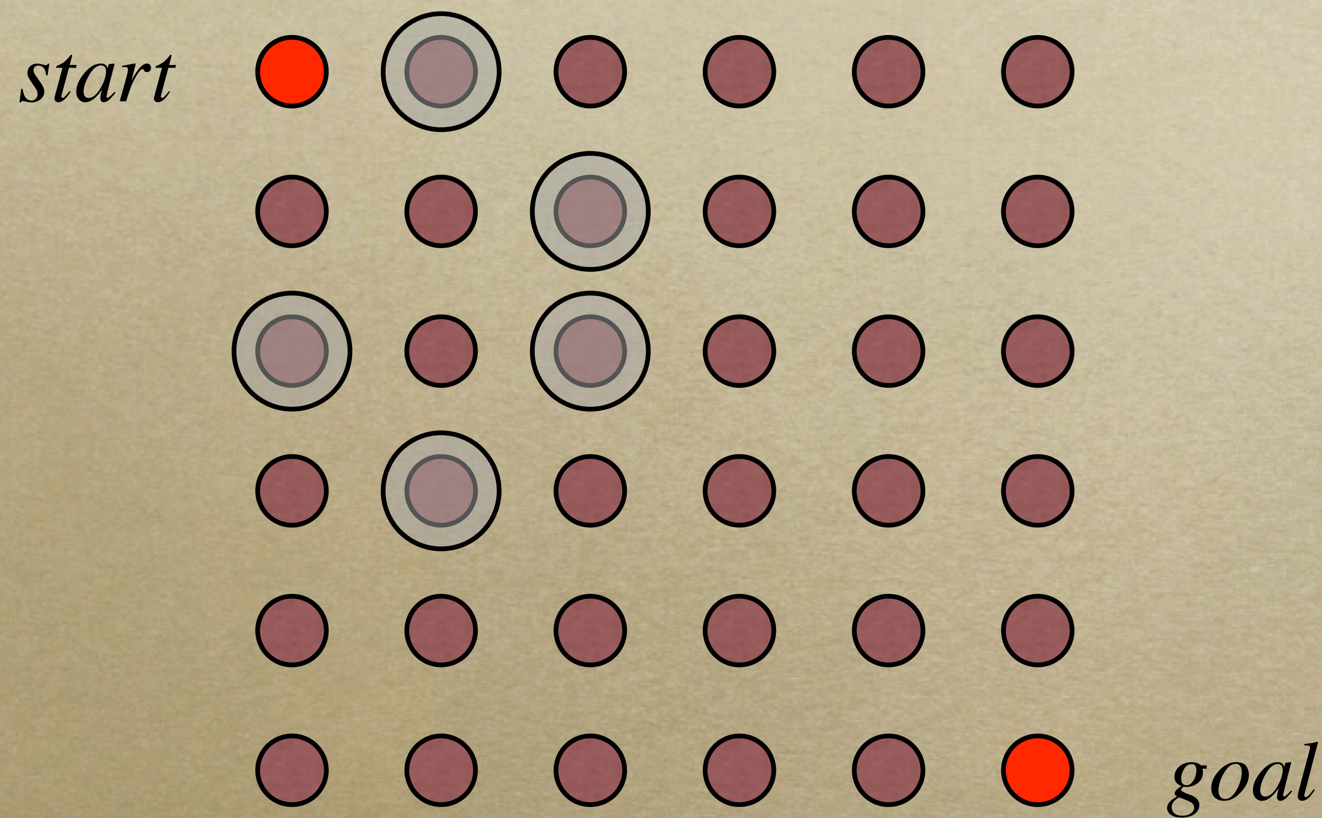
Heuristic search looking smart



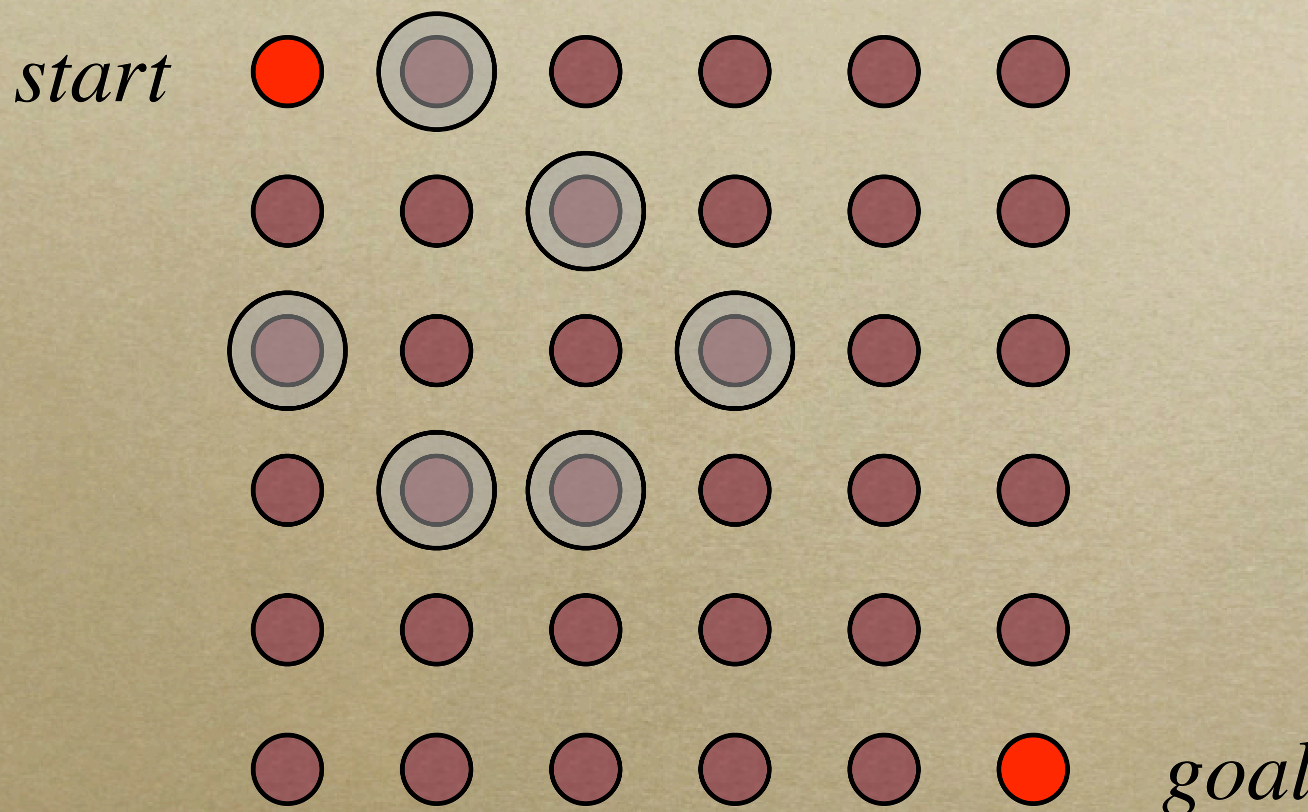
Heuristic search looking smart



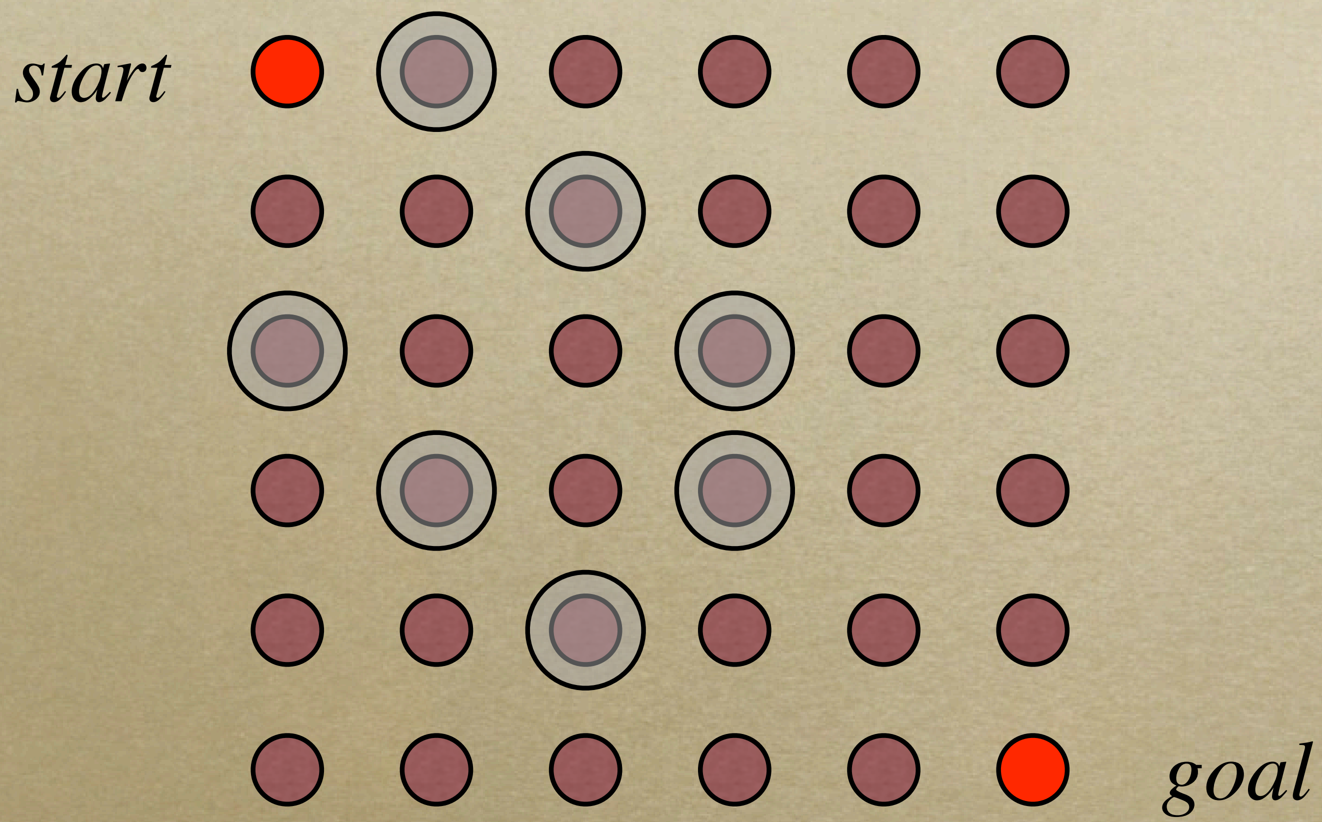
Heuristic search looking smart



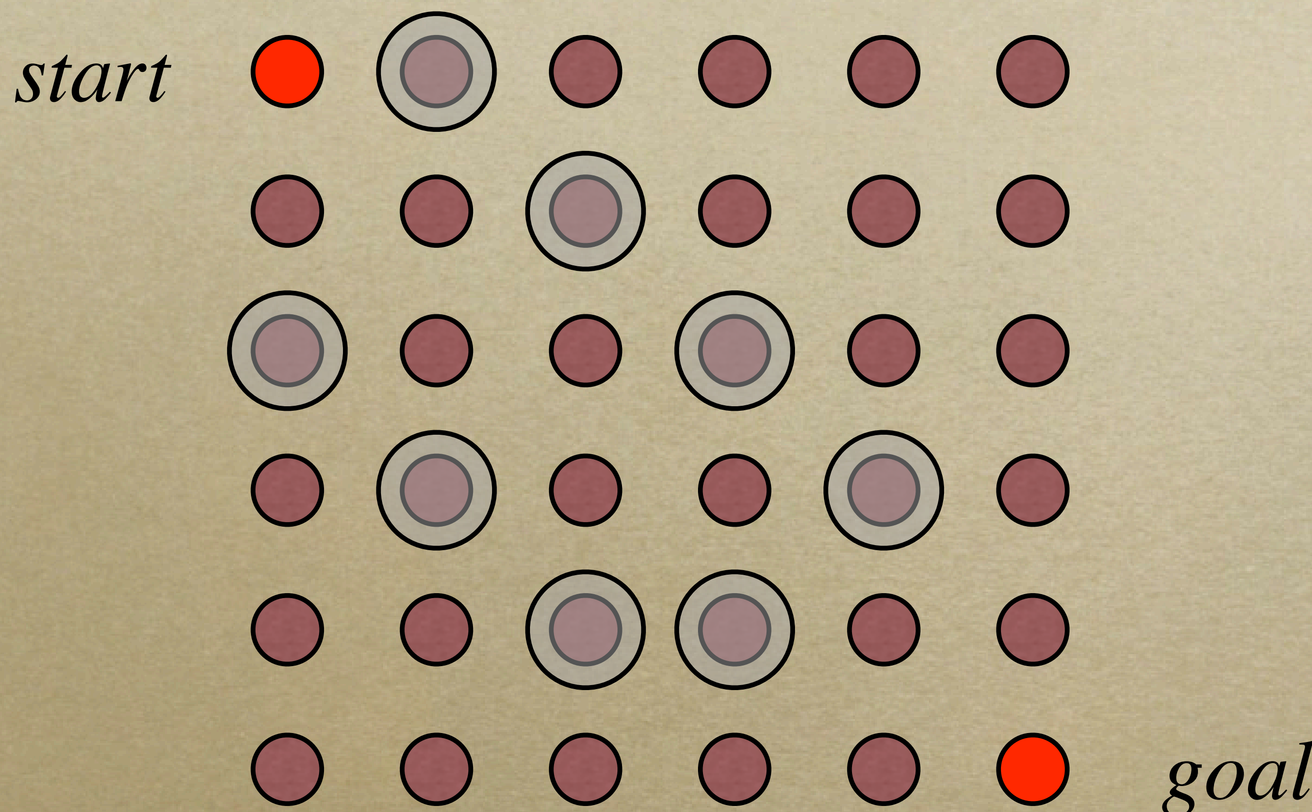
Heuristic search looking smart



Heuristic search looking smart



Heuristic search looking smart



Question

- *What is optimal heuristic?*

Heuristic search discussion

- *Cost: priqueue is more expensive than stack, queue*
- *Benefit: could visit many fewer nodes*

Question

- *When could heuristic search look dumb?*
- *Can we find conditions we can satisfy that guarantee that it won't look dumb?*