CS 4341 Intro to AI

Assignment 2 Writeup

Cuong Nguyen

Gregory Port

Yahel Nachum

Duc Pham

Approach to computing:

Puzzle1 Fitness Function:

The fitness of a number sequence is determined by two variables, the sum of the sequence and if it is a valid sequence given the list of all possible numbers that can be used in a sequence.

Being a valid sequence means that the numbers in the sequence correspond to a unique number in the possible number sequence. For example if the possible number sequence is {1, 2, 3} a valid sequence would be {1, 2} or {1,2,3}, but an invalid sequence would be {1,1} or {3,2,3} since they repeat a possible number more times than it shows up in the possible numbers sequence.

If the sum of the sequence is less than or equal to the goal and it is a valid sequence then its fitness is equal to the sum of the sequence.

If the sum is greater than the goal and it is a valid sequence then its fitness score is (goal - sum of sequence). This was done to still be able to order the sequences that summed above the goal by saying that sequences that are above the goal but close to it are more fit than a sequence that is above the goal and farther away from it. For example, assume the goal is 10, the sequence {5,6} would still be seen as more fit than the sequence {8,9} but they would eventually still receive an official score of 0. This way we can order sequences that are closer to the goal than other ones even if their official scores would be 0.

Lastly whether or not a sequence sums above the goal if it is an invalid sequence we deduct the goal amount from the fitness to make sure the fitness stays below 0 so that a better summing sequence will be more fit.

Puzzle2 Fitness Function:

The fitness function of puzzle 2 is a little bit different from puzzle 1 and it is very simple. It is determined by two variables. The score of first bin plus second bin (we calculate this score by multiplying all population for the first bin plus the sum of all population in second bin) and an integer number called isValidBins.

isValidBins is calculated by iteration through each bin (bin one, two and three). If there is a number in any bin equals a number in list of population twice. We will increase isValidBins by one. By this method, we can check if this number is valid or not and determine the fitness of a generation. The result of fitness is the score of this generation minus isValidBins multiplied by one hundred.

Puzzle3 Fitness Function:

The fitness function of a Building is very robust because of all the attributes that a Building has to have to be considered valid and more fit than another Building.

The simplest part of the fitness function of a Building is that it adds 50 points for a door being on the bottom level and another 50 points for a lookout on top. This is done to give higher scores to legal buildings. Then for every door or lookout that is not in its valid place it deducts 50 points. This deduction lets invalid Buildings have bad fitnesses but one that has 5 doors compared to 2 doors can still be considered worse and have a lower fitness.

The next step of the function is to test that the width of a higher piece is never bigger than the minimum width at that. For every time that the function iterates over that width that is bigger it deducts 10 points. This helps the fitness function score buildings that can actually be considered a valid building higher than ones that can’t. Just like the door and lookout algorithm, the closer the building is to a valid building the better it will be scored. For example the following width sequences would be in order of most fit to least fit: {5,3,2}, {5,6,2}, and {5,6,7}.

The strength test of the building is similar to the width test only the strength test has to keep track of the strength of every piece it goes over, deducting a strength point for every BuildingPiece that is above it in the sequence. Every time the algorithm iterates over that piece and its strength is negative means that the building would collapse and the algorithm deducts 1 point for every collapse. If the building holds it will award 1 point for that iteration over each piece. For example the following strength pieces would be in order of most fit to least fit: {5,4,5}, {1,3}, {1,4,4,5}.

The last step of the fitness function is testing the validness of the building and its score. If the building is not valid or gets a score of 0 then it will be deducted 100 points but if the score is positive it will get awarded 5 times the amount of the score it got.

The validness of a building is determined by if the pieces used in the building correspond to one unique piece from the inputted file. For example, assume the possible pieces a building could have are {Door, Door, Wall, Lookout}. A valid buildings would be {Door, Wall, Lookout}, {Wall, Door}, or {Lookout, Door, Wall, Door}. An invalid building would be {Wall, Wall, Lookout}.

The score is determined by the project specifications. If the building is legal, meaning it has a door on the bottom, a lookout on the top, only walls in between, the strength of each piece is equal or greater than the amount of pieces on top of it, and the width of the pieces from bottom to top is non-increasing, then the building will get a score of (10+height^2). An illegal building will receive a score of 0.

All Puzzles:

Selection:

Selection is used for picking parents to reproduce and create children. First the population is sorted in terms of highest to lowest fitness. The function (x^2)/1000 = y is used to pick a random parent by favoring more fit sequences and choosing them more to reproduce than less fit sequences. For example a random integer would be generated by the equation randomInt(sqrt(Population\_Size\*1000))=x where randomInt chooses a number between [0, number inputted). Then x is pushed through the function (x^2)/1000 = y to determine the parent to use. With a population of 100, the most fit sequence has about a 30/325 chance of being chosen while the least fit has about a 1/300 to 2/300 chance of being chosen to reproduce. This function is essential to create children that are created from more fit parents in the population

Child Generation:

Child generation includes the selecting two parents (Selection), crossing them over (Crossover), and finally mutating them (Mutation). Please look in those section to learn how children are generated.

Crossover:

Crossover is performed similarly throughout the puzzles. It works by choosing a random point in the sequence and creating two children. One child is made from the left portion from the point in the first parent and the right portion from the point in the second parent. The second child is made the exact opposite way, by taking the left portion from the point in the second parent and the right portion from the point in the first parent.

For example, assume two sequences {1,2,3}, {4,5,6,7}, and a splicing index of 2. This would produce the two children {1,2,6,7} and {4,5,3}. Illegal crossovers will be handled in the next section about illegal children.

Mutation:

Mutations occur only in children. A mutation has two equally random options. It will either change a number in the sequence, or add a number on to the sequence in a random spot. Each puzzle does it similarly where it will chose a random choice and a random index. Then for a change in the sequence it will choose a random possible number / BuildingPiece to change at the random index. For an addition to the sequence it will choose a random possible number / BuildingPiece to insert into the sequence at the random index. For example, assume the sequence {1,2,3} and a possible number sequence {1,2,3,4,5}, a random change would look like the sequence turning into {1,4,3} while a random addition would look like {1,2,4,3}. Illegal mutations will be handled in the next section about illegal children.

Population Size:

The population size for each puzzle was chosen to be 100. (\*\*\*\* need reason)

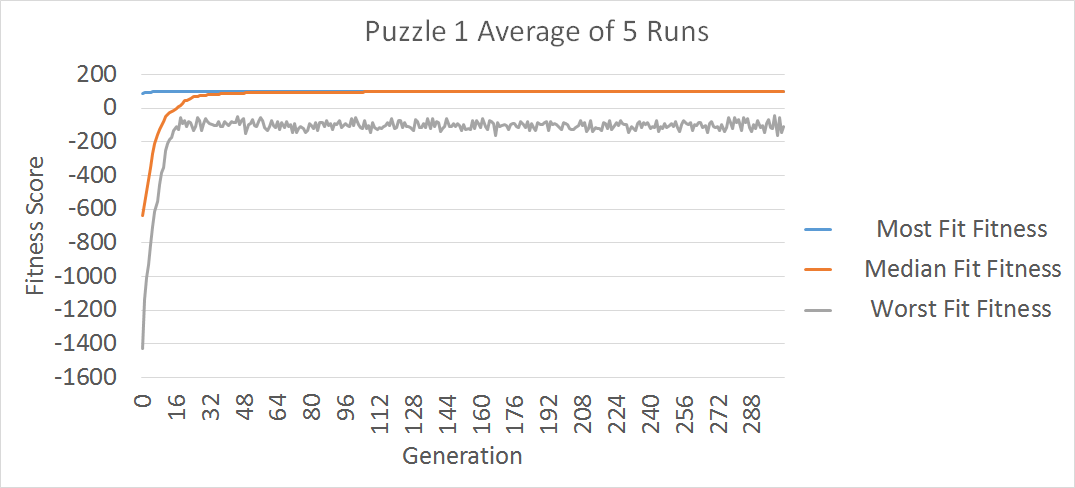
Dealing with illegal children:

Illegal children are taken care of the same way throughout all of the puzzles. An illegal child is given a low score by the fitness function making it less possible to reproduce the next generation and making it more possible to die in this generation. Legal children almost always given scores higher than illegal children except from some very exact edge cases. This way a legal child will almost always be placed higher than an illegal child and will reproduce more giving the population a higher mean score / fitness. To sum it up the algorithm does not look for illegal children directly to fix them or kill them off, it is done indirectly using the fitness function to score them lower.

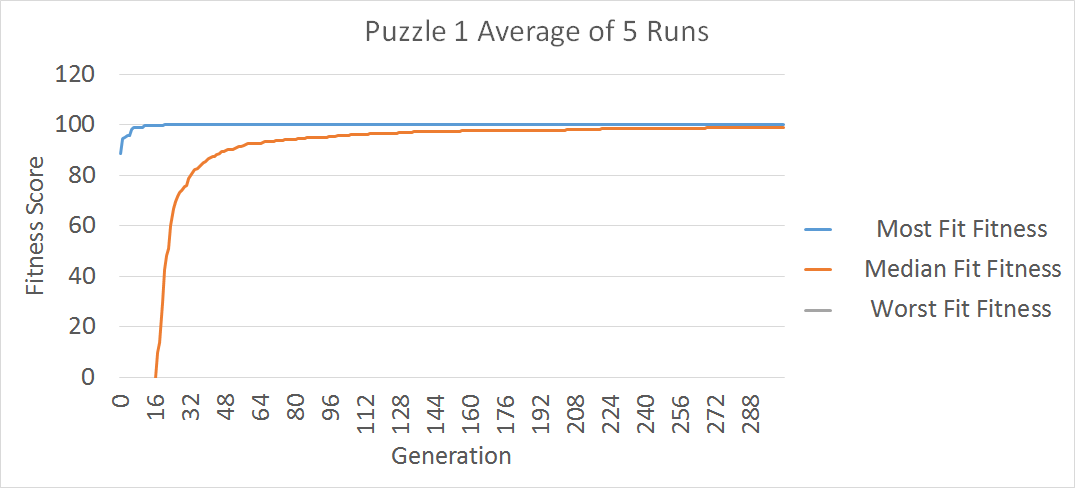
Graphs:

Puzzle1:

The graph below shows the most fit, median fit, and worst fit NumberSequences for Puzzle1 in the population over the first 288 generation. As you can see the most fit stays really fit, while the median fit gets closer to the most fit over time. The worst fit starts out with a very low score but eventually reaches its final linear value of about -100. The worst fit does not ever get any better than -100 because of the algorithm making children randomly without much thought of why they are getting that score. Thus after about the 16th generation the worst fit will be at its peak and will not get much fitter than that.

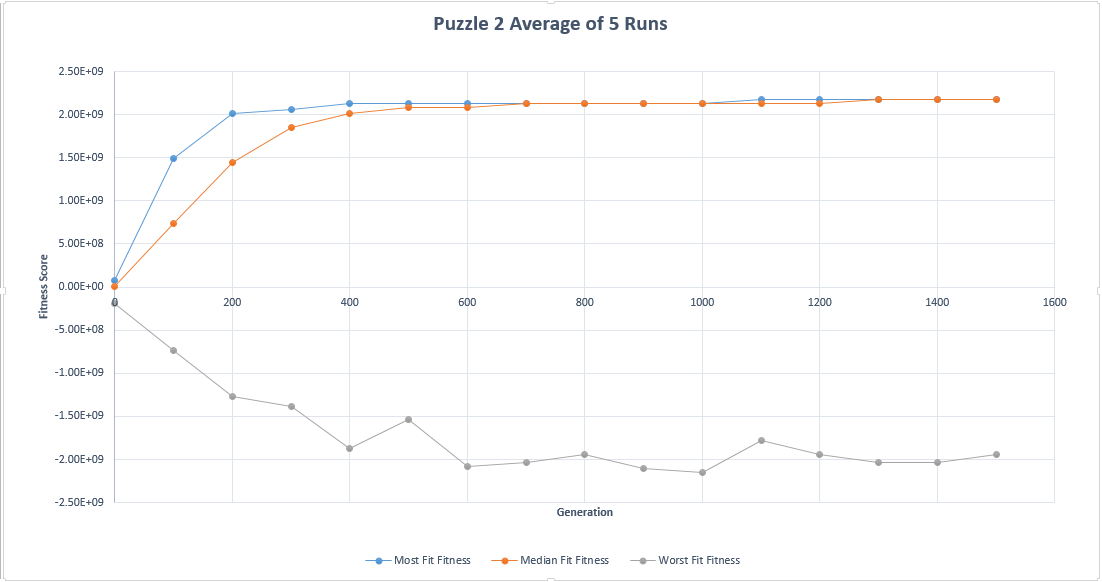


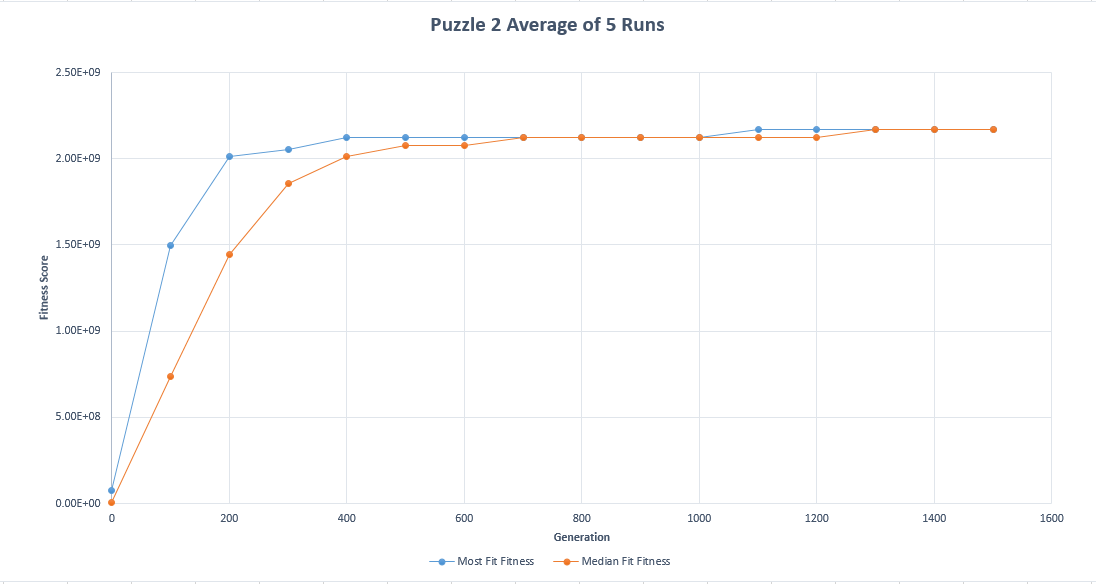
The graph below is just a close up of the most fit and median fit from the previous graph. This displays the quick ascent of the fitness of the upper two thirds of the population and how quickly the median fit will stick to the most fit after 50 generations or so.



Puzzle2:

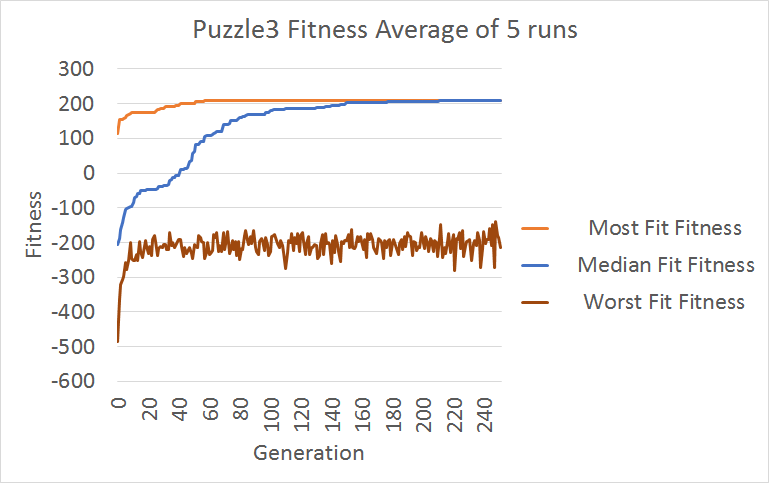
The graph below shows the most fit, median fit, and worst fit fitness score of three bins for Puzzle2 in the population over the first 1500 generation. You can see that all of three fit start with very low score (close to 0). Most fit fitness score increased rapidly and and it closed to the highest point very soon at generation 500 and almost linear (remain unchanged) from then and reached maximum at generation 1100. The medium fit was also increased very fast. However, it was always lower than the most fit from generation 0 to 700 and its development trend was approaching The most fit line. The highest point of median fit was the same as the most fit and it reached the highest point at generation 1300. Nothing to say much about the worse fit. It was always negative and changing unpredictable. However, from generation 1200. It was become more linear.

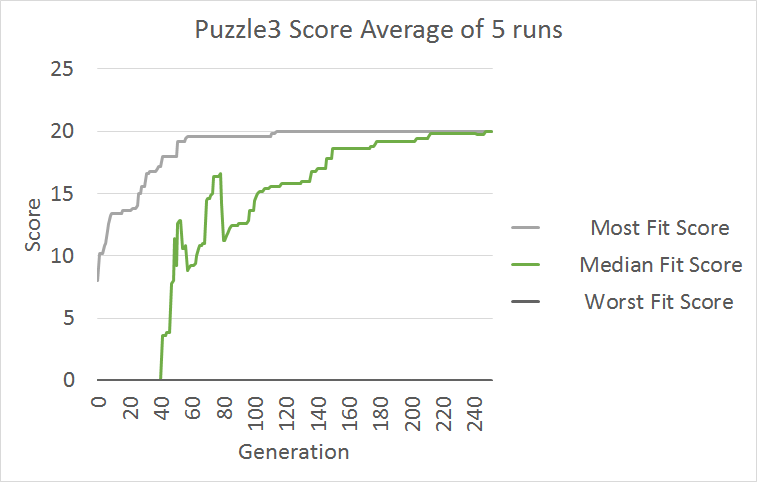




Puzzle3:

The graph below shows the fitness of Puzzle3’s population over the course of 240 generations. One interesting aspect about this graph is that the median fit Building’s fitness stays almost stagnant between generation 10 and 45. We believe this is partly due to the way the fitness function defines the genetic algorithm’s actions. The fitness function of a building may need more tweaking to get rid of the plateau in the 20-40th generations. One possible explanation for this behavior is that at some point the algorithm is trying to optimize one of the attributes of the Building and not weighing the other attributes enough. Then it has optimized the solution on that end so much that the other attributes have enough weight to change the fitness.





Question 4:

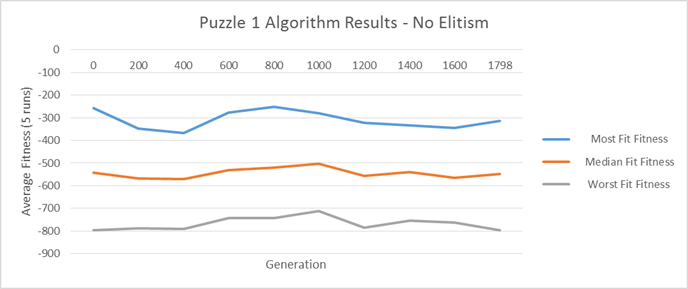
Puzzle 1:

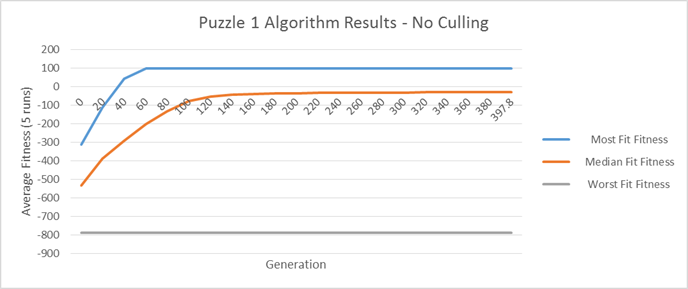
The fitness function uses a system where the result is higher the closer the sum of the numbers is to the desired goal without going over. If the sum does go over the goal, the fitness function is calculated by subtracting the goal by the sum. This results in the function producing a result that is higher the closer the sum is to the goal. For example, if the sum is one over the goal, the function will produce “-1”, whereas if the sum is two over, the function will produce “-2”.

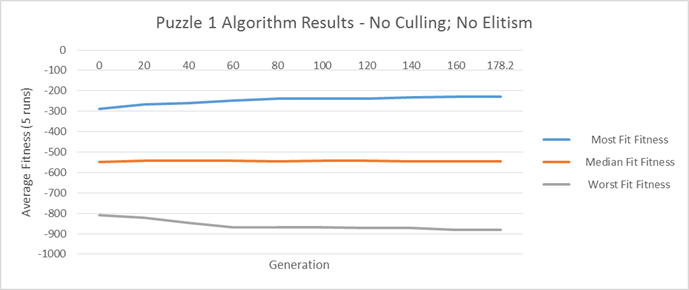
The algorithm used a population size 100. When generating children, the algorithm first sorts the list containing the population by their fitness score. Then, it eliminates the lowest 10% of the population. Once done, it creates another 10 objects by randomly taking two existing objects (the higher the fitness score, the more likely it is chosen), selecting a random spilt point, and mixing the array of Booleans that represent which numbers are within that object’s number sequence at the split point of the two parents to create two children.

Due to an array of Booleans being used to represent which numbers exist within an object in the population, illegal children are not possible. Each Boolean in the array represents whether a particular provided number from the input file exists in object’s number sequence. This array is never bigger nor smaller than the provided number list. In addition, both parents use the same split point, so taking the first half of the split sequence from one and the second half from the other results in two valid children.

This algorithm utilized elitism in that it sorted the list containing the population in such a way that the most fit objects would not be removed. In addition, the most fit objects had the best chance of being used to create children. With regards to culling, a system was implemented where the 10% of the population with the lowest fitness scores were eliminated. Both of these were implemented to increase the likelihood of creating more fit children each generation. However, it was done in a way as to not over-cull the population and allow members of the population with lower fitness scores to still produce children to allow for more “genetic diversity.”







Puzzle 2:

The two graphs below shows that there is a different between when Culling is activated, and when Elitism is activated. For the most fit, the performance between two technicals is also the same and not have much different. However, for the median fit, we can clearly see that the median fit line is not approach the most fit anymore in Elitism technical. Its development trend is same as the worst fit that is not stable and unpredictable . However, it is always higher than the worst fit line. The record line of median fit in Etilism is become a little bit stable at the end of record line. This make Elitism technical become more appealing. For the worst fit, we can see that the fluctuation margin of the drawn line in Elitism is less than in Culling. However, the score is still always negative and difference between two technical is very little and not having much to say.

