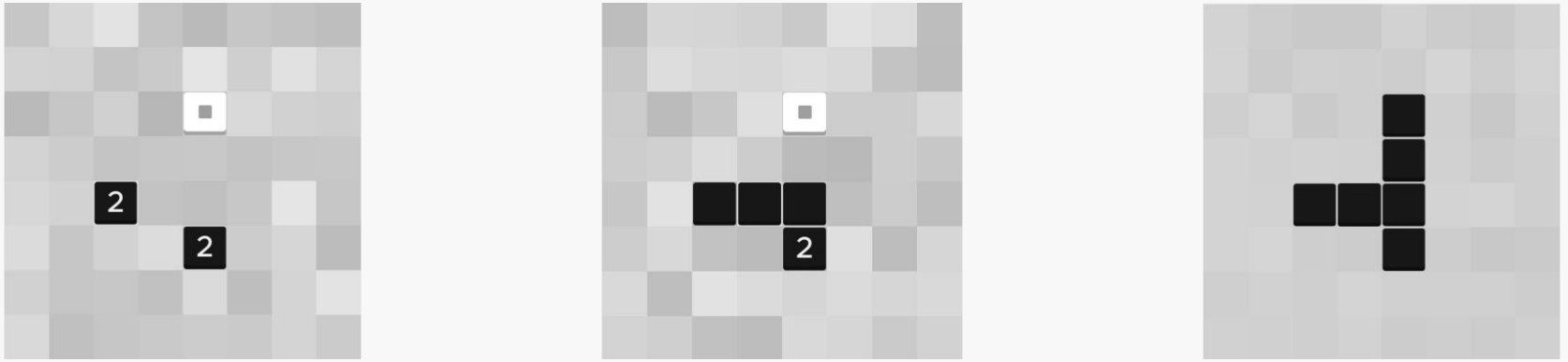
IART – CHECKPOINT 1

Daniel Brandão – 201705812
Pedro Moás – 201705208
Gaspar Pinheiro - 201704700

ZHED - Single Player Game



- ZHED is a grid based puzzle game where in order to complete each level a numbered cell must be expanded to reach the goal cell.
- Each numbered cell can be expanded in one of four directions and overlapped. Each cell expands *n* cells in the direction chosen, decreasing by one for each empty cell. When a expanding cell overlaps an already filled cell, the number of cells to be filled in the direction of the expansion is not decreased.

Problem Formulation

For each level, the puzzle size and the numbered cells' positions are diferent. Therefore we are using as an example for the formulation a puzzle size of 4.

- State Representation: NxN Matrix (List of Lists of integers, N = Puzzle Size), where each cell can have a value, *Val*, of:
  - A positive number, representing the expandable length of the cell;
  - 0, representing an empty cell;
  - -1, representing an expanded cell;
  - -2, representing the goal cell.
  - -3, representing the reached goal cell.
- Initial State: Matrix with at least one Val = -2 , at least one Val > 0, and no Val = –1 or Val = –3 cells
- Final State: Matrix with one Val = -3

1	0	-2	0
0	0	0	0
0	0	0	3
0	2	0	0

Initial State example

-1	-1	-3	0
0	-1	0	0
-1	-1	-1	-1
0	-1	0	0

Final State example

Operators

Operator	Pre-conditions	Effects	Cost

Move [X,Y] Up	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X, Y – i) != 0) continue;
		Board(X, Y – i) = -1; i++;
			-30

Move [X,Y] Down	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X, Y + i) != 0) continue;
		Board(X, Y + i) = -1; i++;
			-30

Move [X,Y] Left	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X - i, Y) != 0) continue;
		Board(X - i, Y) = -1; i++;
			-30

Move [X,Y] Right	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X + i, Y) != 0) continue;
		Board(X + i, Y) = -1; i++;
			-30

Approach

- For the heuristic methods (greedy, A*), we used different heuristics, such as:
  - H1 = Minimum *Zhed Distance* between a Value Cell and a Finish Tile.
  - *H2 = (1 - Number of Reached Finish Tiles) / (Number of tiles aligned with a Finish Tile)*
  - *H3 = (1 - Number of Reached Finish Tiles) * 1000 /*BoardMaxValue
  - H4 = *(1 - Number of Reached Finish Tiles) * 1000 /*BoardTotalValue
  - *H5 = H2 + H4*

The Zhed Distance between a Value Cell and a Tile, **only applicable when they are in the same row or column**, consists of:
(The actual distance between them) -
(The number of used tiles between the) -
(The Cell's Value)

*The **Tile Extension Value** is the number of tiles a Value Tile can extend in a certain direction (extending through a Value Tile only adds half the value as other Tiles)*

***BoardMaxValue:** Sum of the Maximum Tile Extension Value of each Value Tile in the Board and the number of expanded cells*

***BoardTotalValue:** Sum of the Average of all 4 Tile Extension Values of each Value Tile in the Board and the number of expanded cells*

Implemented algorithms

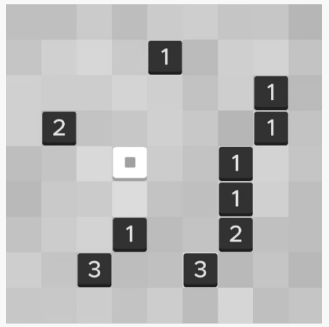
- We implemented DFS, BFS, Greedy, A* and Uniform Cost Algorithms.
- For DFS and BFS, no heuristic and cost functions were used.
- For Greedy and A*, several heuristics were tried, as explained in the previous slide.
- For Uniform Cost and A*, only one cost function was used, as it experimentally did not significantly impact the obtained results.

Development

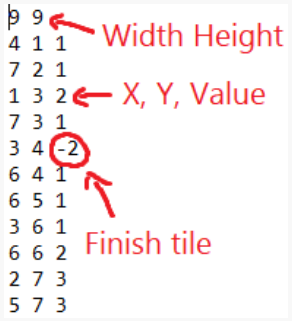
- We used Unity and C# as a programming language .

For Zhed Level 41, its file structure, console and graphical representation are shown below:

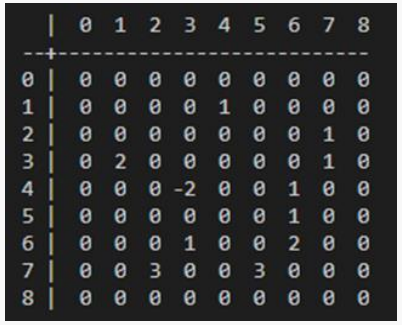
Zhed App



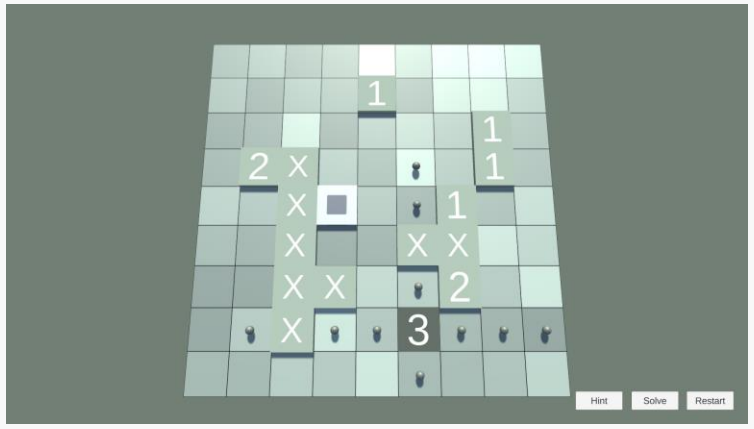
File representation



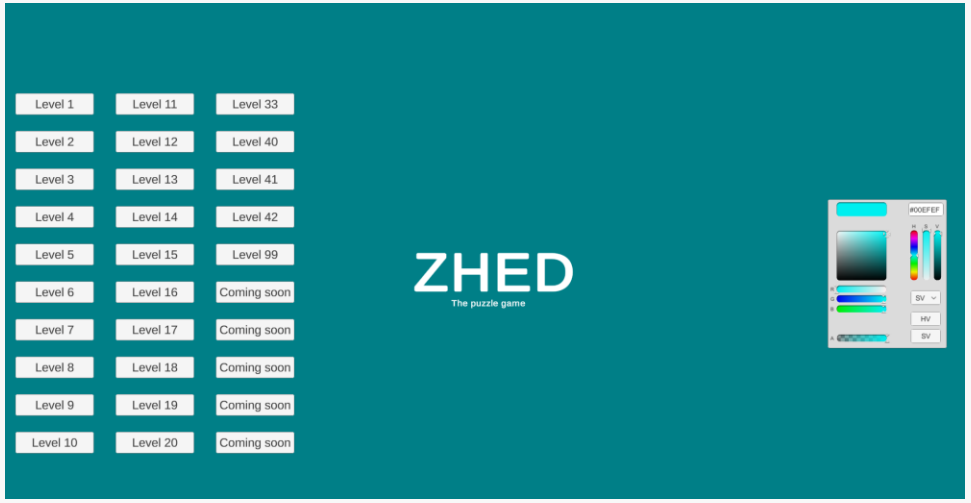
Console



Graphical

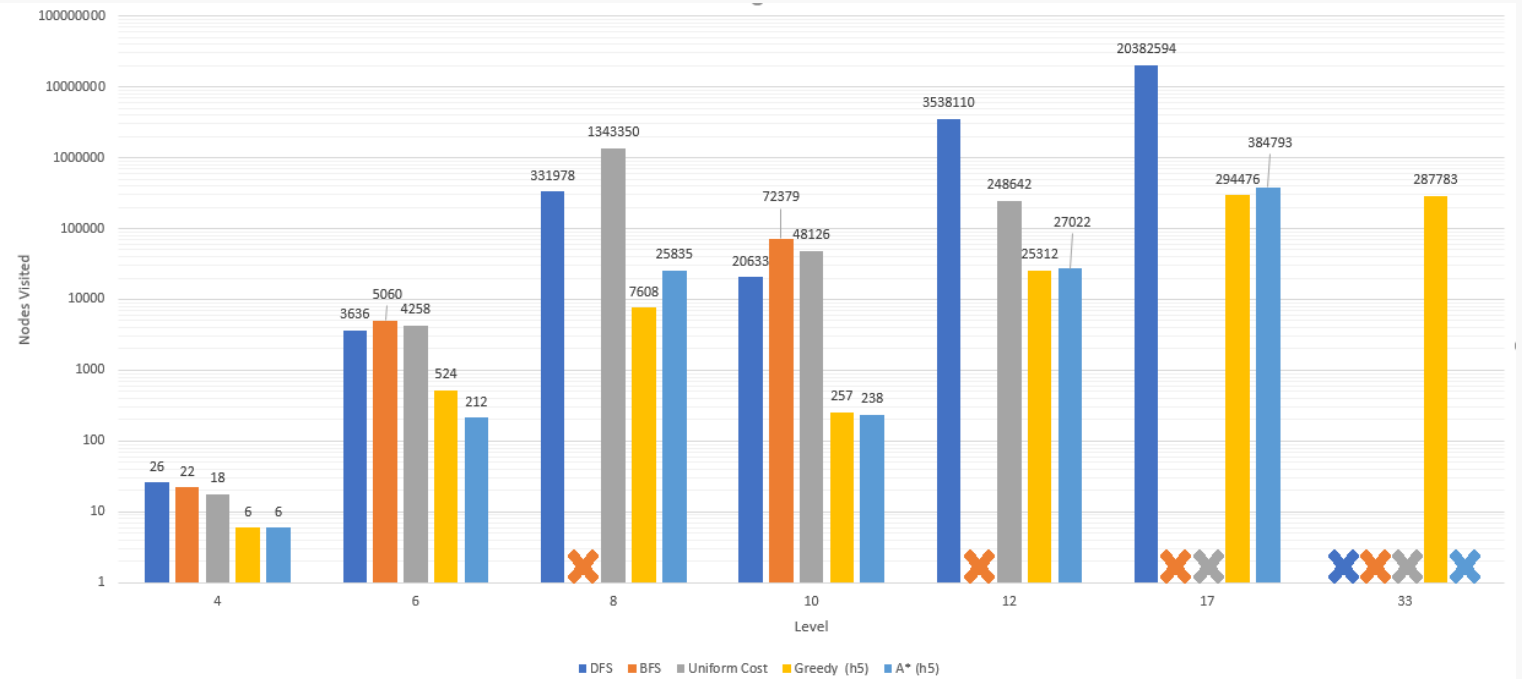


Zhed - Menu



Results

Nodes Visited in different levels for each algorithm



Nodes Visited in different levels by diferent heuristics with greedy algorithm

Level	h1	h2	h3	h4	h5 (h2 + h4)
6	1268	463	246	5083	524
8	157997	61788	10532	77683	7608
10	18231	2960	864	1298	257
12	163121	45540	333449	645831	25312

Solve time(ms) for diferent levels by diferent heuristics with greedy algorithm

Level	h1	h2	h3	h4	h5 (h2 + h4)
6	39	18	11	27	18
8	4658	2436	522	2496	461
10	492	115	37	42	12
12	5123	2042	10855	3014	827

Conclusions

- Overall, we have successfully implemented search algorithms on a Zhed Solver, not only uninformed ones such as DFS and BFS, which are simpler to implement but end up taking 5-10 times as long, and also informed search algorithms, which can have better heuristics for different levels, as the solutions for each level can be very different.
- For instance, heuristic 4 performs a lot better than heuristic 2 (more than 2x) on Level 10, but worse on Level 12 (~1.5x).

Related Work

- https://www.wilgysef.com/articles/zhed-solver/
- https://github.com/WiLGYSeF/zhed-solver
- https://www.cin.ufpe.br/~if684/EC/aulas-IASimbolica/korf96-search.pdf
- http://archive.oreilly.com/oreillyschool/courses/data-structures-algorithms/singlePlayer.html
- http://www.pvv.ntnu.no/~spaans/spec-cs.pdf