Battlecode 2020 - Serpentine Noodles

On the 7th of January the Battlecode competition started. Battlecode is a programming contest organized by a student team from MIT, who come up with a different game each year. This year's edition was all about climate change and the rising water level. Our team consisted of: Manuel Ronhaar, Dik van Genuchten and Bram Grooten. You can find the codebase for our submission on GitHub.

Game summary

> First a short summary of the game. There are two teams, red and blue, on a 2D map of approximately 40x40 tiles. Each team begins with one robot, namely your headquarters (HQ). If this robot gets destroyed, the other team wins. A building, such as the HQ, can be destroyed by either flooding or a pile of dirt on top. Flooding happens over time, as the water level keeps rising exponentially (slowly during the first 1000 rounds, but quite fast after that).
> 'Soup' is the resource of the game, shared between a team. Teams need to mine and refine it, before it can be used. Other types of robots are landscapers, which are able to change the elevation of tiles, by picking up and dropping dirt. Drones can fly over water and carry other robots. Each robot is independently run by a copy of the same code. Communication betweens robots can be done over a 'blockchain'. Read the game specs for more details.
> {.is-info}

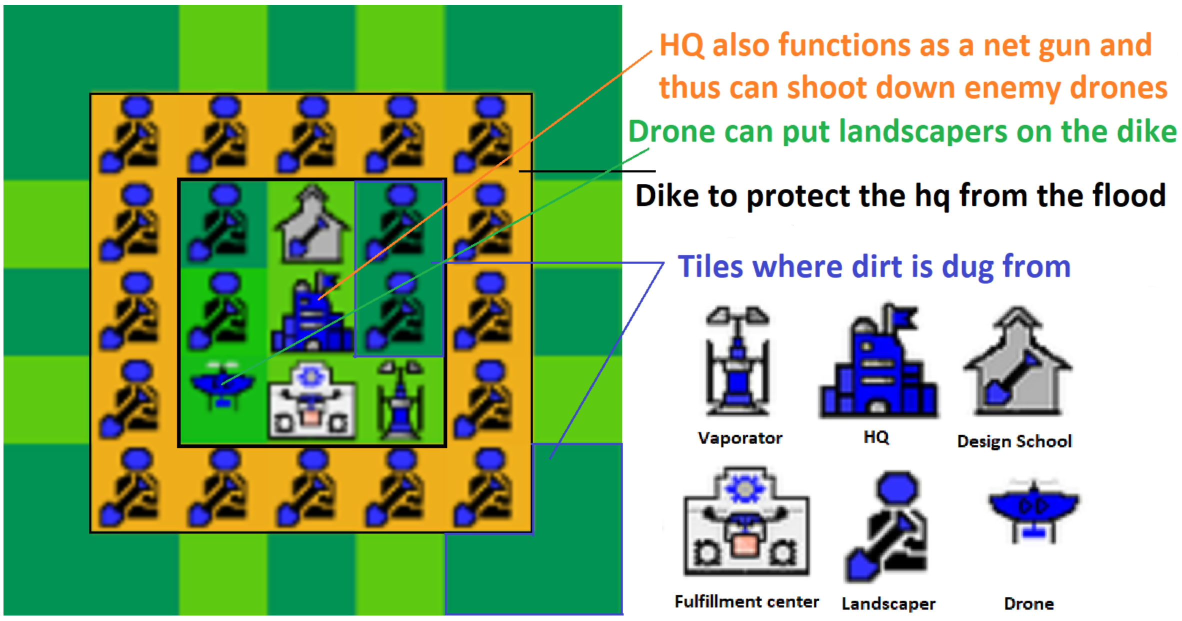
Initial strategy

After we read these game specs, our Dutch genes immediately said: let's build dikes! This tactic became standard among almost all the teams. Although some teams would almost solely focus on destroying the opponents HQ. After we got a good overview of the game, we started with this basic strategy:

1. Produce miners
2. Gather soup
3. Bring soup to HQ (which has a built-in refinery)
4. Build refineries (in neighborhood of soup fields)
5. Now bring soup to refineries instead of HQ
6. Build design school (which can produce landscapers)
7. Produce landscapers
8. Collect dirt
9. Build dike

10. Build fulfillment center (which can produce drones)
11. Produce drones

Thus, robots that weren't in our original strategy are vaporators and net guns. We saw their usefulness, but we wanted to get this basic plan off the ground. We had to determine where we wanted the dike. The standard option was of course on the 8 adjacent tiles to the HQ. But we wanted some more space to put some other buildings, so we came up with a wall of 16 blocks:


HQ also functions as a net gun and thus can shoot down enemy drones
Drone can put landscapers on the dike
Dike to protect the hq from the flood
Tiles where dirt is dug from
Vaporator
HQ
Design School
Fulfillment center
Landscaper
Drone

After uploading the first version of our bot, we played automatic matches against other teams. This was useful, because we noticed that there was another kind of strategy: [the rush-strategy]. The team in first place used this: bring a miner to the enemy HQ, let it build a design school, produce landscapers, and let the landscapers dig dirt onto the HQ. Within no time the HQ would be under 50 dirt, and therefore destroyed. We needed some strategy to defend against this. First we came up with drones that would take the enemy miner away but miners can walk faster than the drones so this is impractical or putting some of our own miners all around our HQ as temporary protection. However we would spend more resources defending than the enemy spent attacking. Then we thought of the logical strategy of putting landscapers near our HQ quickly, so they could take the dirt off and later start building the dike. We programmed the Design School to always produce two landscapers and up to four if the design school could sense an enemy design school or landscaper. This always worked as this only leaves three tiles empty next to the HQ so our landscapers will always be in the majority.

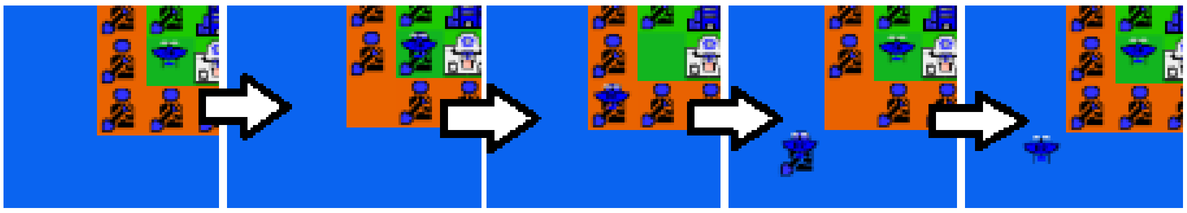
Strategy improvements

After evaluation of our initial strategy we discovered some points at which this strategy failed:

1. The HQ didn't have enough range to shoot enemy drones flying in the very corner of the base.
2. Opponents using a dike layout with more landscapers could survive the flood much longer.
3. In some of the later maps the HQ was located in a corner. As we hard coded the location of buildings relative to our HQ, this meant trouble.
4. Although we countered enemy landscapers destroying our HQ pretty soon we didn't fly them away from our base. This meant that this also countered the correct setup of our base.
5. Another strategy was to build a lot of vaporators, which meant the team could build a lot of drones and landscapers to overrun the enemy base just before the flood would destroy their own HQ.

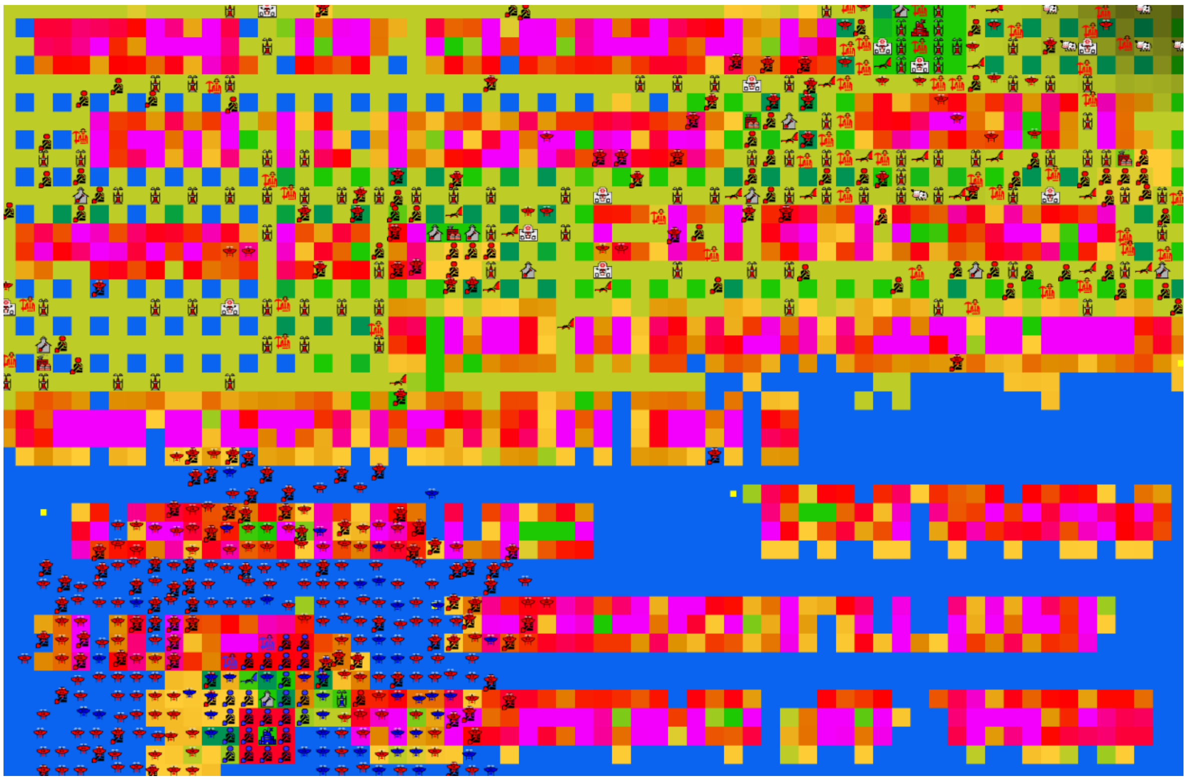
For problem one we saw two solutions: letting our own drones fly to the location where the enemy drones could otherwise grab our landscapers without being shot or not letting the landscapers walk to tiles where enemy drones could grab them if they sense an enemy drone. Although the second option slows down the dike building process a bit we went with this option as managing the drones so one goes to each of the problematic tiles was actually quite complicated.

The base used by teams with a more efficient dike set-up didn't allow for spawning extra landscapers if we grabbed the landscapers of their dike. Thus, we implemented drone attacks. The first step was enabling drones to escape the base if all tiles on the dike were already filled with landscapers. The second step was to let the drones fly to the enemy base and wait until there are enough drones to get a notable amount of landscapers off the wall.



Problem three and four were a lot more complicated. How could you make the base building adapt to all possible maps? We made the HQ compute a viable layout and communicate this layout via the blockchain so other robots could follow this plan. However not only the optimal base computation needed to be programmed, but the code of every robot needed to change so these robots could actually follow the adapting layout. In the end the HQ could at least rotate the base to fit the terrain. Along with this the landscapers that we spawned at the start tried to equalize the terrain adjacent to the HQ as much as possible allowing for our ideal layout to be built in more scenarios.

The tactic to build an insane amount of vaporators to acquire an even more insane amount of soup was harder to counter though. The only real counter we thought of was to attack the opponent after they had built some vaporators but had not yet profited from them. This would slow down our base building considerably and it was also questionable how good it would work as the opponent could stop building vaporators as soon as they noticed an attack. "If you can't beat them, join them" would be good advice if we had the time to implement this strategy, but we didn't.



Results

And that is how it ended, after two weeks of hard work we were ready to upload our final code. Although we still had ideas to improve our program we were also proud of what we had made and now we are proud to share the results with you! We ended around the 70th place in the rankings and had the honor of eliminating the other serpentine team in the international qualifying tournament. We are happy with these results, although we will of course try to end higher next year!