P5: Evolving Mario Levels

Jackie Sanchez, Elroy Saltzherr

University California Santa Cruz

2/13/2024

Explanation of Crossover:

The generate children function in Individual_DE is taking a random integer based on the length of the genome. It then crosses them over using the random integer to select where we are splicing the genome. We combine all of the genomes from the first parent up until our random number, with the other parent's genomes which are the respective other random number onward. We do this with both sets of genomes to create 2 children that are spliced oppositely, and then mutate them.

Explanation of Mutation:

The mutation function in the Individual_DE calculates a random number. From there if the number meets a criteria, it will pick a random piece of the world to change. Depending on the item that has been chosen, and another random number, the item will either be offset left, right, up or down, or changed to a different item altogether. After it has made the change, it will pop off the piece that it was originally, and push on the piece that it is changing to be onto the world (genome).

Additional Details:

What changed from the template and why, especially related to your selection strategies, fitness functions, crossover and mutation operators, etc.

In our Calculate Fitness, we added a few more metrics to influence the breeding done for each generation. We added in the decoration percentage to help reward levels that had more than a few items showing in the level. We also added a meaningful jumps parameter to also help encourage more things for Mario to have to jump to.

For our generate children function, we chose to do an elitist strategy to breed children that had higher fitness levels. In addition to this we chose to do a roulette wheel breeding to create another child. We felt that this would allow for the reward system in place for calculating fitness to produce levels that were varied, but still allow for some type of random breeding to take place.

Our mutate function then would roll some type of random number. From that outcome, there is a ⅔ chance something in the level will move

left or right. The other ⅓ chance will take an entire row that an item is found in and replace it with another random item. This allowed for varying levels throughout breeding to mitigate for all levels becoming the same elitist type of level.

Something about each of our favorite levels: why do we like them? How many generations did it take and how many seconds to generate these levels?

The reason we liked our favorite level was because it had a good amount of jumps that you had to do, so there weren't any spaces where you just had to run across an empty space. In addition, there was a good amount of challenge to the level itself with needing to use the sprint button in order to clear some jumps.

In each of the different sections of the level there is also enough “?” blocks or coins in order to give the player a reason to go there or even try and explore a slightly different path.

For any of the tougher jumps you are either able to try them again or you have a place to die, so there are no points where you are forced to reset due to getting into impossible to escape areas.

It also ends with a couple challenging jumps where if you miss it you die. So, the toughest area is at the very end, which indicates good challenge scaling.

In order to generate the level we had 10 generations and to get through all 10 generations it took about 2.5 seconds