Local Search & Games

CSCI 4511/6511

Joe Goldfrank

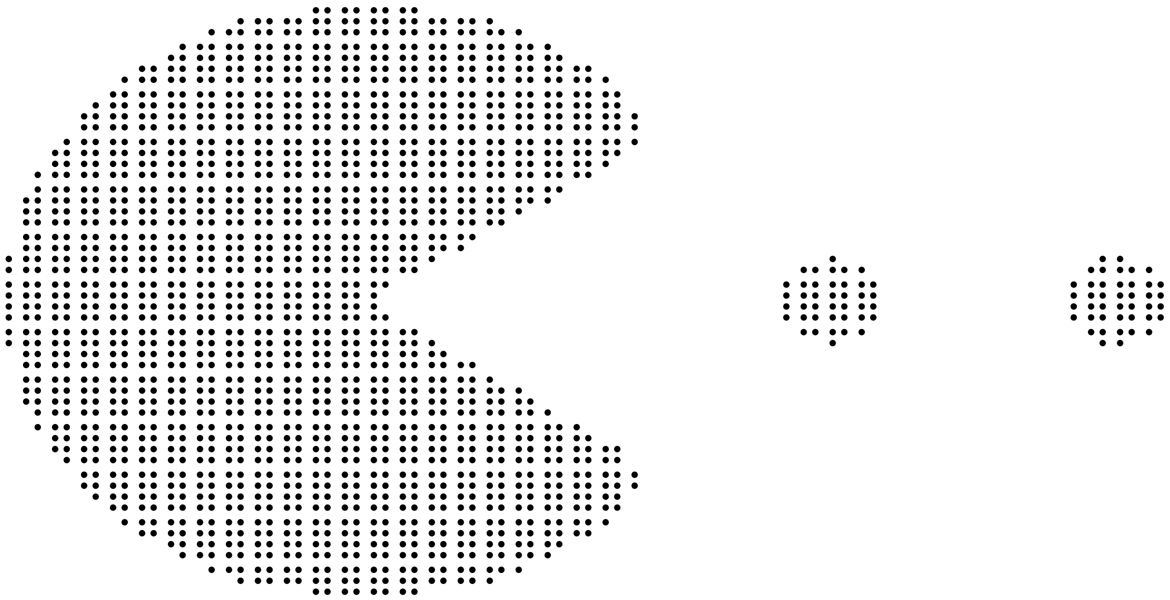
Announcements

- Homework 1 is due on 15 September at 11:55 PM
  - Late submission policy
- Homework 2 is due on 29 September at 11:55 PM
- Fri 13 Sep Office Hours moved: 12 PM - 3 PM
- Fri 20 Sep Office Hours moved: 12 PM - 3 PM

Review

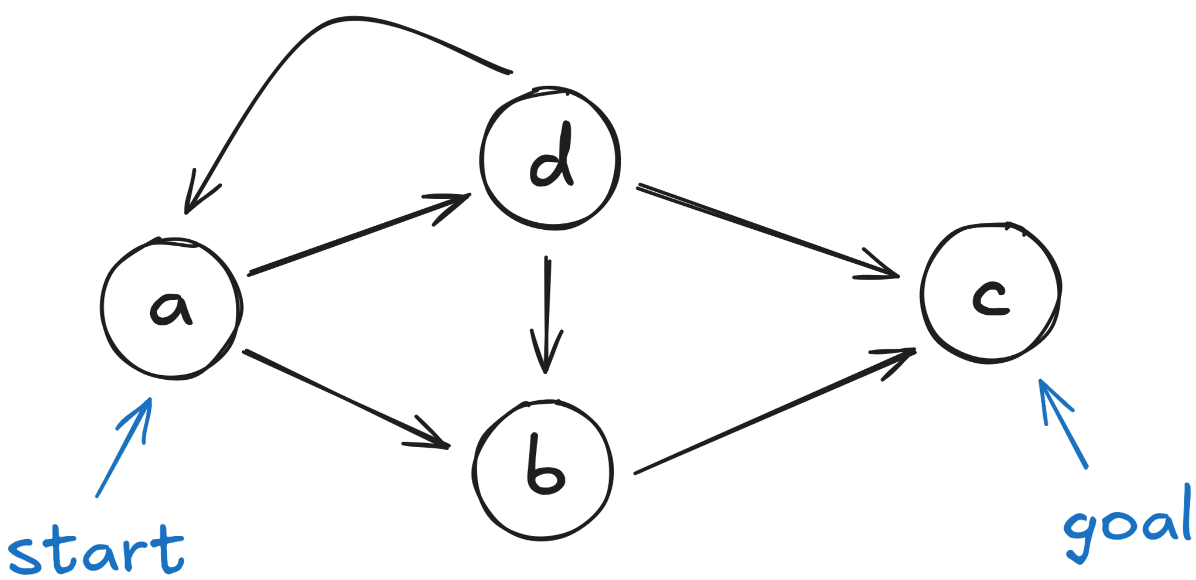
Why Are We Here?

Why Are We Here?

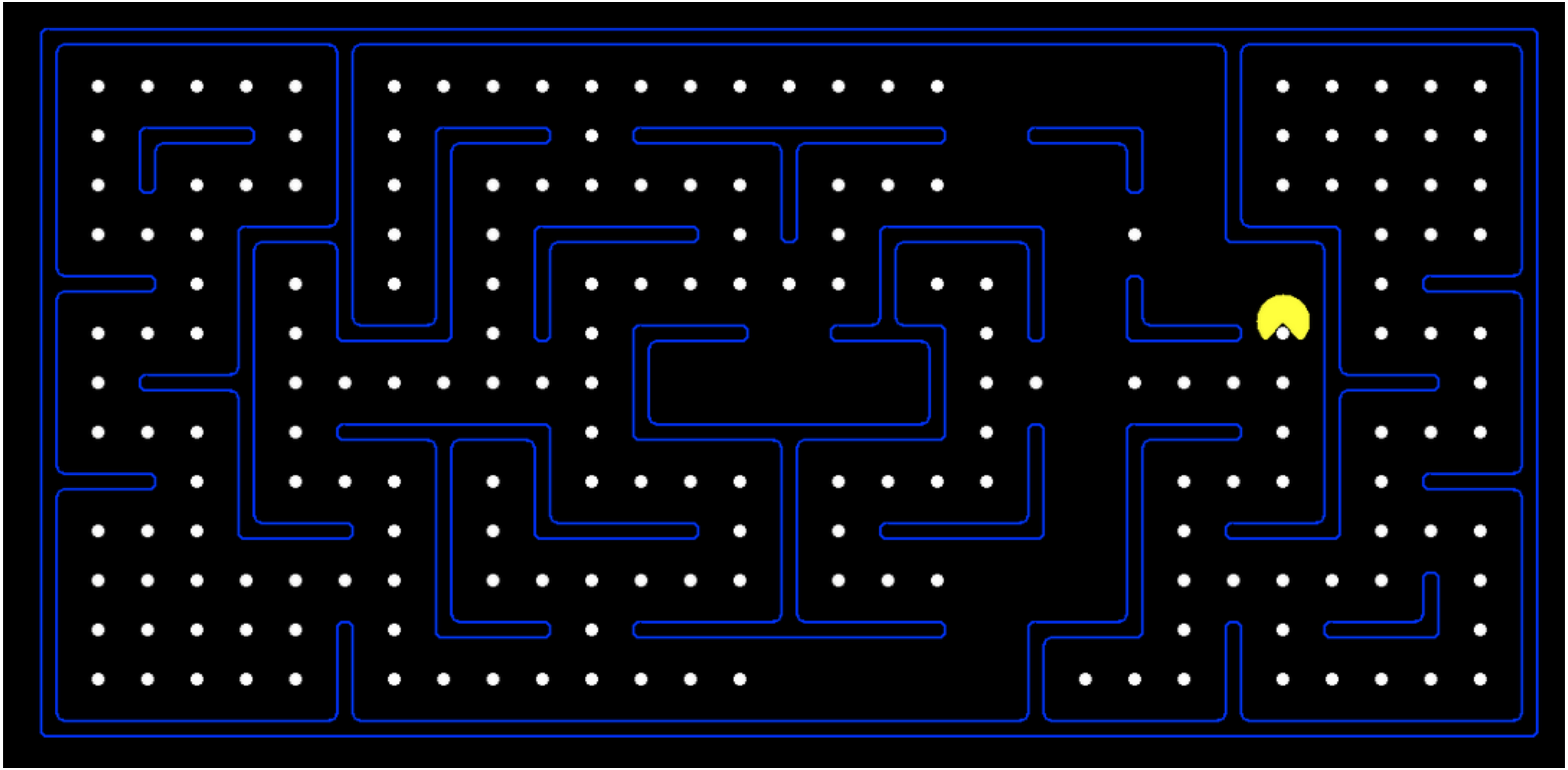


Search: Why?

- Fully-observed problem
- Deterministic actions and state
- Well-defined *start* and *goal*
  - "Well-defined"



Goal Tests



Goal Tests

Best-First Search

Algorithm Best-First Search

```
function BEST-FIRST-SEARCH(problem, f)
    node ← NODE(STATE=problem.INITIAL)
    frontier ← priority queue ordered by f
    frontier.ADD(node)
    reached ← lookup table
    reached[node] ← problem.INITIAL
    while not IS-EMPTY(frontier) do
        node ← POP(frontier)
        if problem.IS-GOAL(node.STATE) then
            return node
        for each child in EXPAND(problem,node) do
            s ← child.STATE
            if not s ∈ reached or child.PATH-COST < reached[s].PATH-COST then
                reached[s] ← child
                frontier.ADD(child)
    return failure

function EXPAND(problem, node)
    s ← node.STATE
    for each action in problem.ACTIONS(s) do
        s′ ← problem.RESULT(s, action)
        cost ← node.PATH-COST +problem.ACTION-COST(s, action, s′)
        yield NODE(STATE= s′, PARENT=node, ACTION=action, PATH-COST=cost)
```

A* Search

- Include path-cost $g(n)$
  - $f(n) = g(n) + h(n)$

Algorithm A* Search

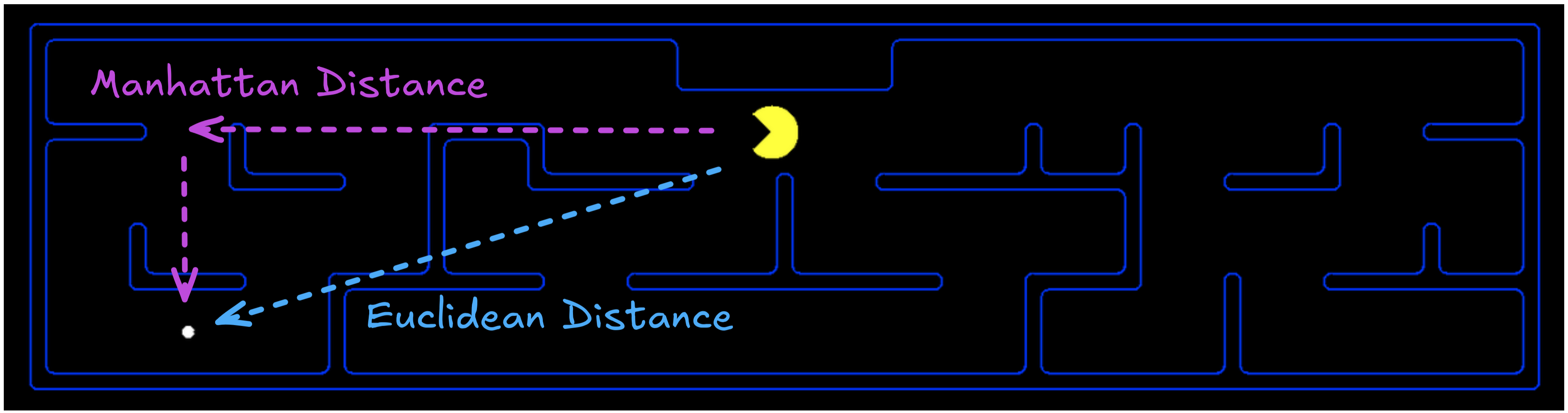
```
function A*-SEARCH(problem)
    return BEST-FIRST-SEARCH(problem, g(n) + h(n))
```

- Complete (always)
- Optimal (sometimes)
- Painful $O(b^m)$ time and space complexity

A* vs. Dijkstra

Choosing Heuristics

- Recall: $h(n)$ estimates cost from n to goal



- Admissibility
- Consistency

Choosing Heuristics

- Admissibility
  - *Never* overestimates cost from n to goal
  - Cost-optimal!
- Consistency
  - $h(n) \leq c(n, a, n') + h(n')$
  - n' successors of n
  - $c(n, a, n')$ cost from n to n' given action a

Iterative-Deepening A* Search

"IDA*" Search

- Similar to Iterative Deepening with Depth-First Search
  - DFS uses depth cutoff
  - IDA* uses $h(n) + g(n)$ cutoff *with DFS*
  - Once cutoff breached, new cutoff:
    - Typically next-largest $h(n) + g(n)$
  - $O(b^m)$ time complexity 😔
  - $O(d)$ space complexity 😌

Beam Search

Best-First Search:

- Frontier is all expanded nodes

Beam Search:

- k "best" nodes are kept on frontier
  - Others discarded
- Alt: all nodes within δ of best node
- Not Optimal
- Not Complete

Recursive Best-First Search (RBFS)

- No *reached* table is kept
- Second-best node $f(n)$ retained
  - Search from each node cannot exceed this limit
  - If exceeded, recursion “backs up” to previous node
- Memory-efficient
  - Can “cycle” between branches

Recursive Best-First Search (RBFS)

Algorithm Recursive Best-First Search

```
function RECURSIVE-BEST-FIRST-SEARCH(problem)
    solution, f_value ← RFBS(problem,NODE(problem.INITIAL), ∞)
    return solution

function RBFS(problem, node, f_limit)
    if problem.IS-GOAL(node.STATE) then
        return node
    successors ← LIST(EXPAND(node))
    if IS-EMPTY(successors) then
        return failure, ∞
    for each s in successors do
        s.f ← MAX(s.PATH-COST + h(s), node.f)
    while True do
        best ← node in successors with lowest f
        if best.f > f_limit then
            return failure, best.f
        alternative ← node in successors with second-lowest f
        result, best.f ← RBFS(problem, best,MIN(f_limit, alterative))
        if result ≠ failure then
            return result, best.f
```

Heuristic Characteristics

- What makes a “good” heuristic?
  - We know about admissability and consistency
  - What about performance?
- Effective branching factor
- Effective depth
- # of nodes expanded

Where Do Heuristics Come From?

- Intuition
  - "Just Be Really Smart"
- Relaxation
  - The problem is constrained
  - Remove the constraint
- Pre-computation
  - Sub problems
- Learning

Local Search

What Even Is The Goal?

Uninformed/Informed Search:

- Known start, known goal
- Search for optimal path

Local Search:

- "Start" is irrelevant
- Goal is not known
  - But we know it when we see it
- Search for *goal*

Brutal Example

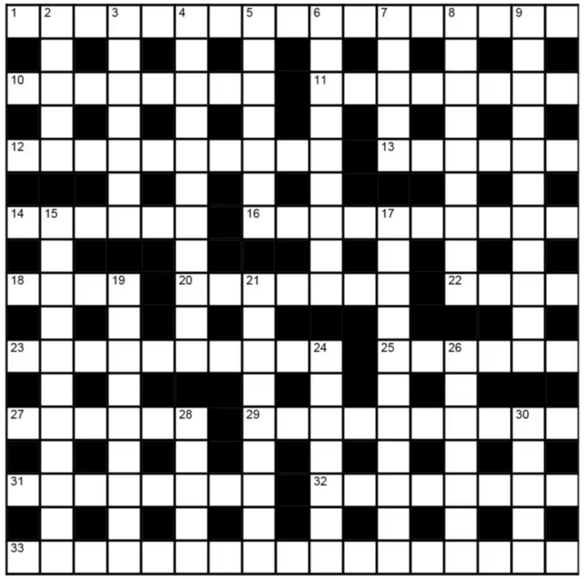
POLYMATH 1,296 by SLEUTH

ACROSS

1 Bushy male sideburns popular during the Victorian period (10,7)
10 Indian city in which snooker is thought to have originated (8)
11 Nickname of King John of England due to his poor inheritance (8)
12 A bishop's move in chess to control the board's long diagonal (10)
13 1986 horror film starring Jeff Goldblum as scientist Seth Brundle (3,3)
14 Port city in western Saudi Arabia where pilgrims land for the haj (6)
16 A set of principles to do with the nature and appreciation of beauty (10)
18 ___ Knievel, US daredevil showman and stunt rider (4)
20 Historic part of North Yorkshire that contains the market town of Malton (7)
22 Dannie ___, Welsh poet and physician born in 1923 (4)
23 Athletics event for which Jonathan Edwards holds the world record (6,4)
25 Large, fish-eating raptor that is brown on its upper parts (6)
27 Altered ___, new wave band whose lead vocalist is Clare Grogan (6)
29 Town in north Hertfordshire that was Britain's first garden city (10)
31 Tending to intrude on a person's thoughts or privacy (8)
32 One who improvises lines or a speech (2-6)
33 Fourth studio album by The Police released in 1981 (5,2,3,7)

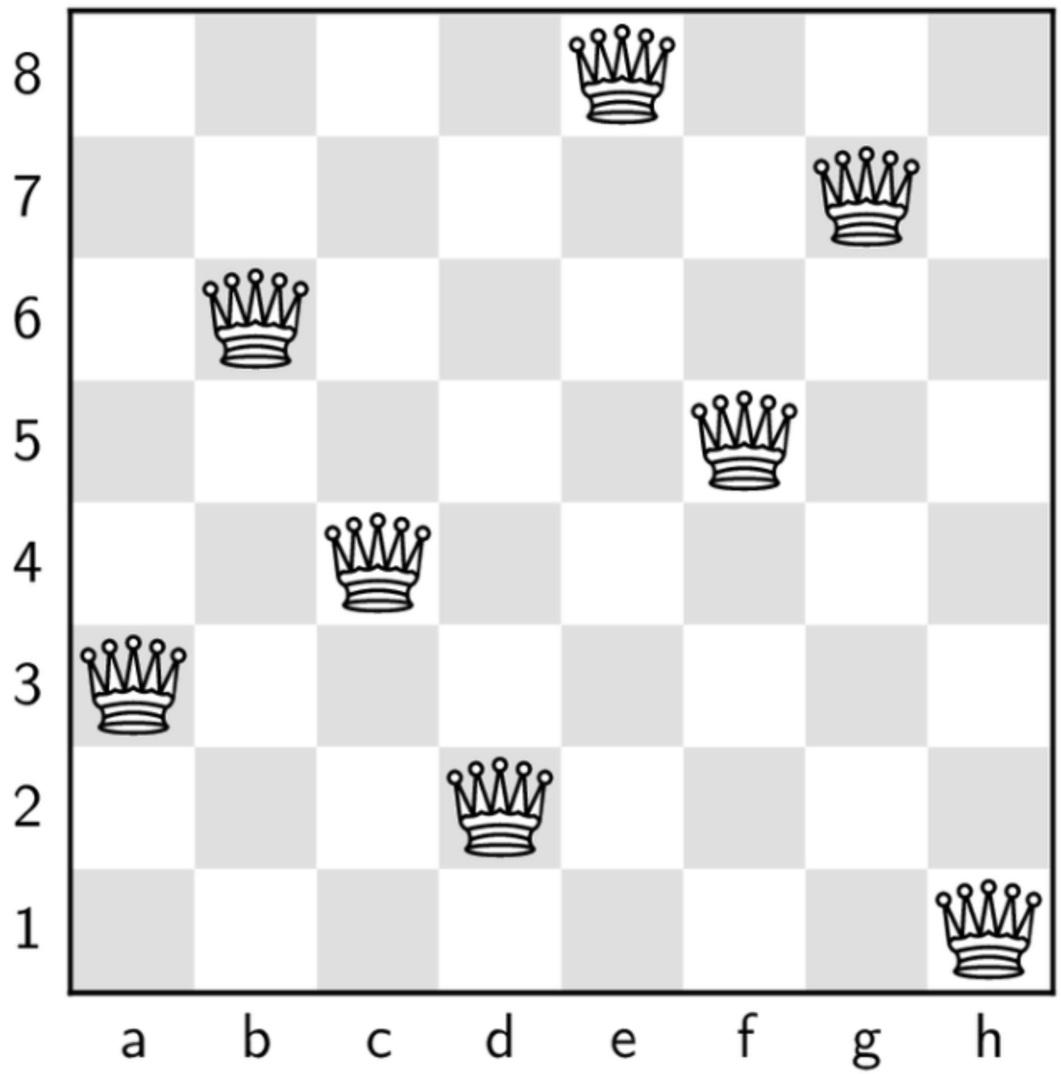
DOWN

2 Ronnie ___, English cricket all-rounder who played in 31 ODI matches (5)
3 Irish band formed in 1970 who fused folk, rock and new age (7)
4 Explosive dropped from a ship or aircraft to attack a submarine (5,6)
5 ___ Lynn, singer who was subject of the film *Coal Miner's Daughter (7)*
6 Body of water between mainland China and the Korean peninsula (6,3)
7 Theme park near Orlando in Florida that opened in 1982 (5)
8 Gymnasium or wrestling school in ancient Greece and Rome (9)
9 French tennis player born in 1904 nicknamed The Crocodile (4,7)
15 Norwegian artist noted for his Frieze of Life series (6,5)
17 Hereditary disorder that affected the Romanov dynasty in Russia (11)
19 Compositions in which a writer omits a certain letter of the alphabet (9)
21 Cheerful and highly energetic (9)
24 A thick meat or vegetable soup (7)
26 The ultimate ruler in Gilbert & Sullivan's operetta *The Mikado (4-3)*
28 Genre of literature for which the annual Hugo Awards are given (3-2)
30 Small domestic wooden objects, especially antiques (5)



Solution 1,295

Less-Brutal Example



“Real-World” Examples

- Scheduling
- Layout optimization
  - Factories
  - Circuits
- Portfolio management
- Others?

Objective Function

- Do you know what you want?[1]
- Can you express it mathematically?[2]
  - A single value
  - More is better
- Objective function: a function of *state*

1. If not, you might be human
2. If not, you might be human

Hill-Climbing

- Objective function
- State space mapping
    - Neighbors

Algorithm Hill-Climbing

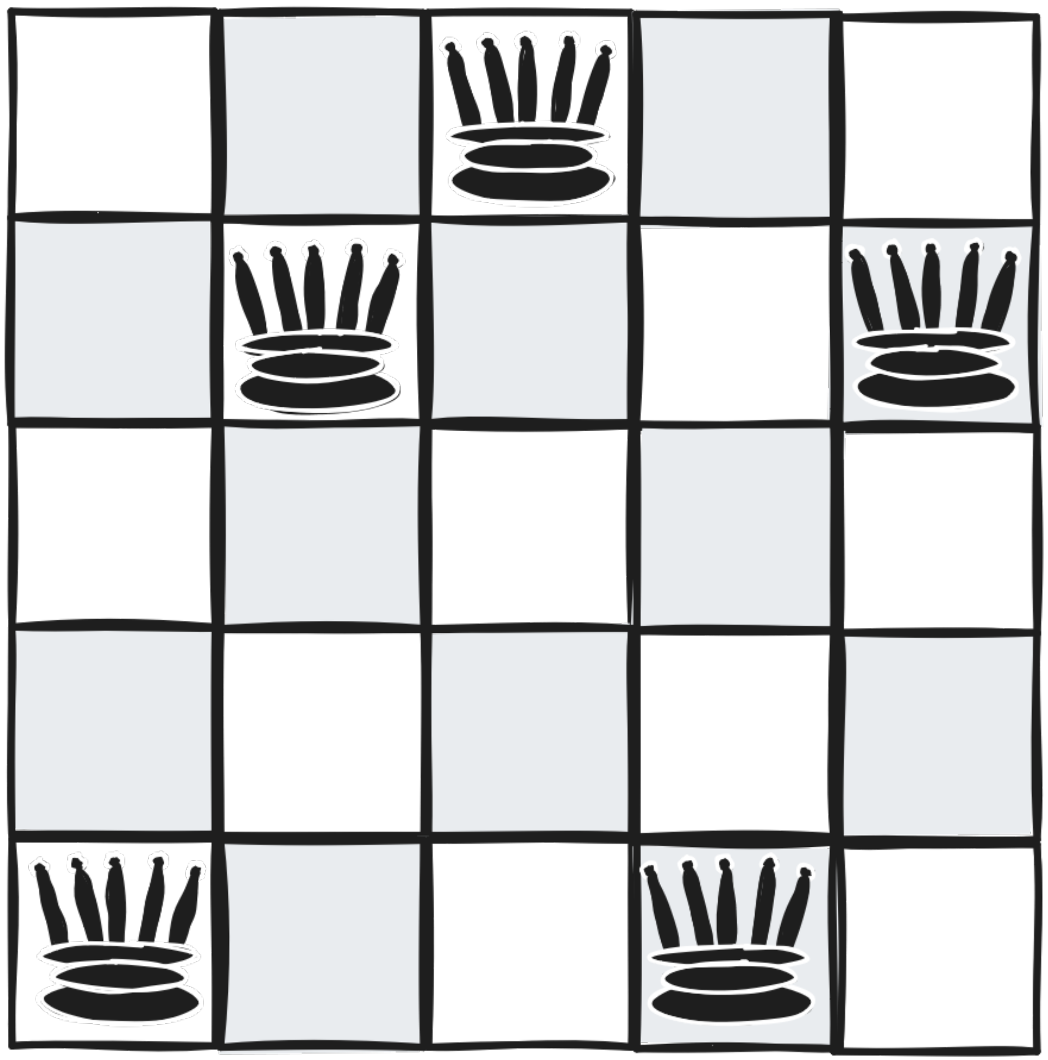
```
function HILL-CLIMBING(problem)
    current ← problem.INITIAL
    while True do
        neighbor ← successor of current with greatest objective function value
        if VALUE(neighbor) ≤ VALUE(current) then
            return current
        current ← neighbor
```

Hill-Climbing

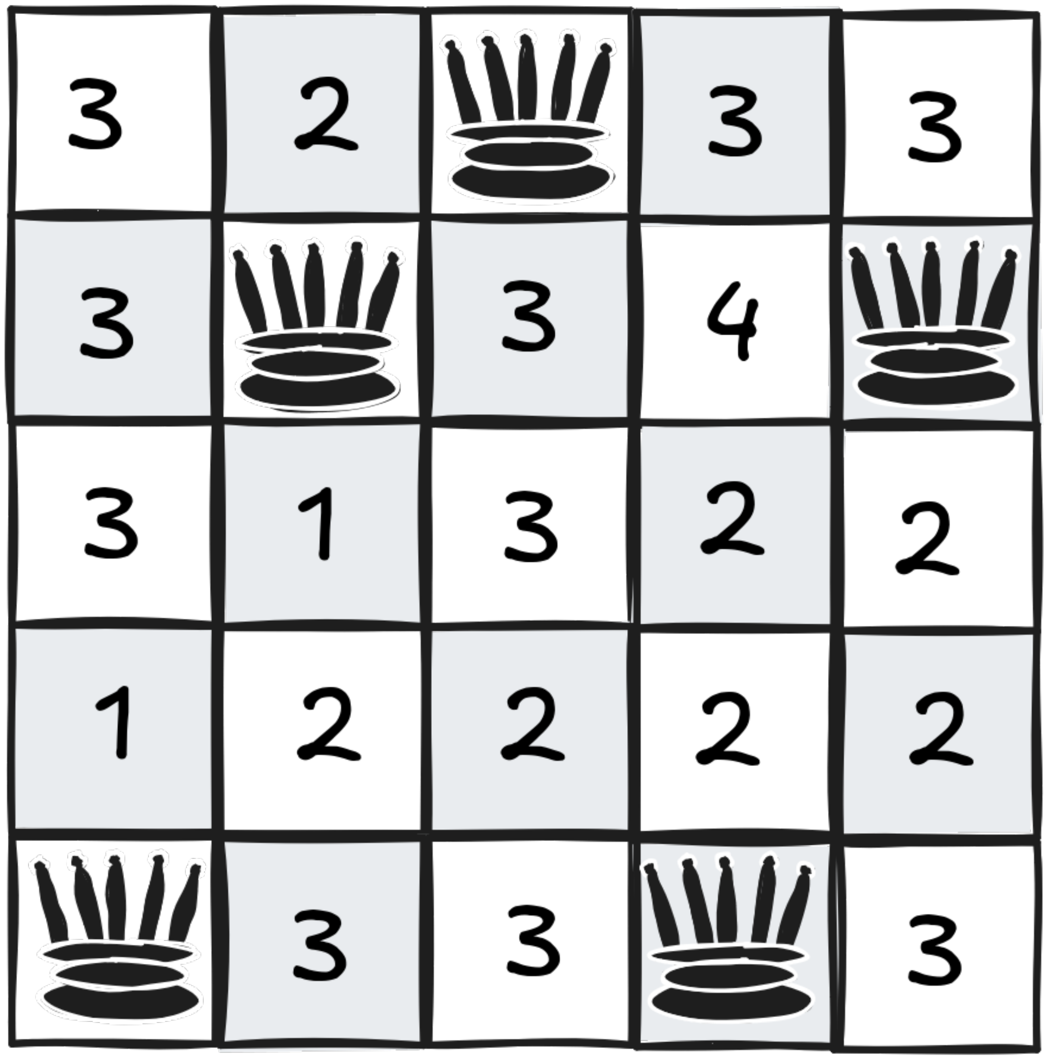
The Hazards of Climbing Hills

- Local maxima
- Plateaus
- Ridges

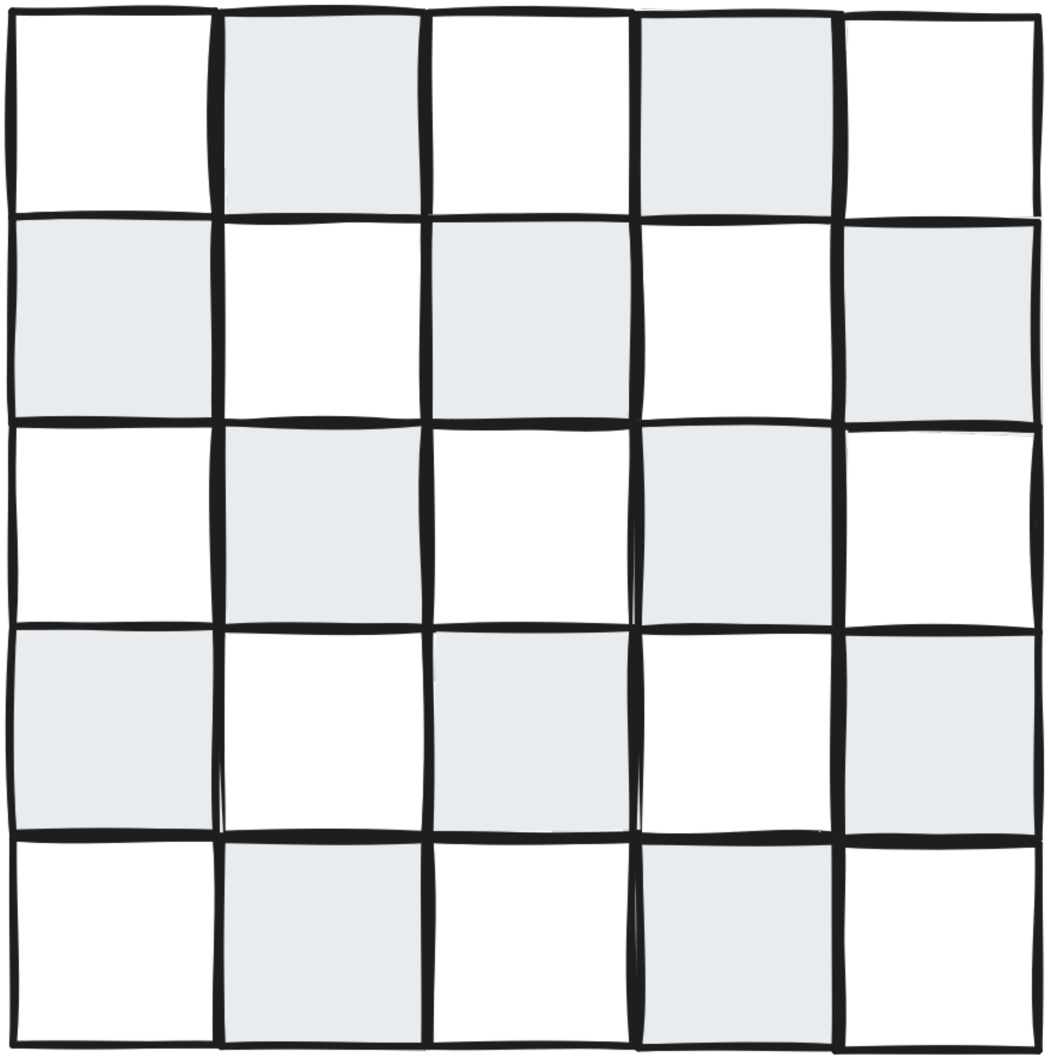
Five Queens



Five Queens



Five Queens



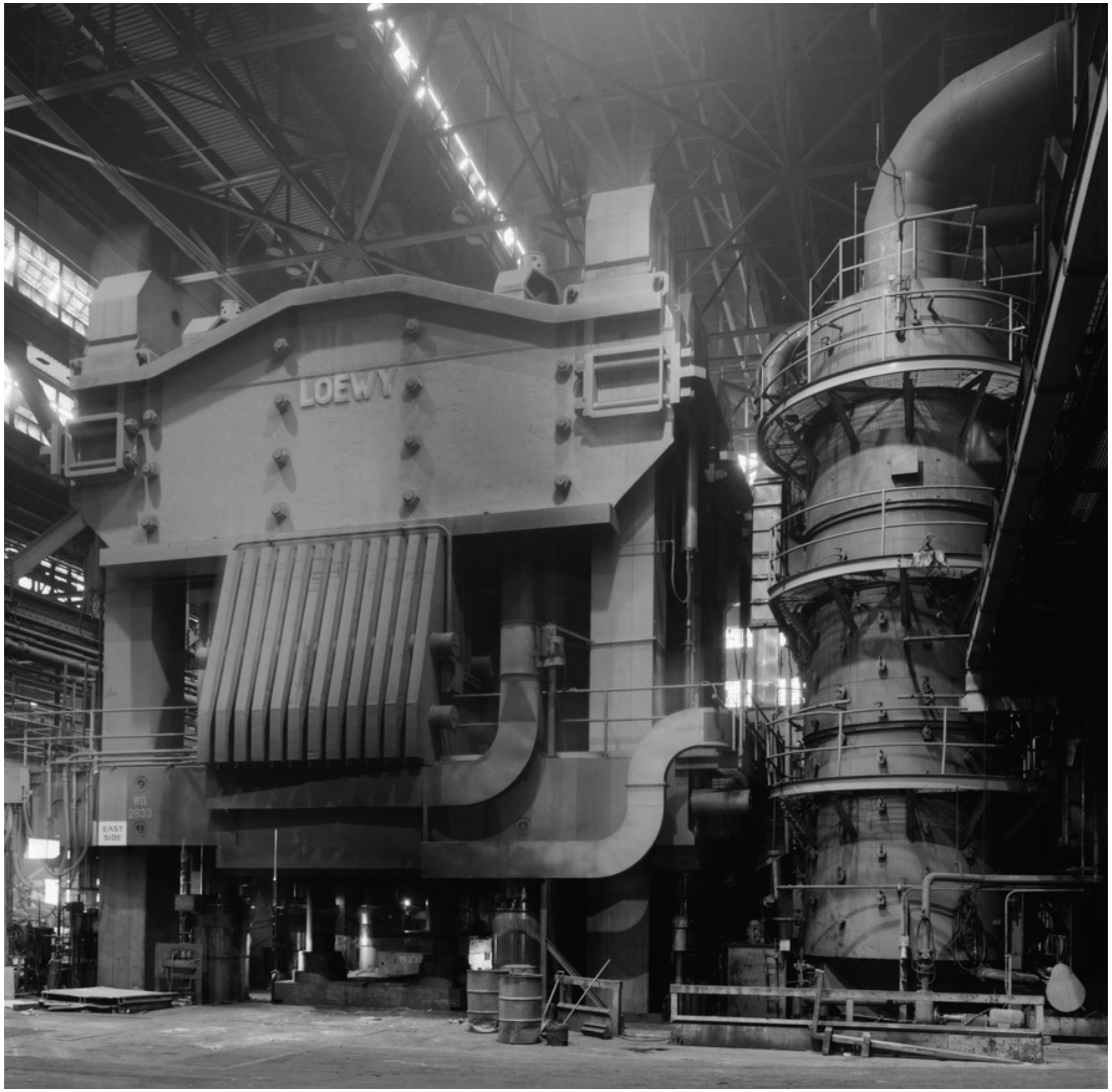
Variations

- Sideways moves
  - Not free
- Stochastic moves
  - Full set
  - First choice
- Random restarts
  - If at first you don't succeed, ~~you fail~~ try again!
  - Complete 😌

The Trouble with Local Maxima

- We don’t know that they’re local maxima
  - Unless we do?
- Hill climbing is efficient
  - But gets trapped
- Exhaustive search is complete
  - But it’s exhaustive!
  - Stochastic methods are ‘exhaustive’

Simulated Annealing



Simulated Annealing

- Doesn’t actually have anything to do with metallurgy
- Search begins with high “temperature”
  - Temperature decreases during search
- Next state selected randomly
  - Improvements always accepted
  - Non-improvements rejected stochastically
  - Higher temperature, less rejection
  - “Worse” result, more rejection

Simulated Annealing

Algorithm Simulated Annealing

```
function SIMULATED-ANNEALING(problem, current)
    current ← problem.INITIAL
    t ← 1
    while True do
        T ← schedule(t)
        if T = MIN(schedule) then
            return current
        next ← random successor of current
        ΔE ← VALUE(current) - VALUE(next)
        if ΔE > 0 then
            current ← next
        else
            p ← sample from U(0, 1)
            if p < e^(−ΔE/T) then
                current ← next
```

Local Beam Search

Recall:

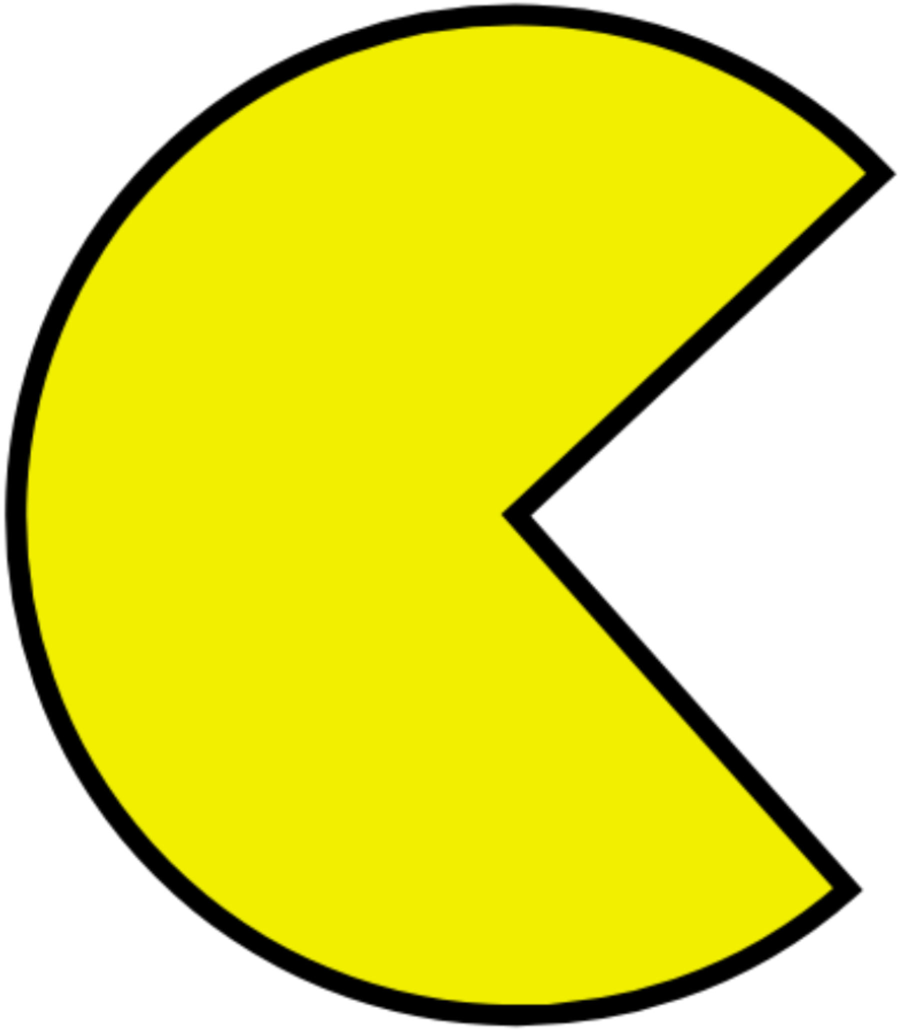
- Beam search keeps track of k “best” branches

Local Beam Search:

- Hill climbing search, keeping track of k successors
  - Deterministic
  - Stochastic

Local Beam Search

The Real World Is Discrete



(it isn't)

The Real World Is Not Discrete

- Discretize continuous space
  - Works iff no objective function discontinuities
  - What happens if there are discontinuities?
  - How do we know that there are discontinuities?

Gradient Descent

- Minimize loss instead of climb hill
  - Still the same idea

Consider:

- One state variable, x
- Objective function $f(x)$
  - How do we minimize $f(x)$?
  - Is there a closed form $\frac{d}{dx}$?

Gradient Descent

Multivariate $\vec{x} = x_0, x_1, \ldots$

Instead of derivative, gradient:

$$\nabla f(\vec{x}) = \left[\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial x_1}, \ldots \right]$$

"Locally" descend gradient:

$$\vec{x} \leftarrow \vec{x} + \alpha \nabla f(\vec{x})$$

Games

Adversity

So far:

- The world does not care about us
- This is a simplifying assumption!

Reality:

- The world does not care us
- …but it wants things for “itself”
- …and we don’t want the same things

The Adversary

One extreme:

- Single adversary
  - Adversary wants the *exact opposite* from us
  - If adversary “wins,” we lose

😐

Other extreme:

- An entire world of agents with different values
  - They might want some things similar to us
- “Economics”

😐

Simple Games

- Two-player
- Turn-taking
- Discrete-state
- Fully-observable
- Zero-sum
  - This does some work for us!

Max and Min

- Two players want the opposite of each other
- State takes into account both agents
  - Actions depend on whose turn it is

Minimax

- Initial state s_0
- ACTIONS(s) and TO-MOVE(s)
- RESULT(s, a)
- IS-TERMINAL(s)
- UTILITY(s, p)

Minimax

Minimax

Algorithm Minimax Search

```
function MINIMAX-SEARCH(game, state)
    player ← game.TO-MOVE(state)
    value, move ← MAX-VALUE(game, state)
    return move

function MAX-VALUE(game, state)
    if game.IS-TERMINAL(state) then
        return game.UTILITY(state, player),null
    v ← −∞
    for each a in game.ACTIONS(state) do
        v2, a2 ←MIN-VALUE(game, game.RESULT(state, a))
        if v2 > v then
            v, move ← v2, a
    return v, move

function MIN-VALUE(game, state)
    if game.IS-TERMINAL(state) then
        return game.UTILITY(state, player),null
    v ← ∞
    for each a in game.ACTIONS(state) do
        v2, a2 ←MAX-VALUE(game, game.RESULT(state, a))
        if v2 < v then
            v, move ← v2, a
    return v, move
```

More Than Two Players

- Two players, two values: v_A, v_B
  - Zero-sum: $v_A = -v_B$
  - Only one value needs to be explicitly represented
- >2 players:
  - $v_A, v_B, v_C \ldots$
  - Value scalar becomes $\vec{v}$

Society

- >2 players, only one can win
- Cooperation can be rational!

Example:

- A & B: 30% win probability each
- C: 40% win probability
- A & B cooperate to eliminate C
  - $\rightarrow$ A & B: 50% win probability each

…what about friendship?

Minimax Efficiency

Pruning removes the need to explore the full tree.

- Max and Min nodes alternate
- Once *one* value has been found, we can eliminate parts of search
  - Lower values, for Max
  - Higher values, for Min
- Remember highest value (α) for Max
- Remember lowest value (β) for Min

Pruning

Heuristics 😌

- In practice, trees are far too deep to completely search
- Heuristic: replace utility with evaluation function
  - Better than losing, worse than winning
  - Represents chance of winning
- Chance? 🎲🎲
  - Even in deterministic games
  - Why?

More Pruning

- Don't bother further searching bad moves
  - Examples?
- Beam search
  - Lee Sedol's singular win against AlphaGo

Other Techniques

- Move ordering
  - How do we decide?
- Lookup tables
  - For subsets of games

Monte Carlo Tree Search

- Many games are too large even for an efficient α-β search 😔
    - We can still play them
- *Simulate* plays of entire games from starting state
    - Update win probability from each node (for each player) based on result
- "Explore/exploit" paradigm for move selection

Choosing Moves

- We want our search to pick good moves
- We want our search to pick unknown moves
- We *don’t* want our search to pick bad moves
    - (Assuming they’re actually bad moves)

Select moves based on a heuristic.

Games of Luck

- Real-world problems are rarely deterministic
- Non-deterministic state evolution:
  - Roll a die to determine next position
  - Toss a coin to determine who picks candy first
  - Precise trajectory of kicked football[1]
  - Others?

1. Any definition of "football"

Solving Non-Deterministic Games

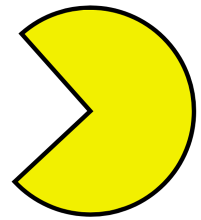
Previously: Max and Min alternate turns

Now:

- Max
- Chance
- Min
- Chance

😣

Expectiminimax

- “Expected value” of next position 

- How does this impact branching factor of the search?

Expectiminimax

Filled With Uncertainty

What is to be done?

- Pruning is still possible
  - How?
- Heuristic evaluation functions
  - Choose carefully!

Non-Optimal Adversaries

- Is deterministic “best” behavior optimal?
- Are all adversaries rational?

- Expectimax

References

- Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4th Edition, 2020.
- Mykal Kochenderfer, Tim Wheeler, and Kyle Wray. *Algorithms for Decision Making*. 1st Edition, 2022.
- Stanford CS231
- Stanford CS228
- UC Berkeley CS188