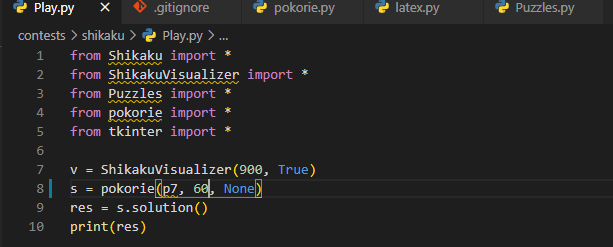
Artificial Intelligence

Shikaku

The shikaku project was challenging as we will soon see the different iterations my code went through to its current optimal form. The goal of the project was to take the basic solution provided by the professor, Dr. David Fletcher, and significantly improve it to solve puzzles of increasing complexity and scope.

I had 7 different test puzzles with each puzzle being more complex than the previous one to test my code. Some guiding principles that I adhered to for an optimal solution included the following:

1. Find a solution to each puzzle
2. Do so as fast as you can
3. Use methods like pruning, inference and backtracking to optimize solution



To achieve this, I wrote a little test script in *python* to aid me to test my solution as I developed it. The script is shown above. The script imports the necessary libraries including my module, *pokorie*, which houses the logic to solve a Shikaku puzzle.

*ShikakuVisualizer* is a library that allows us to visualize in 2D a Shikaku board and watch as the algorithm fills in the different regions with their appropriate assigned colors. The algorithm has 60 seconds to find a solution.

Now, onto the several iterations my code went through to its optimal state now.

**Trial 1: If region is a prime number, only include rectangles that contain the origin’s row or column**

From the onset I knew I wanted to prune the options for each region. As many false or invalid options that I could prune away, the better for the algorithm. This was because less options per region equated to less node to explore looking for a solution; thereby speeding up the search.

With that in mind, if a region was a prime number, I only included options on the horizontal or vertical axis of where the region was located. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |

With this first iteration I only managed to solve the first two puzzles under 60 seconds.

**Trial 2: Only include rectangles that do not contain other regions**

Next was to include, for a particular region, rectangles that do not form the coordinates of other regions. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |

Still unable to solve puzzles P3 – P7. However, the overall search time decreased.

**Trial 3: If region is not a prime number, only include rectangles that can form a possible region (using Manhattan distance)**

Next was to include, for a particular non-prime region, rectangles whose Manhattan distance were less than or equal to the size of the region. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |

Still unable to solve puzzles P3 – P7. However, the number of nodes explored, and the overall search time decreased significantly for puzzles P1 and P2.

**Trial 4: Sort the regions to fill in higher regions first**

The algorithm is set up in such a way that all the valid rectangles of each region are precalculated first, then the search for the solution begins for each region. A careful look at the code revealed that the order in which the regions were sorted could affect how fast a solution was attained. As such, I sorted the regions based on their sizes from highest to lowest. This meant that the region with the highest size was explored first. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |

We have some improvements. I was able to solve puzzles P3 and P4 in addition to P1 and P2. Also, notice that the number of nodes explored for P2 decreased significantly. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |

**Trial 5 – Fill in the highest region first, then those with less rectangles next**

Still on the logic of sorting. Next, I tried sorting by regions with less rectangles (options) after sorting by theones with higher sizes. The idea was that if two distinct regions had the same size, then the one with fewer options will be considered first by the algorithm. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |

Still unable to solve puzzles P5 – P7. However, the search time for P4 improved as less nodes were explored. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |

**Trial 6: Infer by removing filled-in options from other regions**

Next was to try to discard rectangles from other regions that were no longer valid, after a region, *x*, was filled. The idea was that this would in turn mean less nodes were considered by the algorithm when trying to solve a region, improving speed.

The logic in this trial was simple. Once a region, *x*, was filled by one of its potential rectangles, *r*, I looped through all the other regions and removed any rectangle that equaled *r.* Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |
| 6 | 22 | 51 | 188 | 1756 | - | - | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |
| 6 | 13 | 37 | 162 | 1718 | - | - | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |
| 6 | 0.00027 | 0.00039 | 0.00081 | 0.00095 | - | - | - |

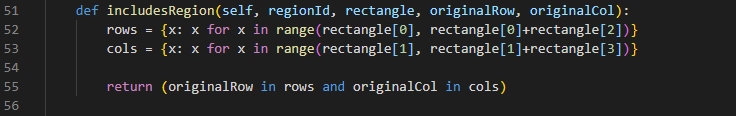
**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |
| 6 | 0.0060 | 0.020 | 0.16 | 1.66 | - | - | - |

Still unable to solve puzzles P5 – P7. However, the search times across P1 – P4 were improved as less time was spent at each node. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |
| 6 | 6 | 70.26 | 62.181 | pokorie | 0.019 | 0.251 | 2.441 | 128.194 | 180.000 |

**Trial 7: Check if rectangle includes region before adding as possible option**

Still unable to solve problems P5 – P7 under 60 seconds, I continued to look for ways to prune the rectangles for each region. I noticed that I could reuse the logic that checked if an option (rectangle) for a region was consistent. That is, if it had not yet been filled or filled with the right region. After a slight modification, I came up with the following logic that set a valid option for a region to be the one that contained the coordinates of the region when filled. 

With the above logic in place during the generation of possible rectangles per region, my test results came up as follows.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |
| 6 | 22 | 51 | 188 | 1756 | - | - | - |
| 7 | 18 | 47 | 94 | 239 | 5784 | 23491 | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |
| 6 | 13 | 37 | 162 | 1718 | - | - | - |
| 7 | 9 | 33 | 68 | 201 | 5727 | 23407 | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |
| 6 | 0.00027 | 0.00039 | 0.00081 | 0.00095 | - | - | - |
| 7 | 0.00039 | 0.00049 | 0.00093 | 0.0015 | 0.0012 | 0.0021 | - |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |
| 6 | 0.0060 | 0.020 | 0.16 | 1.66 | - | - | - |
| 7 | 0.0070 | 0.023 | 0.088 | 0.37 | 6.86 | 49.98 | - |

Now I can solve P1 – P6. Also notice that fewer backtracks were made due to the more precise pruning. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |
| 6 | 6 | 70.26 | 62.181 | pokorie | 0.019 | 0.251 | 2.441 | 128.194 | 180.000 |
| 7 | 6 | 74.43 | 38.224 | pokorie | 0.016 | 0.145 | 0.671 | 10.289 | 180.000 |

**Trial 8: Optimize inference logic**

We are edging closer. I took another look at my previous inference logic and looked for ways to improve it. I noticed that removing filled-in rectangles from other regions was not enough. I went a step further and removed rectangles from other regions that were no longer consistent due to the just filled-in region. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |
| 6 | 22 | 51 | 188 | 1756 | - | - | - |
| 7 | 18 | 47 | 94 | 239 | 5784 | 23491 | - |
| 8 | 18 | 37 | 93 | 216 | 5238 | 19161 | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |
| 6 | 13 | 37 | 162 | 1718 | - | - | - |
| 7 | 9 | 33 | 68 | 201 | 5727 | 23407 | - |
| 8 | 9 | 26 | 68 | 182 | 5183 | 19080 | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |
| 6 | 0.00027 | 0.00039 | 0.00081 | 0.00095 | - | - | - |
| 7 | 0.00039 | 0.00049 | 0.00093 | 0.0015 | 0.0012 | 0.0021 | - |
| 8 | 0.00033 | 0.00040 | 0.00060 | 0.00096 | 0.00057 | 0.00090 | - |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |
| 6 | 0.0060 | 0.020 | 0.16 | 1.66 | - | - | - |
| 7 | 0.0070 | 0.023 | 0.088 | 0.37 | 6.86 | 49.98 | - |
| 8 | 0.0060 | 0.015 | 0.056 | 0.21 | 2.99 | 17.34 | - |

Significant improvement! Across P1 – P7, the numbers decreased. Fewer backtracking, less time spent at each node and less search time. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |
| 6 | 6 | 70.26 | 62.181 | pokorie | 0.019 | 0.251 | 2.441 | 128.194 | 180.000 |
| 7 | 6 | 74.43 | 38.224 | pokorie | 0.016 | 0.145 | 0.671 | 10.289 | 180.000 |
| 8 | 5 | 76.02 | 37.039 | pokorie | 0.010 | 0.089 | 0.320 | 4.775 | 180.000 |

**Trial 9: Infer by filling regions with only an option recursively**

Still on improving the infer logic. After pruning away invalid rectangles, I filled regions with only an option left and recursively repeated the inference process until no more regions could be filled. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |
| 6 | 22 | 51 | 188 | 1756 | - | - | - |
| 7 | 18 | 47 | 94 | 239 | 5784 | 23491 | - |
| 8 | 18 | 37 | 93 | 216 | 5238 | 19161 | - |
| 9 | 1 | 1 | 15 | 8 | 22 | 444 | - |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |
| 6 | 13 | 37 | 162 | 1718 | - | - | - |
| 7 | 9 | 33 | 68 | 201 | 5727 | 23407 | - |
| 8 | 9 | 26 | 68 | 182 | 5183 | 19080 | - |
| 9 | 0 | 0 | 10 | 4 | 17 | 432 | - |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |
| 6 | 0.00027 | 0.00039 | 0.00081 | 0.00095 | - | - | - |
| 7 | 0.00039 | 0.00049 | 0.00093 | 0.0015 | 0.0012 | 0.0021 | - |
| 8 | 0.00033 | 0.00040 | 0.00060 | 0.00096 | 0.00057 | 0.00090 | - |
| 9 | 0.0050 | 0.011 | 0.0027 | 0.014 | 0.017 | 0.0044 | - |

**Search time**

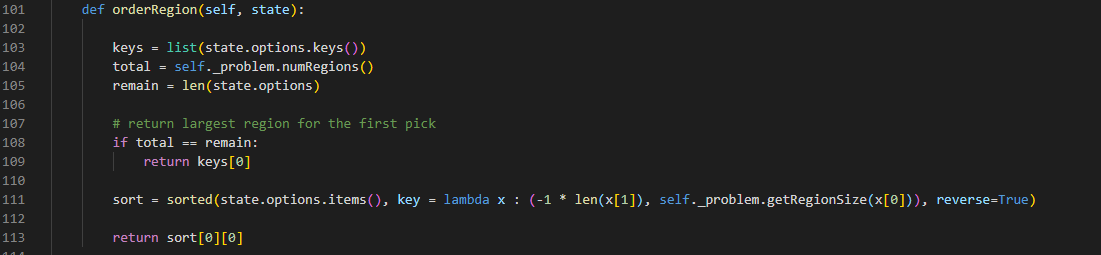
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |
| 6 | 0.0060 | 0.020 | 0.16 | 1.66 | - | - | - |
| 7 | 0.0070 | 0.023 | 0.088 | 0.37 | 6.86 | 49.98 | - |
| 8 | 0.0060 | 0.015 | 0.056 | 0.21 | 2.99 | 17.34 | - |
| 9 | 0.0050 | 0.011 | 0.040 | 0.12 | 0.37 | 1.95 | - |

Another significant jump in speed as more regions are filled in due to the inference. However, P7 was still proving a hard nut to crack. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |
| 6 | 6 | 70.26 | 62.181 | pokorie | 0.019 | 0.251 | 2.441 | 128.194 | 180.000 |
| 7 | 6 | 74.43 | 38.224 | pokorie | 0.016 | 0.145 | 0.671 | 10.289 | 180.000 |
| 8 | 5 | 76.02 | 37.039 | pokorie | 0.010 | 0.089 | 0.320 | 4.775 | 180.000 |
| 9 | 6 | 85.02 | 36.103 | pokorie | 0.006 | 0.049 | 0.110 | 0.350 | 180.000 |

**Trial 10: Adding some ordering before selecting regions to fill**

Without any foreseeable way to improve the inference logic, I looked at other parts of the algorithm that may need tweaking. After visualizing how the algorithm went about solving the problems, I noticed that the region it picked next to fill played an important role as a heuristic. Also, keep in mind that due to inference, it was possible that remaining regions had less options and as such I could sort by that. As such, I added a simple sorting algorithm as follows:



The region with the biggest size was picked first. Subsequently, the remaining regions were sorted by the number of rectangles left in descending order. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |
| 6 | 22 | 51 | 188 | 1756 | - | - | - |
| 7 | 18 | 47 | 94 | 239 | 5784 | 23491 | - |
| 8 | 18 | 37 | 93 | 216 | 5238 | 19161 | - |
| 9 | 1 | 1 | 15 | 8 | 22 | 444 | - |
| 10 | 1 | 1 | 5 | 9 | 10 | 58 | 303 |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |
| 6 | 13 | 37 | 162 | 1718 | - | - | - |
| 7 | 9 | 33 | 68 | 201 | 5727 | 23407 | - |
| 8 | 9 | 26 | 68 | 182 | 5183 | 19080 | - |
| 9 | 0 | 0 | 10 | 4 | 17 | 432 | - |
| 10 | 0 | 0 | 2 | 4 | 5 | 45 | 271 |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |
| 6 | 0.00027 | 0.00039 | 0.00081 | 0.00095 | - | - | - |
| 7 | 0.00039 | 0.00049 | 0.00093 | 0.0015 | 0.0012 | 0.0021 | - |
| 8 | 0.00033 | 0.00040 | 0.00060 | 0.00096 | 0.00057 | 0.00090 | - |
| 9 | 0.0050 | 0.011 | 0.0027 | 0.014 | 0.017 | 0.0044 | - |
| 10 | 0.0030 | 0.0080 | 0.0056 | 0.011 | 0.029 | 0.013 | 0.0091 |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |
| 6 | 0.0060 | 0.020 | 0.16 | 1.66 | - | - | - |
| 7 | 0.0070 | 0.023 | 0.088 | 0.37 | 6.86 | 49.98 | - |
| 8 | 0.0060 | 0.015 | 0.056 | 0.21 | 2.99 | 17.34 | - |
| 9 | 0.0050 | 0.011 | 0.040 | 0.12 | 0.37 | 1.95 | - |
| 10 | 0.0030 | 0.0080 | 0.028 | 0.096 | 0.29 | 0.77 | 2.76 |

Finally, P7 solved! Oh, the magic of sorting! The search time improved across the board as well. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |
| 6 | 6 | 70.26 | 62.181 | pokorie | 0.019 | 0.251 | 2.441 | 128.194 | 180.000 |
| 7 | 6 | 74.43 | 38.224 | pokorie | 0.016 | 0.145 | 0.671 | 10.289 | 180.000 |
| 8 | 5 | 76.02 | 37.039 | pokorie | 0.010 | 0.089 | 0.320 | 4.775 | 180.000 |
| 9 | 6 | 85.02 | 36.103 | pokorie | 0.006 | 0.049 | 0.110 | 0.350 | 180.000 |
| 10 | 4 | 101.38 | 0.537 | pokorie | 0.006 | 0.027 | 0.083 | 0.226 | 2.343 |

**Trial 11: Code clean ups**

Next, I took a fresh look at my code and cleaned up some unnecessary lines of codes. I rewrote some logic to make use of better data structures like using dictionaries instead of lists for faster searching. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |
| 6 | 22 | 51 | 188 | 1756 | - | - | - |
| 7 | 18 | 47 | 94 | 239 | 5784 | 23491 | - |
| 8 | 18 | 37 | 93 | 216 | 5238 | 19161 | - |
| 9 | 1 | 1 | 15 | 8 | 22 | 444 | - |
| 10 | 1 | 1 | 5 | 9 | 10 | 58 | 303 |
| 11 | 3 | 1 | 5 | 9 | 10 | 58 | 303 |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |
| 6 | 13 | 37 | 162 | 1718 | - | - | - |
| 7 | 9 | 33 | 68 | 201 | 5727 | 23407 | - |
| 8 | 9 | 26 | 68 | 182 | 5183 | 19080 | - |
| 9 | 0 | 0 | 10 | 4 | 17 | 432 | - |
| 10 | 0 | 0 | 2 | 4 | 5 | 45 | 271 |
| 11 | 1 | 0 | 2 | 4 | 5 | 45 | 271 |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |
| 6 | 0.00027 | 0.00039 | 0.00081 | 0.00095 | - | - | - |
| 7 | 0.00039 | 0.00049 | 0.00093 | 0.0015 | 0.0012 | 0.0021 | - |
| 8 | 0.00033 | 0.00040 | 0.00060 | 0.00096 | 0.00057 | 0.00090 | - |
| 9 | 0.0050 | 0.011 | 0.0027 | 0.014 | 0.017 | 0.0044 | - |
| 10 | 0.0030 | 0.0080 | 0.0056 | 0.011 | 0.029 | 0.013 | 0.0091 |
| 11 | 0.0010 | 0.0040 | 0.0030 | 0.0049 | 0.012 | 0.0054 | 0.0051 |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |
| 6 | 0.0060 | 0.020 | 0.16 | 1.66 | - | - | - |
| 7 | 0.0070 | 0.023 | 0.088 | 0.37 | 6.86 | 49.98 | - |
| 8 | 0.0060 | 0.015 | 0.056 | 0.21 | 2.99 | 17.34 | - |
| 9 | 0.0050 | 0.011 | 0.040 | 0.12 | 0.37 | 1.95 | - |
| 10 | 0.0030 | 0.0080 | 0.028 | 0.096 | 0.29 | 0.77 | 2.76 |
| 11 | 0.0030 | 0.0040 | 0.015 | 0.044 | 0.12 | 0.31 | 1.55 |

The search time decreased on P2 – P7 by half. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |
| 6 | 6 | 70.26 | 62.181 | pokorie | 0.019 | 0.251 | 2.441 | 128.194 | 180.000 |
| 7 | 6 | 74.43 | 38.224 | pokorie | 0.016 | 0.145 | 0.671 | 10.289 | 180.000 |
| 8 | 5 | 76.02 | 37.039 | pokorie | 0.010 | 0.089 | 0.320 | 4.775 | 180.000 |
| 9 | 6 | 85.02 | 36.103 | pokorie | 0.006 | 0.049 | 0.110 | 0.350 | 180.000 |
| 10 | 4 | 101.38 | 0.537 | pokorie | 0.006 | 0.027 | 0.083 | 0.226 | 2.343 |
| 11 | 2 | 104.49 | 0.359 | pokorie | 0.004 | 0.017 | 0.047 | 0.141 | 1.586 |

**Step 12: Add Manhattan distance for each option and sort by smallest**

Could more be done to improve the code? Well, I gave hard thought about it and saw an area that could be improved. The logic behind the algorithm dictates that the code could explore all valid rectangles (nodes) of a region until it is filled, and the entire puzzle is solved. Could we then sort, for each region, the valid options so that the ones more likely to lead to a solution faster are explored first?

Well, I explored using the Manhattan distance between an option and the region’s original coordinate (rectangle) to sort the rectangles (options) of a region. Below are my test results.

**Number of nodes expanded**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 77 | - | - | - | - | - | - |
| 1 | 29 | 14746 | - | - | - | - | - |
| 2 | 29 | 14746 | - | - | - | - | - |
| 3 | 9 | 627 | - | - | - | - | - |
| 4 | 14 | 49 | 125 | 4935 | - | - | - |
| 5 | 22 | 51 | 188 | 1756 | - | - | - |
| 6 | 22 | 51 | 188 | 1756 | - | - | - |
| 7 | 18 | 47 | 94 | 239 | 5784 | 23491 | - |
| 8 | 18 | 37 | 93 | 216 | 5238 | 19161 | - |
| 9 | 1 | 1 | 15 | 8 | 22 | 444 | - |
| 10 | 1 | 1 | 5 | 9 | 10 | 58 | 303 |
| 11 | 3 | 1 | 5 | 9 | 10 | 58 | 303 |
| 12 | 3 | 1 | 6 | 6 | 7 | 60 | 171 |

**Number of backtracks**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 68 | - | - | - | - | - | - |
| 1 | 20 | 14732 | - | - | - | - | - |
| 2 | 20 | 14732 | - | - | - | - | - |
| 3 | 0 | 613 | - | - | - | - | - |
| 4 | 5 | 35 | 99 | 4897 | - | - | - |
| 5 | 13 | 37 | 162 | 1718 | - | - | - |
| 6 | 13 | 37 | 162 | 1718 | - | - | - |
| 7 | 9 | 33 | 68 | 201 | 5727 | 23407 | - |
| 8 | 9 | 26 | 68 | 182 | 5183 | 19080 | - |
| 9 | 0 | 0 | 10 | 4 | 17 | 432 | - |
| 10 | 0 | 0 | 2 | 4 | 5 | 45 | 271 |
| 11 | 1 | 0 | 2 | 4 | 5 | 45 | 271 |
| 12 | 1 | 0 | 1 | 2 | 2 | 47 | 145 |

**Search time per node**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.00066 | - | - | - | - | - | - |
| 1 | 0.00048 | 0.0012 | - | - | - | - | - |
| 2 | 0.00041 | 0.0010 | - | - | - | - | - |
| 3 | 0.00055 | 0.00075 | - | - | - | - | - |
| 4 | 0.00036 | 0.00047 | 0.0013 | 0.0015 | - | - | - |
| 5 | 0.00041 | 0.00053 | 0.0012 | 0.0015 | - | - | - |
| 6 | 0.00027 | 0.00039 | 0.00081 | 0.00095 | - | - | - |
| 7 | 0.00039 | 0.00049 | 0.00093 | 0.0015 | 0.0012 | 0.0021 | - |
| 8 | 0.00033 | 0.00040 | 0.00060 | 0.00096 | 0.00057 | 0.00090 | - |
| 9 | 0.0050 | 0.011 | 0.0027 | 0.014 | 0.017 | 0.0044 | - |
| 10 | 0.0030 | 0.0080 | 0.0056 | 0.011 | 0.029 | 0.013 | 0.0091 |
| 11 | 0.0010 | 0.0040 | 0.0030 | 0.0049 | 0.012 | 0.0054 | 0.0051 |
| 12 | 0.0010 | 0.0040 | 0.0022 | 0.0068 | 0.014 | 0.0048 | 0.0036 |

**Search time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** |
| 0 | 0.051 | - | - | - | - | - | - |
| 1 | 0.014 | 17.83 | - | - | - | - | - |
| 2 | 0.012 | 15.42 | - | - | - | - | - |
| 3 | 0.0050 | 0.47 | - | - | - | - | - |
| 4 | 0.0050 | 0.023 | 0.17 | 7.32 | - | - | - |
| 5 | 0.0090 | 0.027 | 0.22 | 2.59 | - | - | - |
| 6 | 0.0060 | 0.020 | 0.16 | 1.66 | - | - | - |
| 7 | 0.0070 | 0.023 | 0.088 | 0.37 | 6.86 | 49.98 | - |
| 8 | 0.0060 | 0.015 | 0.056 | 0.21 | 2.99 | 17.34 | - |
| 9 | 0.0050 | 0.011 | 0.040 | 0.12 | 0.37 | 1.95 | - |
| 10 | 0.0030 | 0.0080 | 0.028 | 0.096 | 0.29 | 0.77 | 2.76 |
| 11 | 0.0030 | 0.0040 | 0.015 | 0.044 | 0.12 | 0.31 | 1.55 |
| 12 | 0.0030 | 0.0040 | 0.022 | 0.041 | 0.10 | 0.29 | 0.61 |

And it worked. P7’s search time decreased. After this trial, I uploaded my code to the contest and my results were as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trial** | **Rank** | **Score** | **Average Time** | **Agent** | **P1** | **P2** | **P3** | **P4** | **P5** |
| 4 | 6 | 69.12 | 74.174 | pokorie | 0.009 | 0.247 | 10.614 | 180.000 | 180.000 |
| 5 | 6 | 69.41 | 72.807 | pokorie | 0.012 | 0.348 | 3.674 | 180.000 | 180.000 |
| 6 | 6 | 70.26 | 62.181 | pokorie | 0.019 | 0.251 | 2.441 | 128.194 | 180.000 |
| 7 | 6 | 74.43 | 38.224 | pokorie | 0.016 | 0.145 | 0.671 | 10.289 | 180.000 |
| 8 | 5 | 76.02 | 37.039 | pokorie | 0.010 | 0.089 | 0.320 | 4.775 | 180.000 |
| 9 | 6 | 85.02 | 36.103 | pokorie | 0.006 | 0.049 | 0.110 | 0.350 | 180.000 |
| 10 | 4 | 101.38 | 0.537 | pokorie | 0.006 | 0.027 | 0.083 | 0.226 | 2.343 |
| 11 | 2 | 104.49 | 0.359 | pokorie | 0.004 | 0.017 | 0.047 | 0.141 | 1.586 |
| 12 | 2 | 106.36 | 0.187 | pokorie | 0.004 | 0.023 | 0.070 | 0.151 | 0.687 |

I tried using the Euclidean distance as a sorting tool, but that did not improve the results significantly.

**Conclusion**

Through sorting, inference, pruning and backtracking I have shown how I improved the basic shikaku solver to find a solution in as little time as possible. I averaged a search time of approximately 152 *ms* across the 7 puzzles.