UE Artificial Intelligence

344.021, 344.022, 344.023
WS 2015

Filip Korzeniowski, Rainer Kelz

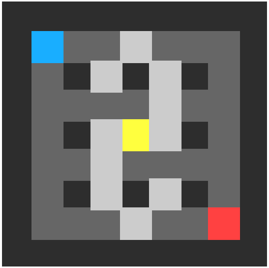
Department of Computational Perception
Johannes Kepler University
Linz, Austria

JOHANNES KEPLER UNIVERSITÄT LINZ

Department of Computational Perception

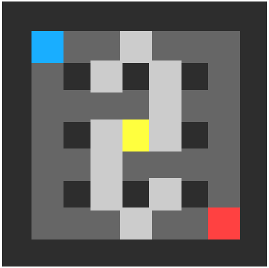
October 12, 2015

A short example - “What if ... ?”



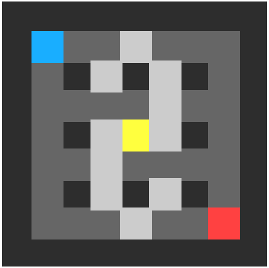
- assuming we wanted to search the state space of the 1 vs. 1 game depicted on the left by exhaustive search - to find the best moves simply by enumerating all possible game states
- to do this, we first have to determine **how many bits** we need to specify **one state**
- 41 paths, 10 clouds, 1 fountain, 2 unicorns, 3 seeds each, move limit is 50

Unicorn position



- if we simply index the walkable tiles (paths), the position of the unicorns can be represented by such an index value
- we need to be able to distinguish 41 positions for each unicorn
- for 2 unicorns, that's $\log_2(41^2)$ = **11 bits**
- the adjacency matrix for indices needs to be remembered only once
- if we would have included the non-walkable tiles, we would have needed $\log_2((9^2)^2) = 13$ bits

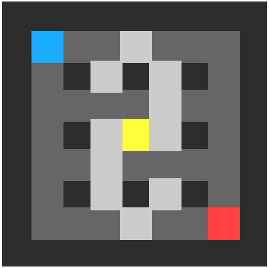
Seed position



- the position of seeds can be represented by an index as well

- a seed has a fuse length of 8
- naively, for 2 unicorns and 3 seeds each, that makes $\log_2(41^6 + 8^6) =$ **33 bits**
- "naively" because we did not take into account the effect that the fuse timing has on seed position - constraining the maximum distance a seed may have from a unicorn

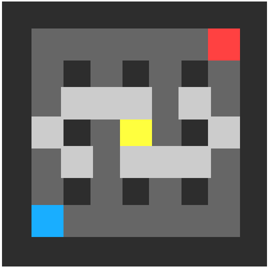
Rainbow positions



- again, the position of rainbows can be represented by an index

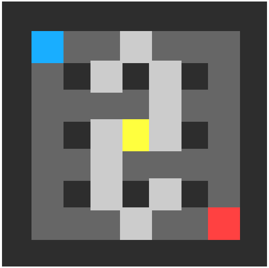
- but in this case, we only need to check which walkable positions are affected
- a seed blossoms into 9 rainbow tiles, so modelling them the same as we did the seeds would be wasteful ($\log_2 41^{2\cdot3\cdot9} = 290$ bits)
- instead, we'll use exactly **41 bits** because we do not care about overlap

Cloud positions



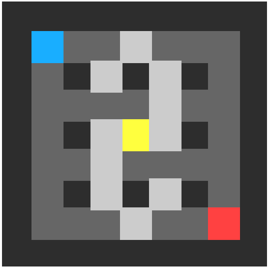
- clouds are a bit different again, insofar as we only need to store whether a walkable tile is blocked
- naively 1 bit for each possible position - 41 bits
- actually, we only need enough bits to model the maximum index - if we turn the board, this saves a few bits
- in this example case, we only need **29 bits** if we start in the upper left corner with indexing

Fountains (1)



- we need be able to store scores for all unicorns on the board, for each fountain
- the maximum score possible is the move limit (minus the length of the path in state space from starting position to the fountain in question)
- the path length is 6 moves for a single unicorn, the unicorns take turns, for a path length in state space of 12, the move limit is 50, which (naively) makes a maximum achievable score of 38

Fountains (2)



- naively, because we do not model the possibilities that the other unicorn stands on the same fountain (no golden stars, for either unicorn)
- we need to keep track of scores for 2 unicorns, and 1 fountain
- $\log_2 38^2$ = **11 bits**

How many bits to describe a state ?

Hairy approximations up ahead ...

Unicorns	11 bits
Seeds	33 bits
Rainbows	41 bits
Clouds	29 bits
Fountains	11 bits
Sum	125 bits

- We have 2^{125} possible boards
- Assuming we can compute a score in 50 instructions on an (overclocked) 5GHz machine

@1GHz, 1 instr., T = 1 [ns]

@5GHz, 1 instr., T = 0.2 [ns]

@5GHz, 50 instr., T = 10 [ns]

- $2^{125} \cdot 10 \cdot 10^{-9}$ [s]
- $60 \cdot 60 \cdot 24 \cdot 356$ [s] per year
- $\approx \mathbf{1.3 \cdot 10^{22}}$ **[years]** to exhaustively search the state space!
- $\frac{2^{125}}{1000^8} \approx \mathbf{42 \cdot 10^{12}}$ **[YB]** to store the search space

Perspective

- "an internet" was roughly $2.4 \cdot 10^{21}$ [bits] in 2007
- our state space is equivalent to $1.6 \cdot 10^{16}$ [internets]
- the information capacity of the observable universe is 2^{305} [bits]
- our state space would then be a $\frac{1}{2^{180}}$ th [universe]
- our state space is smaller than Chess, with about 10^{47} states
- our state space is smaller than Go, with about 10^{171} states

Assignment 0 - Proper Formatting

- the zeroth assignment exists to make sure you know how to properly format the **zip** archive that you have to hand in for each assignment
- make sure you and your partner **both** know how to do it
- once you created the archive, **unpack** it in an empty folder to actually **check** you did it correctly

Assignment 1 - Search in Graphs

- uninformed search (BFS, DLDFS, IDS)
- cost sensitive search (UCS)
- heuristic search (GBFS)
- cost and heuristic search (ASTAR)
- search in **graphs** means you have to keep a datastructure around that tells you which nodes you already expanded
- the **asymptotic runtime complexity** of the **insert** and **lookup** operations on this datastructure are **important**

Search in Graphs

- we want to avoid doing more work than necessary
- especially if we have a “loopy” state space
- there are three basic possibilities

Cycle Avoidance - Tradeoffs

- do not expand **parent** node again (trivial, basically free)
- do not expand a node from the **current path** again (“self-avoiding walk”)
- do not expand **any visited** node again (“closed list”)

Organizational Stuff

- Assignments 0 & 1 will be uploaded this afternoon
- You’ll receive your groupnames this afternoon / evening as well

Questions

?