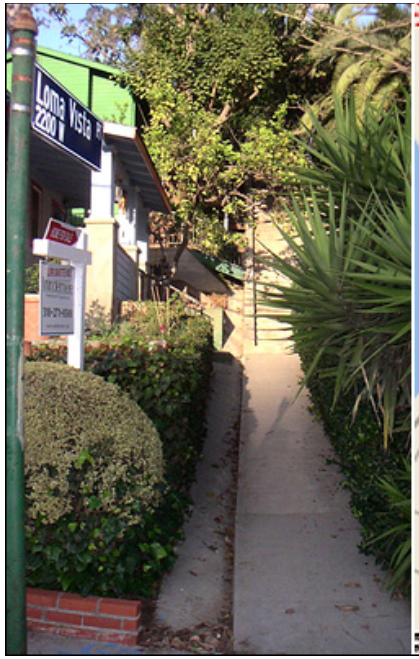
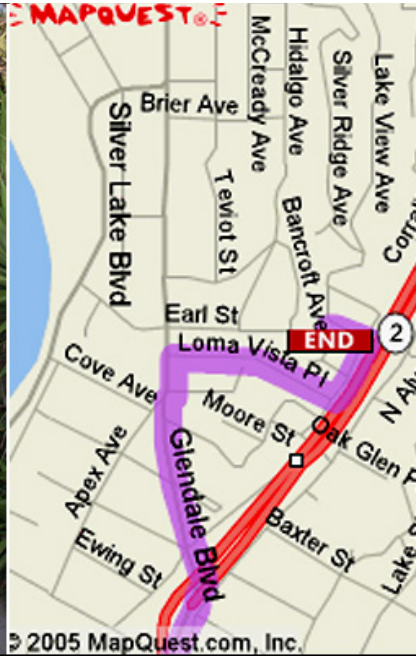
Search Gone Wrong?





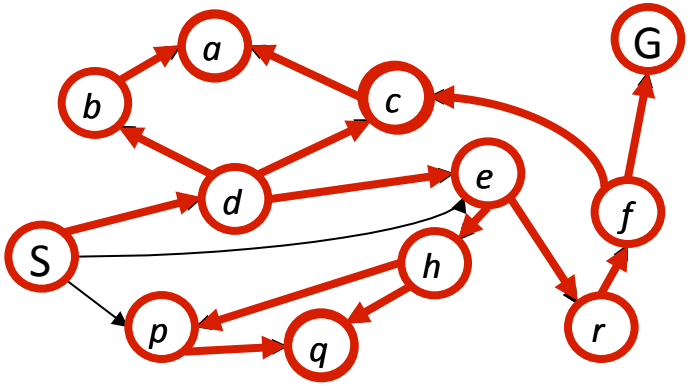
TSP

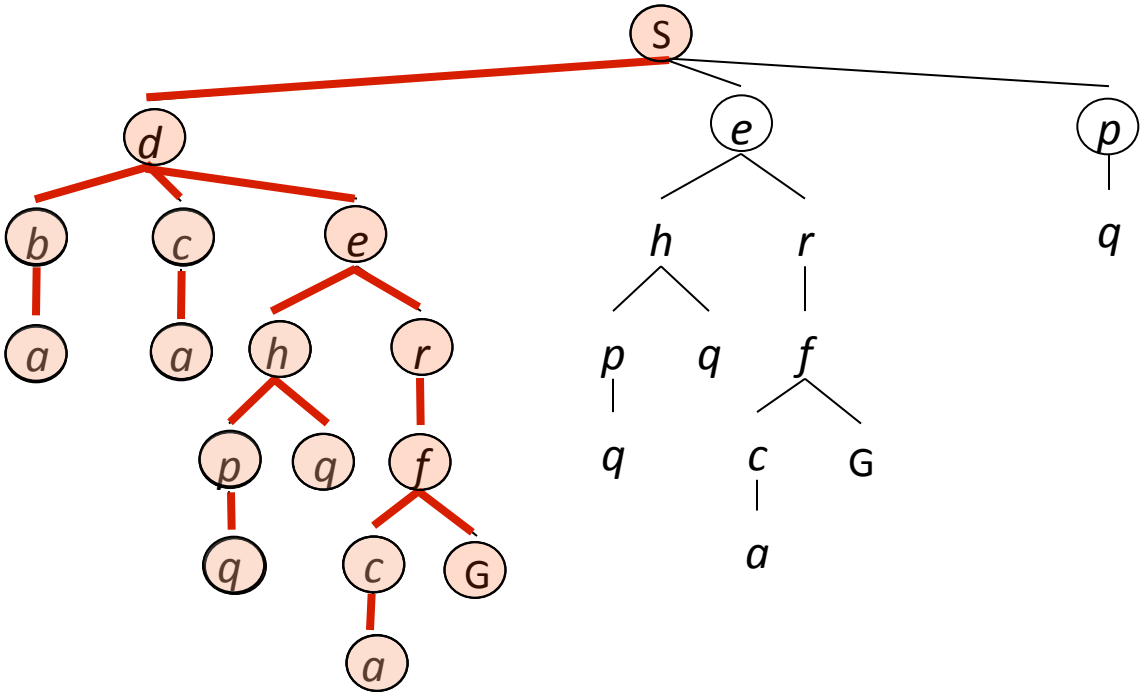
http://idle.slashdot.org/story/12/04/25/1519208/travelling-salesman-thriller-set-in-a-world-where-pnp

Depth First Search

Strategy: expand deepest node first

Implementation: Probably want to use recursion

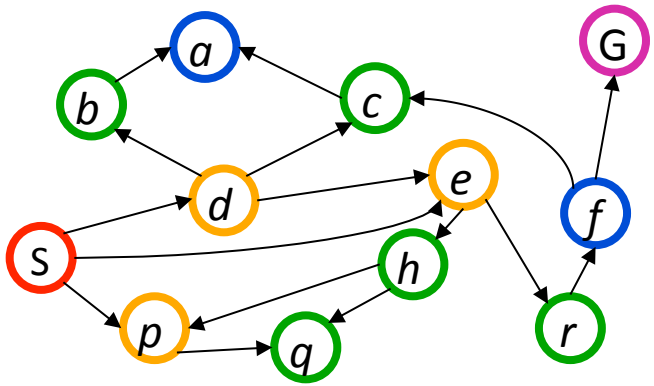


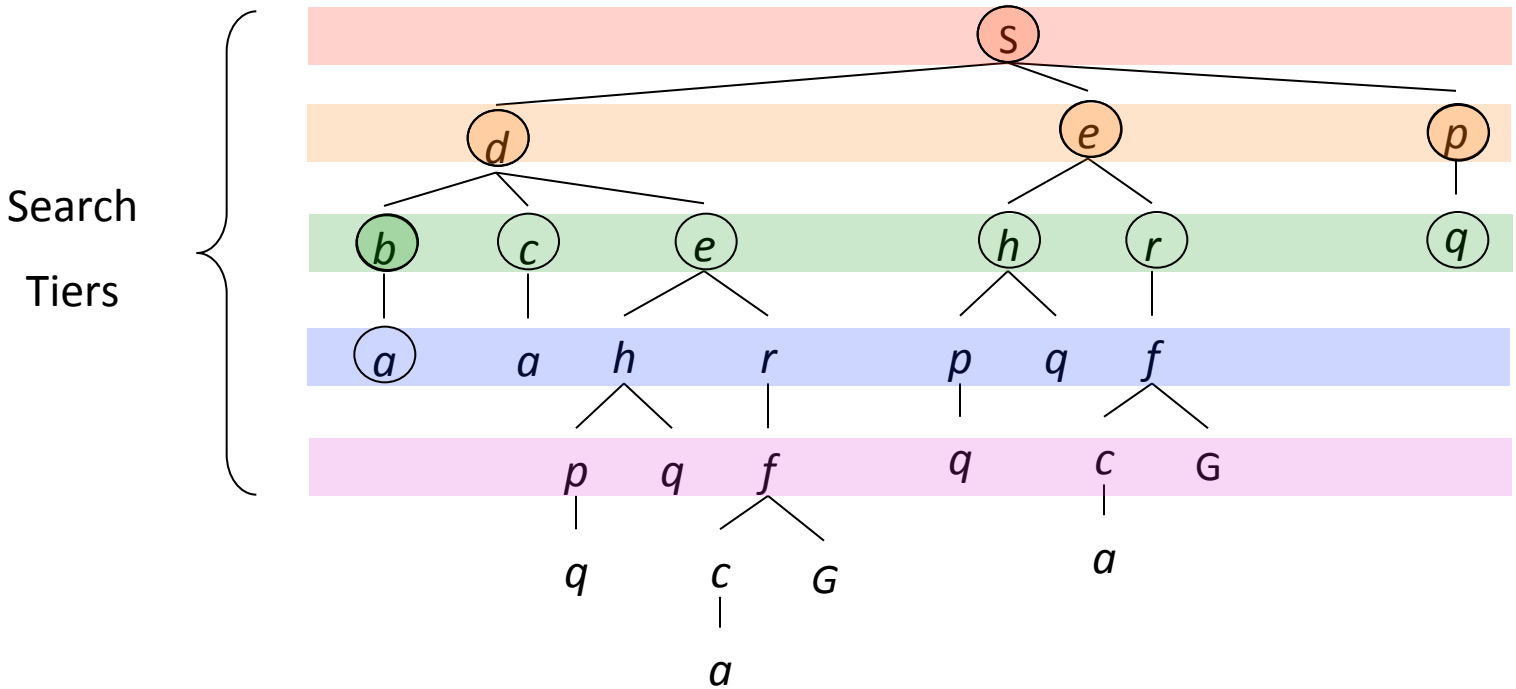


Breadth First Search

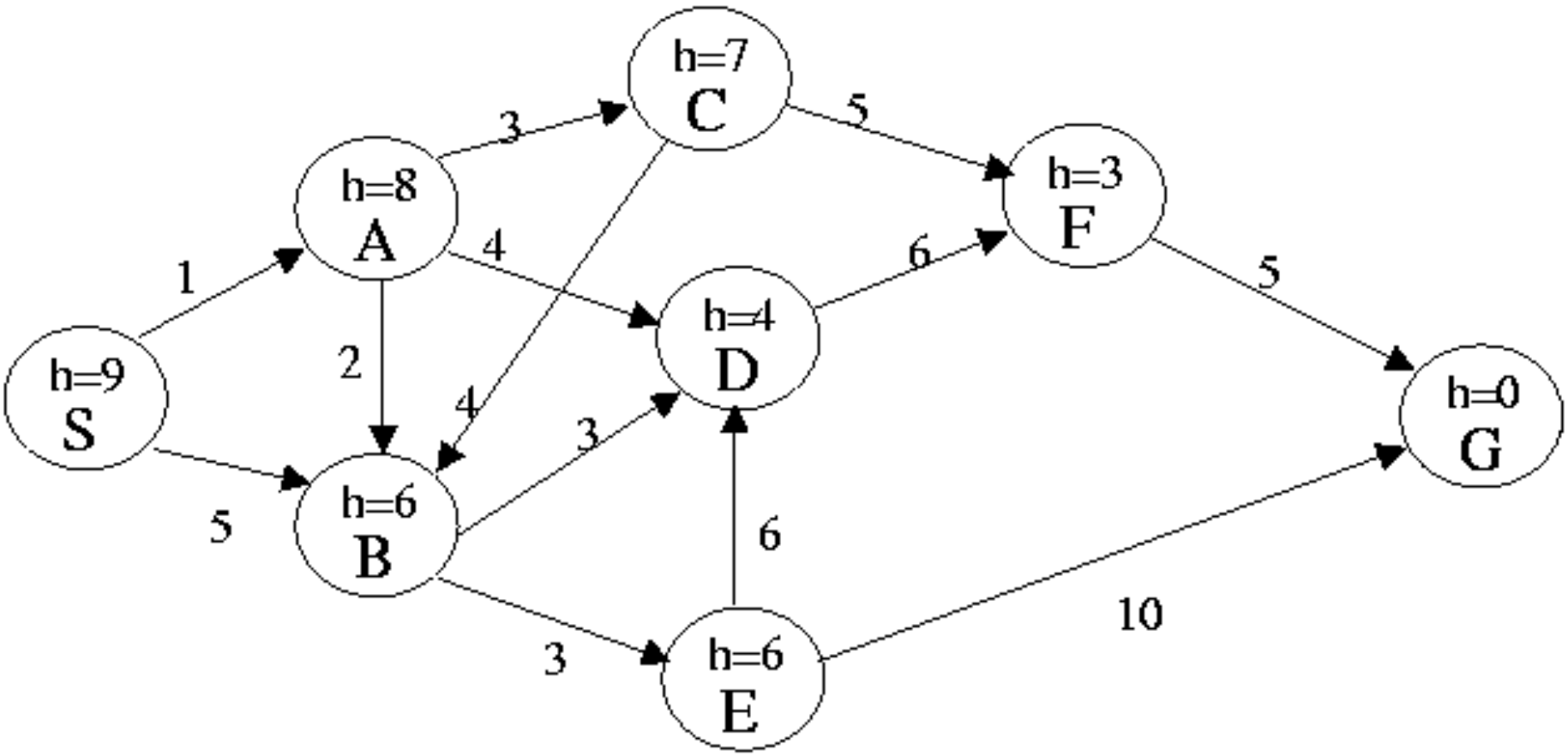
Strategy: expand shallowest node first

Implementation: Fringe is a FIFO queue





DFS vs. BFS



In this problem the start state is **S**, and the goal state is **G.** The transition costs are next to the edges. Ignore "h". Assume that ordering is defined by, and ties are always broken by, choosing the state that comes first alphabetically.

Search Algorithm Properties

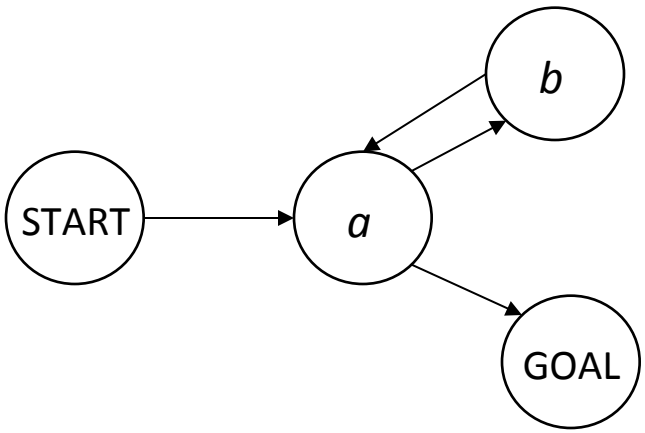
- Complete? Guaranteed to find a solution if one exists?
- Optimal? Guaranteed to find the least cost path?
- Time complexity?
- Space complexity?

Variables:

n	Number of states in the problem
b	The average branching factor B (the average number of successors)
C^*	Cost of least cost solution
s	Depth of the shallowest solution
m	Max depth of the search tree

DFS

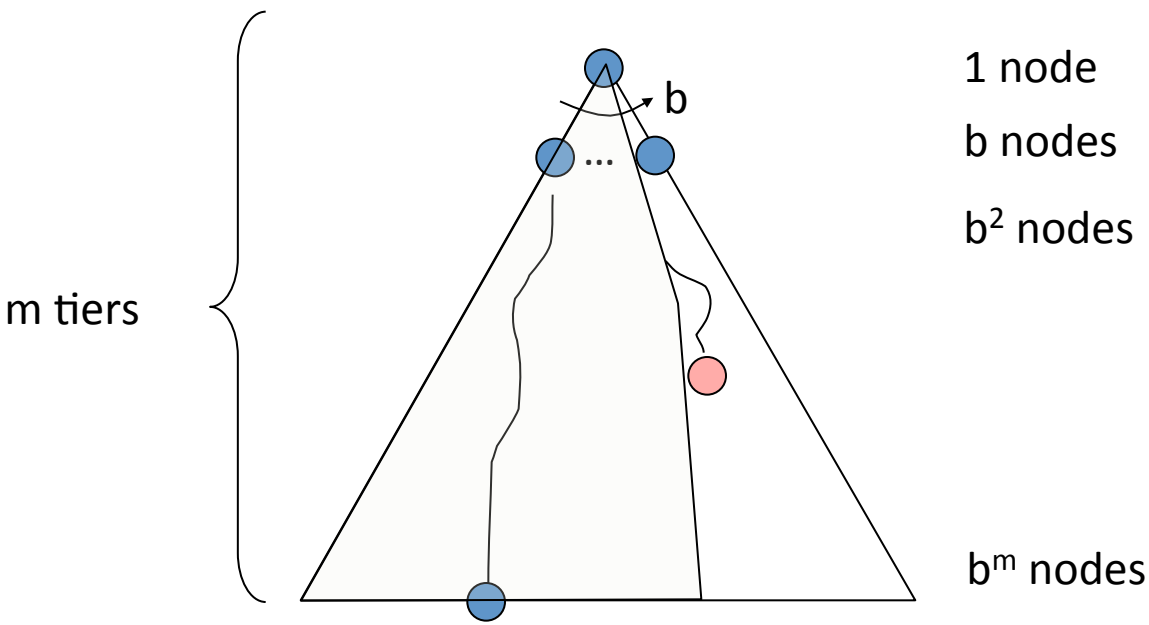
Algorithm		Complete	Optimal	Time	Space
DFS	Depth First Search	N	N	Infinite	Infinite



- Infinite paths make DFS incomplete…
- How can we fix this?

DFS

- With cycle checking, DFS is complete.*

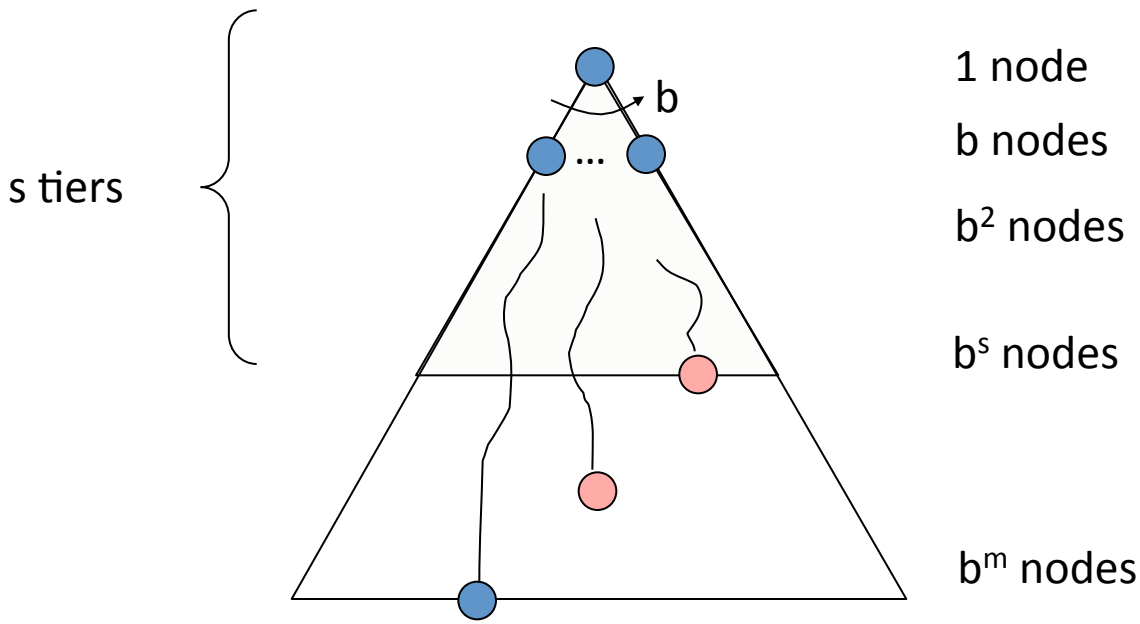


Algorithm		Complete	Optimal	Time	Space
DFS	w/ Path Checking	Y	N	$O(b^{m+1})$	$O(bm)$

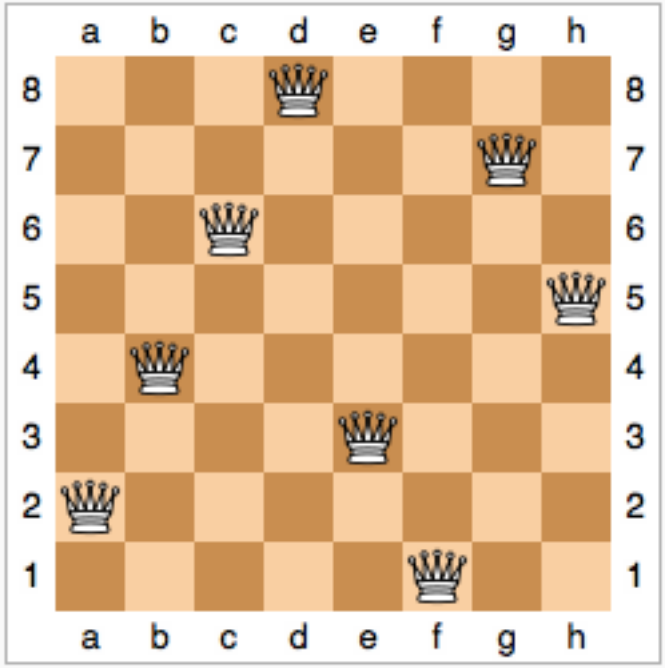
* Or graph search

BFS

Algorithm		Complete	Optimal	Time	Space
DFS	w/ Path Checking	Y	N	$O(b^{m+1})$	$O(bm)$
BFS		Y	N*	$O(b^{s+1})$	$O(b^{s})$



- When is BFS optimal?



One solution to the eight queens puzzle

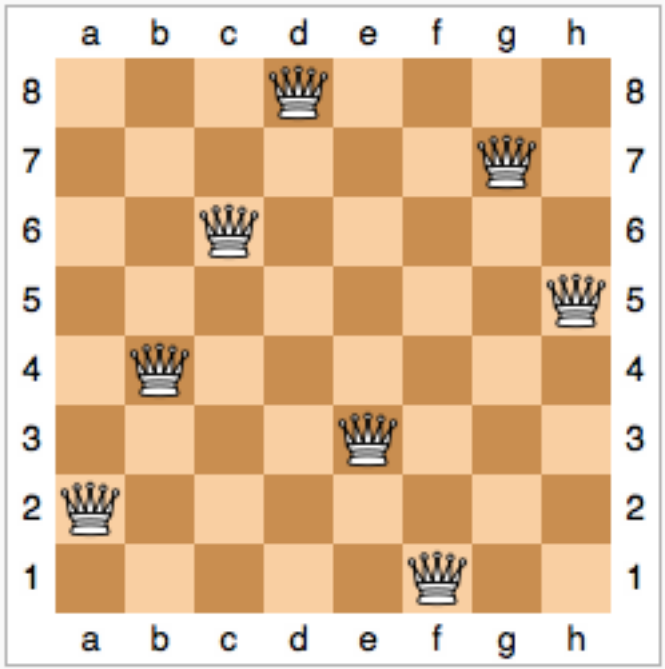
Search tree:

- Sequentially place queens
- Goal condition: no queen can take another

Comparisons

- When will BFS outperform DFS?

- When will DFS outperform BFS?



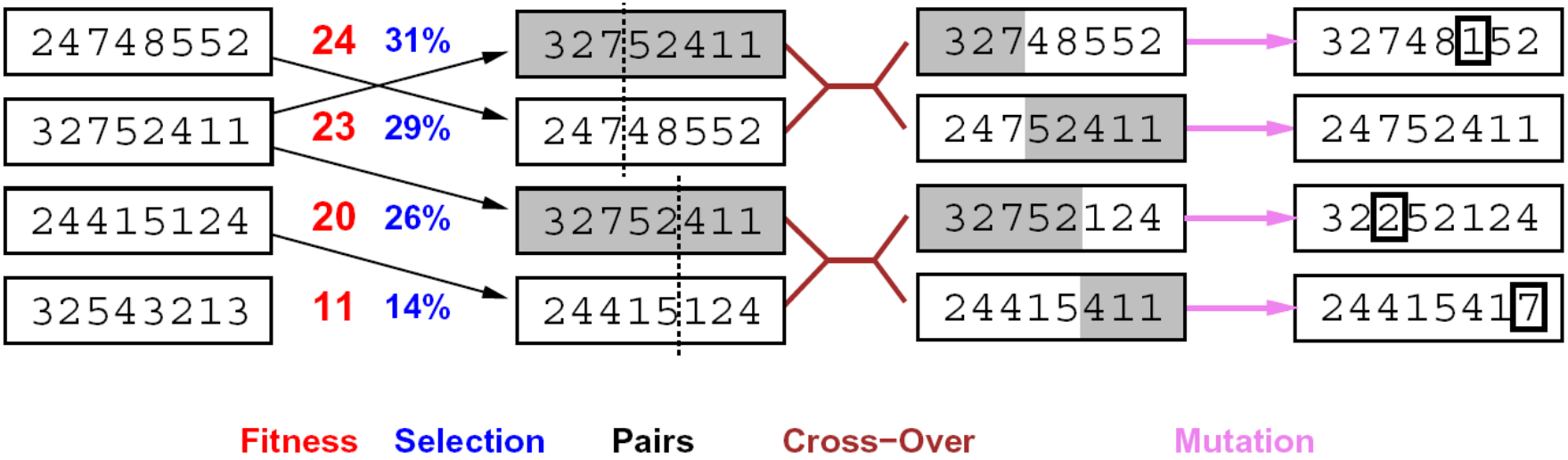
One solution to the eight queens puzzle

24683175

Genetic Algorithms

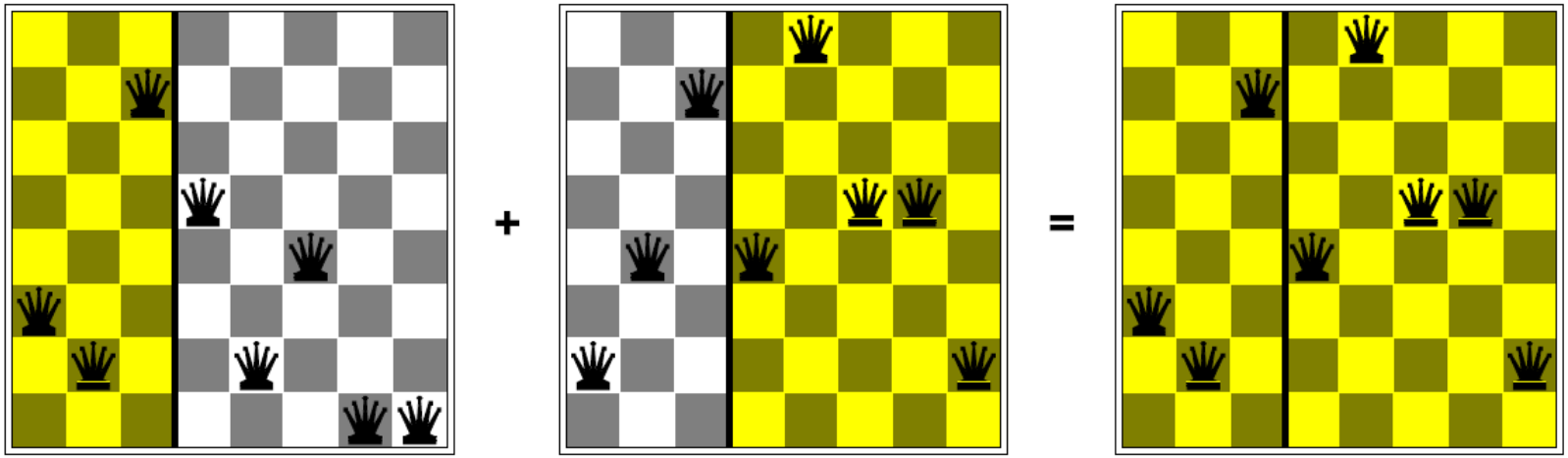
1. Generate population
2. Evaluate solutions
   - Fitness function defines “goodness”
3. Evolve new population
   - Direct selection
   - Crossover
   - Mutation
4. Goto line 2

Genetic Algorithms



- Genetic algorithms use a natural selection metaphor
- Probably the most misunderstood, misapplied (and even maligned) technique around!

Example: N-Queens



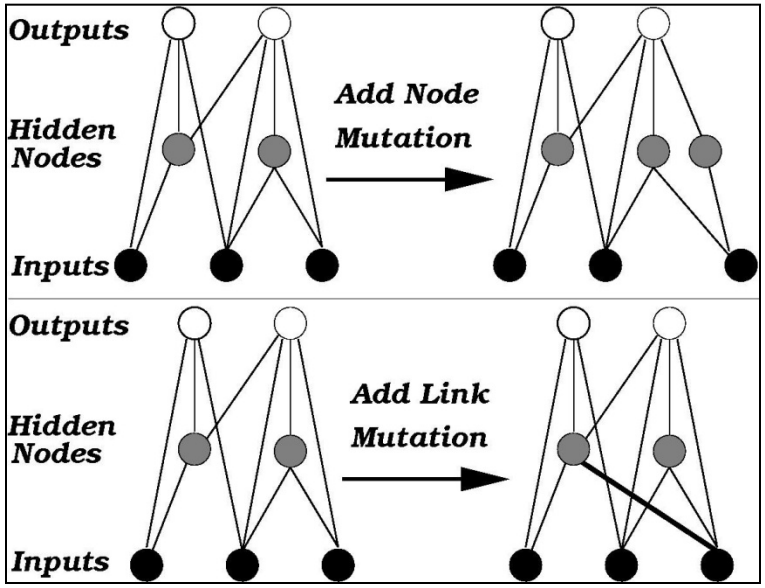
- Why does crossover make sense here?
- When wouldn't it make sense?
- What would mutation be?
- What would a good fitness function be?

- http://electricsheep.org/

NEAT

[Stanley & Miikkulainen, '02]

- NeuroEvolution of Augmenting Topologies:
  - Evolves neural networks with a genetic algorithm
- Evaluate each policy in a population
- Learns structure and weights via:
  - Mutation
  - Genetic crossover
  - Adding nodes
  - Adding links

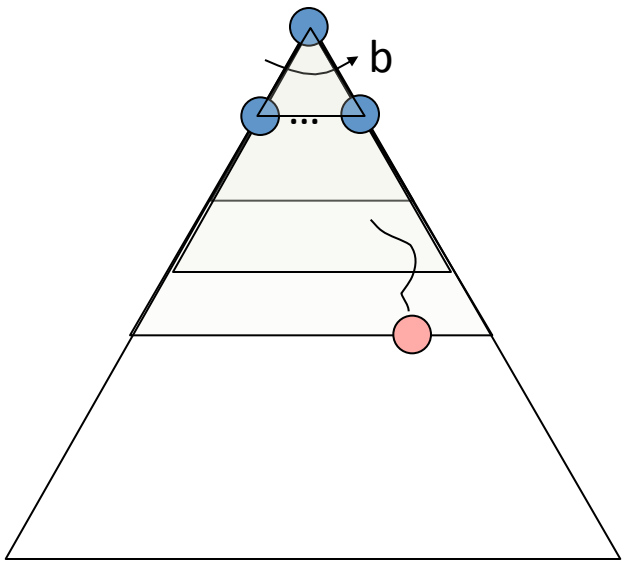


Iterative Deepening

Iterative deepening uses DFS as a subroutine:

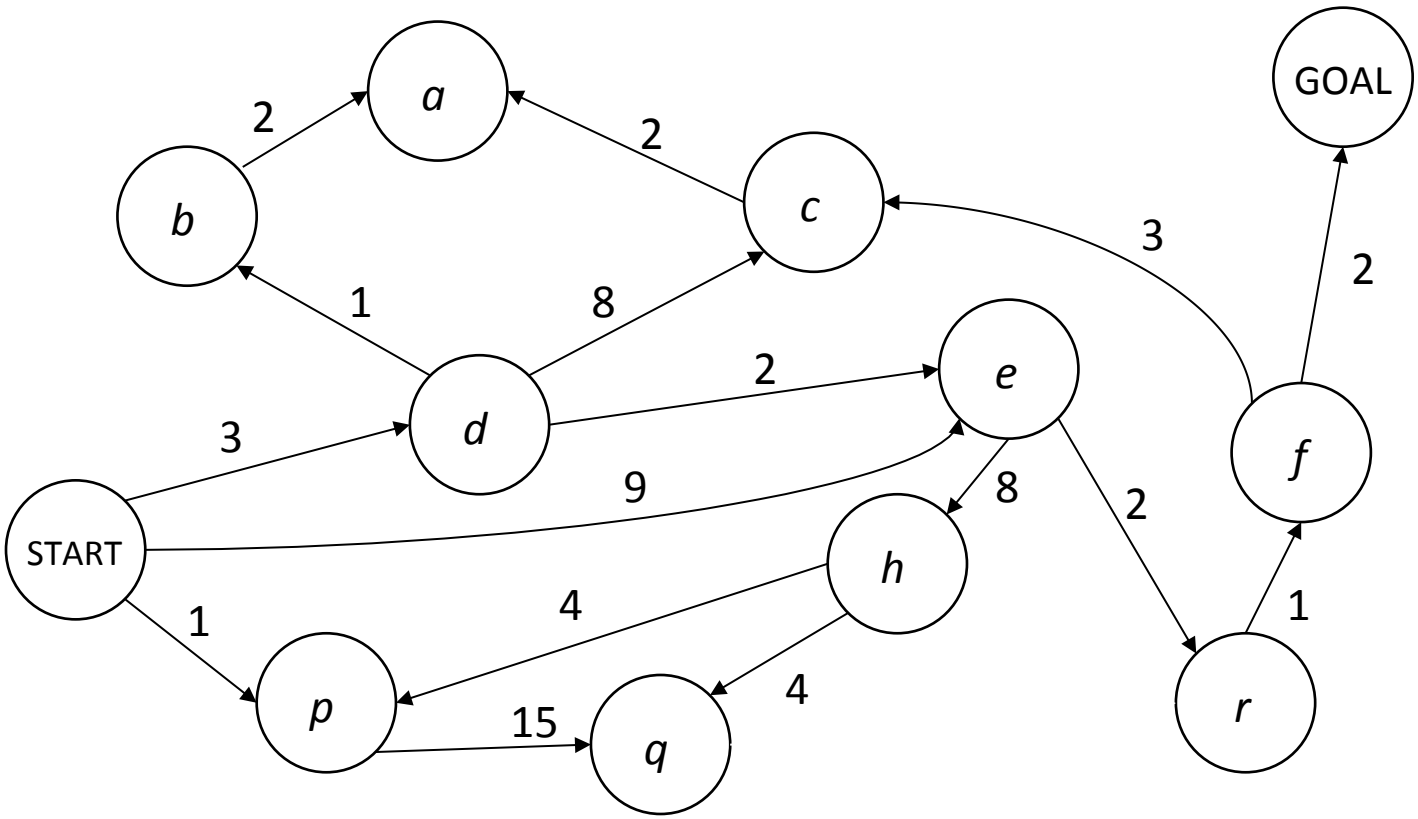
1. Do a DFS which only searches for paths of length 1 or less.
2. If "1" failed, do a DFS which only searches paths of length 2 or less.
3. If "2" failed, do a DFS which only searches paths of length 3 or less.

....and so on.



Algorithm		Complete	Optimal	Time	Space
DFS	w/ Path Checking	Y	N	$O(b^{m+1})$	$O(bm)$
BFS		Y	N*	$O(b^{s+1})$	$O(b^{s})$
ID		Y	N*	$O(b^{s+1})$	$O(bs)$

Costs on Actions



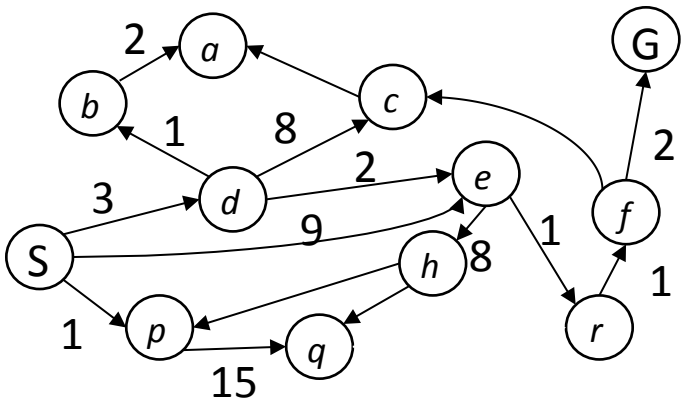
Notice that BFS finds the shortest path in terms of number of transitions. It does not find the least-cost path.

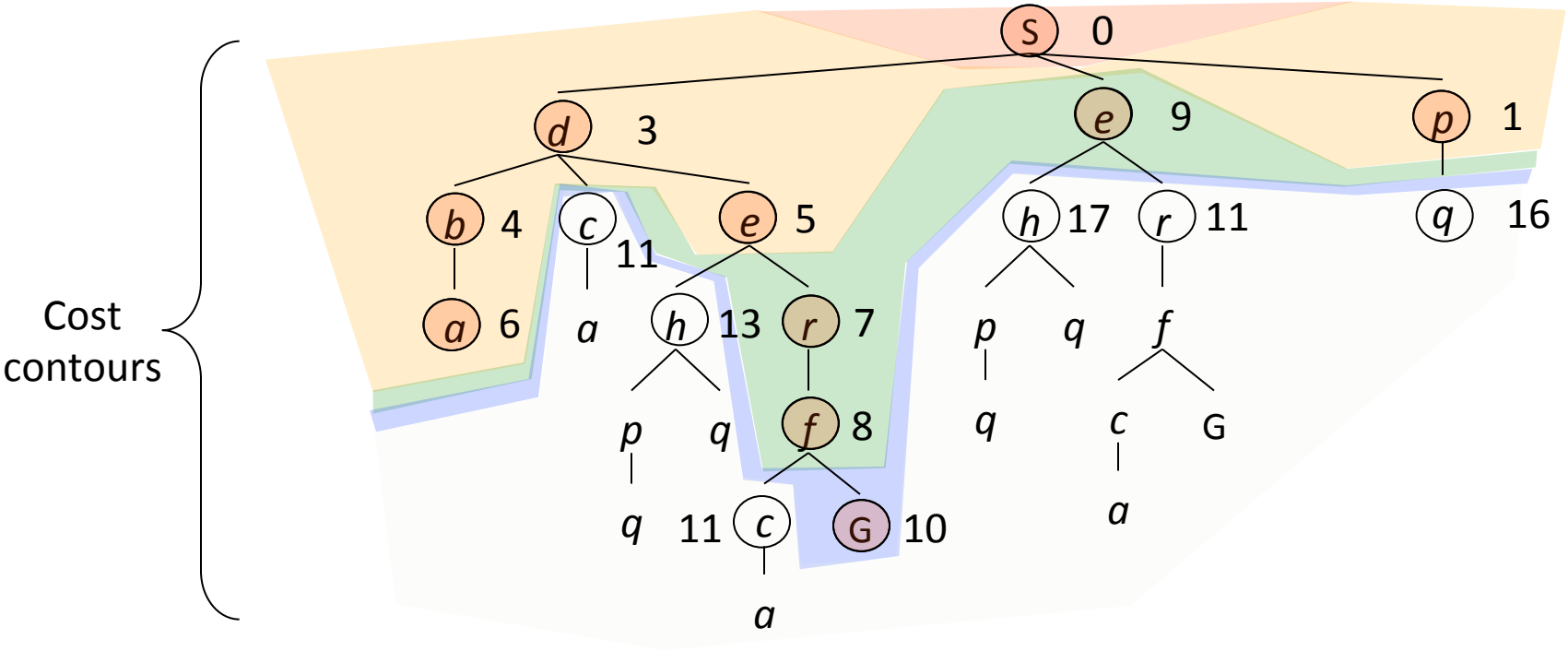
We will quickly cover an algorithm which does find the least-cost path.

Uniform Cost Search

Expand cheapest node first:

Fringe is a priority queue





Priority Queue Refresher

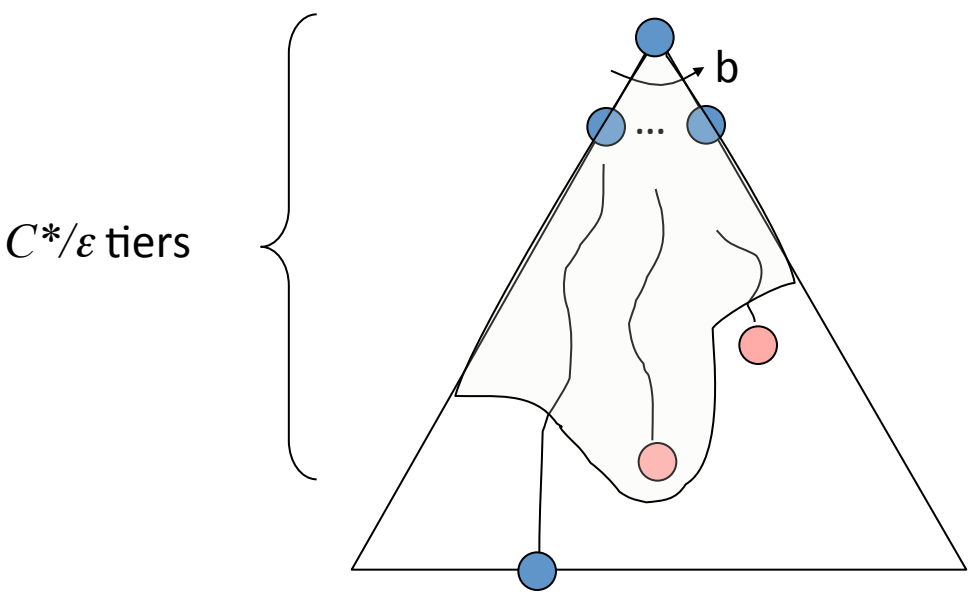
- A priority queue is a data structure in which you can insert and retrieve (key, value) pairs with the following operations:

pq.push(key, value)	inserts *(key, value)* into the queue.
pq.pop()	returns the key with the lowest value, and removes it from the queue.

- You can decrease a key's priority by pushing it again
- Unlike a regular queue, insertions aren't constant time, usually O(log *n*)
- We'll need priority queues for cost-sensitive search methods

Uniform Cost Search

Algorithm		Complete	Optimal	Time	Space
DFS	w/ Path Checking	Y	N	$O(b^{m+1})$	$O(bm)$
BFS		Y	N	$O(b^{s+1})$	$O(b^{s})$
UCS		Y*	Y	$O(b^{C^*/\varepsilon})$	$O(b^{C^*/\varepsilon})$



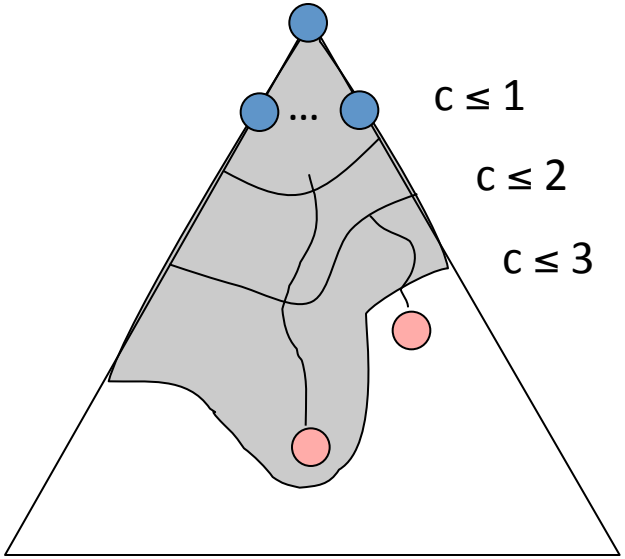
** UCS can fail if actions can get arbitrarily cheap*

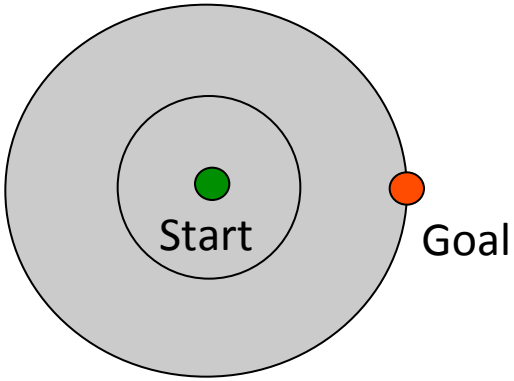
Done with Search?

UCS!

Uniform Cost Issues

- Remember: explores increasing cost contours
- The good: UCS is complete and optimal!
- The bad:
  - Explores options in every "direction"
  - No information about goal location





[demo: search demo empty]

- http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/mlbook/ch9.pdf