Candy Crisis

Comp 472
Team Alpha

Heuristic - Machine Learning (failed attempt)

- Training data: Randomly generated boards solved with iterative deepening (~100 000 data points)
- Classifier: neural networks → ~55% accuracy
- Input data: board vector where every candy is mapped to an integer
  - [0, 1, 1, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6]
- Predictions: right, left, up, down
- Problem: Training data isn't solved methodically therefore classifier fails to find hidden links. It takes many moves to solve a board.
  - Example: In certain situations going left or right are almost as good. Game never halts because of circular patterns. No clear convergence for a solution
- Certain vectors are similar but have very different solutions. Classifier fails to spot major differences thus causing excessive moves to find solution

Heuristic - Pairwise distance (Rule of thumbs)

- Best case scenario: Database of solution for every state permutation.
  - Unfeasible → ~ 15! / 2 permutations or ~ 653 837 184 000 different states
- Therefore Complex puzzle tend to be solved methodically using algorithms and mathematical theorems. Example: rubik's cube, 8 puzzle, 2048…
- Hypothesis: Create a database of move sequences.
  - Approach would be methodical → guarantees a solution
  - Heuristic is admissible
- Only care about empty square, solved squares, and pair to be solved
- New mapping:
  - -1: pieces we don't care about (unsolved pairs and middle pieces)
  - 0: empty square
  - 1: pair to be solved
  - 2: solved pair
- New board representation only has 1320 permutations! → brute force feasible
  - Permutations scaled down from ~10^11 to ~10^3 → Reduce power of magnitude by 8!

Heuristic - Pairwise distance

- Each board state is transformed to it's representative pattern.
- Every pattern is solved using iterative deepening and then stored in a Pattern database. (~ 170 hours to brute force)
- Example: We want to solve for the (y, y) pair then the transformation is as follows:

e	b	y	r	y
w	g	g	r	r
w	b	b	r	b

➡

0	2	1	2	1
-1	-1	-1	-1	-1
-1	2	-1	2	-1

- The transformations T(x) = x', where x is the original board representation on the left, is mapped to x' which constitutes 1 of 1320 patterns.

Pairwise distance - Strength and Weakness

- Pros: **1)** Heuristic is admissible, **2)** Path is short, **3)** Evaluates in O(1) time
- Cons: Low amount of moves **BUT** not the shortest path (compared with iterative deepening)
- Heuristic would be better with a second Pattern database using a 3x3 kernel
- Only use 5x3 kernel when no solutions are found within the 3x3 window
- ~9! / 2 or 181 440 permutations → brute force feasible
- Right now the algorithm solves the pair (1, 1) by preserving the (2,2) pairs
- Sometimes this isn't necessary because breaking solved matching pairs can generate a new pairs

g	y	b	b	b
w	w	e	r	b
g	y	b	b	r

As shown on the left, It is possibly quicker to solve for the (r, r) pair **and** the green (b, b) pair rather than preserving both blue (b, b) pairs

Search Algorithm - Best-first search

- Attempted: Best-first search & Breadth-first search
  - Both algorithms gave very similar results but breadth-first search was considerably slower (~ 4-5 orders of magnitude slower) and not worth it.
- Best-first search chosen
  - Finds all unsolved pairs in a board
  - Transform board T(x) = x’ for every pair
  - The hash of the board H(x’) is the key in the pattern database and its value is the sequence of movements to perform
  - Picks the pair which will take the less amount of moves to solve
  - Worst case scenario: all 5 columns need be solved individually and therefore 5 searches.
- Boards are always solved within 1 ms (lab computer)
- ***NOTE: the 3x3 kernel + 5x3 kernel solution proposed in the previous slide would most likely make this algorithm close to A* instead.*

Challenge

- Time: All boards were solved in less than 0 ms
- Path: The shortest paths were not found although they were quite reasonably low
- Issues encountered: Some libraries were not supported by the lab machines → had to comment out code
- Overall the challenge was a success and we believe we have the quickest heuristic evaluation possible (O(1)) despite the time requirements not being a huge factor in the competition.