Assignment 1

Alexey Iskhakov

17-th February 2017

Algorithms

There are 3 main algorithms used, as requested:

The first one is **Random Search** (or simply **Random**). It just rushes through the space searching for a planet and throws a bomb whenever it gets a chance (literally). It has nothing advanced to it, excluding only a habit not to fly through already explored cells. It has a death limit of 100 and takes the best out of every iteration.

The second one is **Backtracker**. **Random Backtracker**, actually. It functions as a typical backtracker – steps back from a danger when possible and continues its backtracking until some new unexplored cells are noticed nearby. The word "Random" in its name means that in case of exploring cells it just flies in random direction, avoiding already explored ones. And the bomb-launching is fairly simple: when no new unexplored sectors can be found, it looks for *Krakens* on it's explored map. And if any one of them has unknown sectors behind – the backtracker goes there, step by step, and launches a bomb right into the *Kraken.* This is not always necessarily unlocks the way out, of course, but increases chances to get to a *Planet* drastically.

And, finally, the third one – **A***. I decided to implement **A*** in the purest form possible. Therefore, it is very flexible and can be tuned to fit for different map types and to satisfy the need for speed or the shortest distance. Bomb throwing is rather simple too: when **A*** searches through the space for a goal with its "multiroutes-graph-thing" it just "throws" a bomb into the first *Kraken* that got in the way. It's rather efficient and it allows to find shorter routes than the ones without any obstacles. **A*** is still the fastest algorithm among others in many cases.

Part 1

Well… In this case it is fairly simple to conclude that the more advanced algorithm will be more efficient than the less advanced one. Therefore, **A*** almost always defeats **Backtracker** in both speed and path length, **Backtracker** defeats **Random Search** in both cases too, and **Random Search**… well… is just the worst algorithm out there. The only drawback for **A*** is that it cannot work on unexplored maps, uninformed. That means that there's a need to explore the map at first. The first version of **A*** in my solution did it by using BFS or/and **Backtracker**. But for the test purposes, to measure performance purely for the algorithm I decided to just feed the opened map right into it.

Disambiguation:

[ ] – the path of a ship.
. – Empty sector.
K – Kraken.
B – Black hole.
P – Planet.

There are some test maps 10 by 10 by 1 that are good for testing different things:

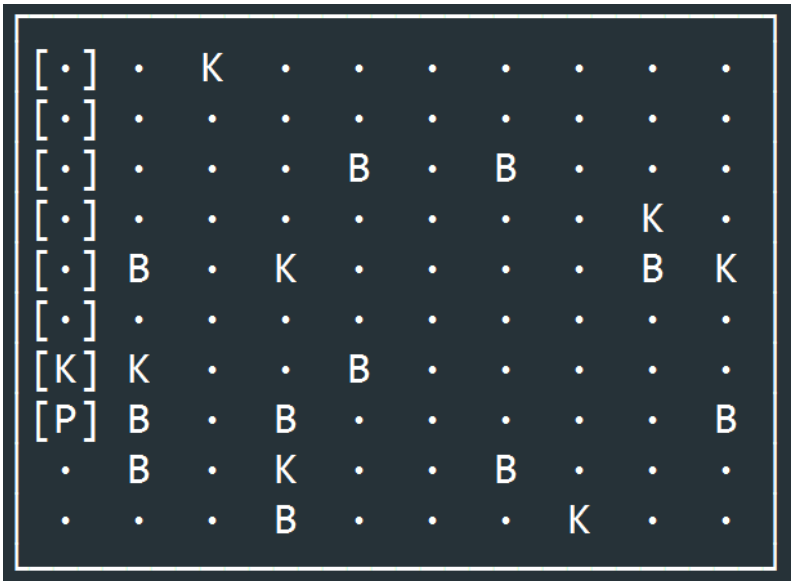
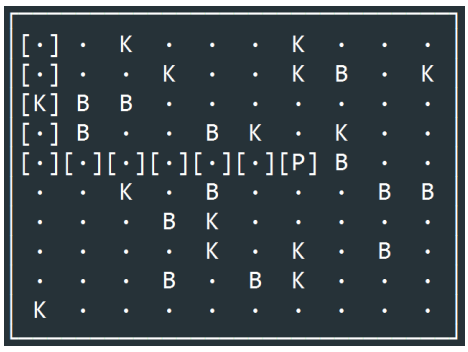


*Figure 1. This one was good to test **A*** bomb throwing (which was rather difficult to implement), as there is a Kraken that lies shortest path.*



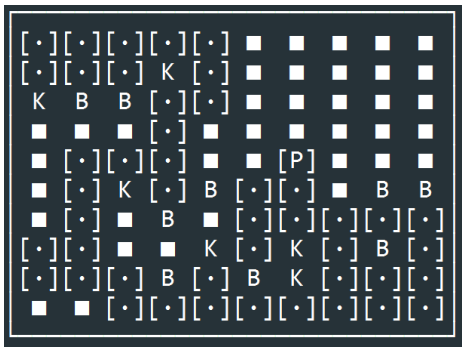


*Figure 2 and 3. These are good for testing all algorithms ability to throw bombs and get out of difficult situations. There is **A*** solution at the left and **Backtracker** solution at the right.*

There are also some maps to test the heuristic function of **A***, but there are too many cases to show.

Part 2

Megabomb is very useful when it comes to double Kraken walls that cannot be destroyed with the simple bomb. It makes many cases with multilayered walls of Krakens solvable. The drawback is obvious – wrongly thrown bomb can tear a planet apart easily and has a lot more chances to do it.

The significant increase in time savings would be in case of a double wall of krakens that separates start and planet points, but not taking the full height of the map. In this case, megabomb will allow to burst through this wall, which will result in both shorter path and less time wasted.

Part 3

The hardest arrangement ever possible, I think, is a big 3D-maze with tiny pathways and walls out of mostly black holes. **Random** and **Backtracker** will most likely not solve such a maze in short time or amount of deaths. The **A*****, in contrary, will feel almost no difficulties with such a map, with only consequence being a big work-time.

And with the impossible maps it's pretty simple – all that have no pathways, too many krakens to blow or that consist entirely of black holes and walls of them. And, of course, the most unsolvable case is the one without a planet…