The visualisation of Genetic Algorithm results using D3.js

Antoine Gaget, William Rao Fernandes

Abstract—A Genetic Algorithm (GA) [6] is an algorithm that changes a population of individuals to resolve a problem. Problems that can be solve using GAs are various, and in most case are difficult problems. It can go from making a population of randomly generated creatures [7] to resolve machine learning problems [4]. Our GA's goal is to evolve a population of football-like players and teach them to be better at this game. We had many logs from our GA and we wanted to exploit them properly. Thus we decided to use the D3.js [1] library to visualize those data and learn from it.

Index Terms—Genetic Algorithm, Data Visualisation, Javascript, D3.js

1 Introduction

A genetic algorithm is commonly used to find solutions to complex search problems. It depend on bio-inspired operators such as selection, crossover and mutation. It starts with a number of random Individuals, named a Population. Each individuals has a set of properties, or "genome", composed of several characteristics called "gene". Those characteristics can mutate. Those genes virtually contains the shape of the wanted solution, fixed by the developers. A fitness function is applied on the population to select the most suitable individuals. The selected ones will join the new generation, and breed new individuals in order to have the same number of individuals than the first generation. The new individuals can also mutate. This operation is useful to keep a random part on the evolution and create new individuals. Mutation of the individuals' genes and "reproduction" of the population to create a new generation is the key of the evolutionary part of a GA. The algorithm stops when a solution that satisfies a criteria is found, or when the maximum number of iteration is reached.
D3.js is a library in JavaScript create to easily implement graphical content. D3.js use a data-based system, usable with different data format like json, csv and more, to bind those data to the elements. It allows users to easily create elements like charts or tables in accordance with the data they want.
Our goal here is to visualize the logs of our genetic algorithm, which teach to the individuals to play a game close to football. To do that, we will use D3.js.

2 Related Work

We started our work by searching other GA's visualisation. We mostly found website showing a GA runs while the visitor can change the parameters to alter the evolution [3, 2]. This was not interesting for us. Indeed, those visualisation were linked to the algorithm in real time while our logs were already made. The second work we found was more interesting. It is a state of the art in evolutionary algorithm visualization [5]. They also describe a tool named GAVEL, used to visualise and examine GA, mainly the mutation and cross-over parts. This work help use to understand how was done GA's visualisation, but it did not helped us to doing it. Indeed, we had to use the D3.js library, and this tool was not compatible.
With all those works, we had a good idea how to begin our first visualisations.

3 Project Description

This project is part of the Data Visualization teaching unit. The data we used came from a genetic algorithm we developed. The algorithm presented in **Algorithm 1** is a simplified version of our algorithm. We decide to not detailed to much our algorithm here because it was not the paper's subject. This algorithm generate logs in format json. Those logs describe the behavior of the algorithm, globaly and for each generation.

Data: $nb_{generation}$: Number of generation for this run,
$nb_{individual}$: Number of individuals in one generation
Result: s : Solution generation
g = Random generation of the first population;
while $i < nb_{generation}$ **do**
  $winners = matchBetweenEachIndividual()$;
  $newPopulation = breed(winners)$;
  $mutation(newPopulation)$;
  $g = newPopulation$;
  $i++$;
end
return g;

Algorithm 1: Our genetic algorithm

We decided to use those data in order to test the efficiency of our algorithm. The first thing we did was drawing the Five Design Sketches to have the big idea of how we wanted to visualize the data we had. We agreed on three visualization designs that we wanted to do to understand the result of the algorithm. The first one was a line chart showing the average characteristics per generation, combined with an other line chart showing the average mutation, also per generation. This visualization is meant to show the development of the individuals characteristics from the first generation to the last one. The mutation line chart is here to explain the uncommon fluctuation of the stats. The second one was a chart representing all the matches per generation. It is possible to scroll from generation to generation with a slider. We added a pie chart showing the percentage of use of each strategies per generation. We added the pie chart to show that the first generations will use one of the three strategies regardless of the efficiency of this one and, the more generation is going, the smarter they get, meaning they change their strategy according to the opponent strategy. The last one was a bubble chart showing the potential of each individual, per generation or for every one that played a match. The size of the bubble is calculated with the sum of the three characteristics of the individual, which means that a large bubble represents a strong individual. We also thought about making a visualization of the genealogical table of each individuals, but due to lack of time, we did not keep this idea. After finishing the five design sketches, we took some time to develop a python script to shape our data like we needed to. This script creates five json files, one for each chart we wanted to do. The first file regroups the average characteristics and the mutation rate of each generation. We used this file for the first visualization. Then we have a file, that gives us how many times a strategy is used per generation, used to create the pie chart. An other files gather all the data about the matches of each generation. Its purpose is to make the matches visualization. The two last files are about the characteristics of each individuals, for the bubble chart, and the genealogical link between

- *Antoine G. UCBL Student in Artificial Intelligence*
- *William R.F. UCBL Student in Artificial Intelligence*

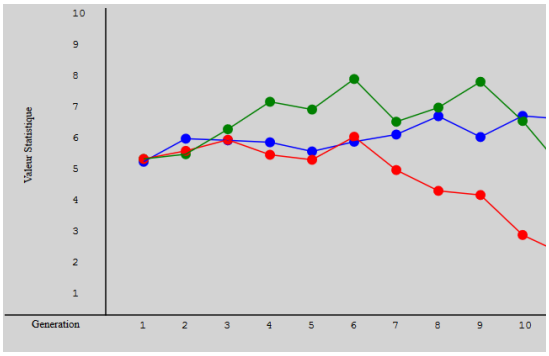

Fig. 1. Characteristics line chart.

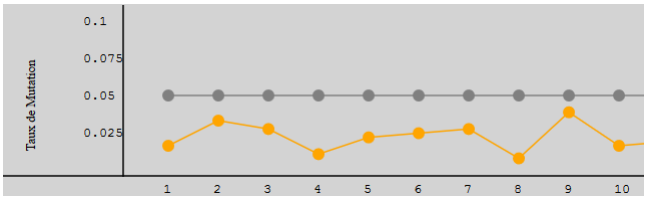

Fig. 2. Mutation rate line chart.

each individuals with his parents.
With all those data, we started to work on the first visualization, the line charts. We started from an work we did during the semester to put the basis of the chart. With that, we imported our data and draw all the points we wanted to have. After that, we simply draw lines following the points. At this point, we had three black lines representing the characteristics of the generations. The next step was to color that to have a better view of the data. Then we added some axis with names to be more aware of the visualization goal.(Figure 1) After that, we did an other line chart in order to visualize the mutation rate of each generation. We used the same type of chart, first setting the circles positions and then drawing lines between them. (Figure 2) Moreover, we put a tool tip (Figure 3) when the mouse pass over a point, in order to ease the understanding of the whole chart. The next step was the matches visualization. This visualisation is composed of three major parts. The first part is a simple slider (Figure 4) that represent the generations of the algorithm. The user can use the slider to change the generation he want to inspect. The user can also press the ”play” and ”pause” button to start the auto-sliding of the generations. This allow the user to watch step by step the evolution of the pie chart (Figure 6). The second part of the visualisation is the visualisation representing, for each generation, the number of points marked by each player for each match (Figure 5). The color represent the team of the player, used only to differentiate the two players. If the players marked the same number of points, the two circles are spread from each other. A tooltip is displayed when the mouse is over one circle. The tooltip display the number of points marked by the player and a line is drawn toward the y axis. The last part of this visualisation is the pie chart representing the repartition of the used strategies during this generation. A tooltip on one share of the pie can show the details percentage of the share.

4 Discussion

At the end of the development time, we had two finished visualisations. The first one is the visualisation of the average characteristics value per characteristics per generation and the mutation rate (real and fixed by the developer) per generation. This visualisation is a line chart. It allowed us to study the trend of the characteristics' evolution. The second visualisation is a bit more complex. It contains, for each generation, the detail of each match. For each match, the points marked by each player is represented by two points. In addition, a pie chart representing the repartition of the used strategies is displayed.

Fig. 3. Tooltip of the global visualization.

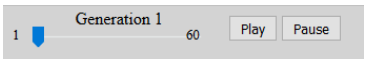

Fig. 4. Slider for the matches visualisation.

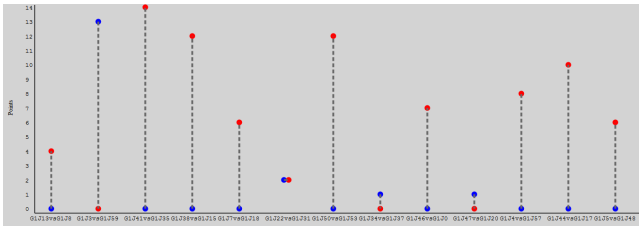

Fig. 5. Match visualisation for a given generation.

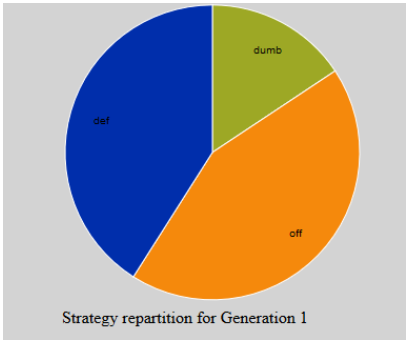

Fig. 6. Pie chart representing strategies.

This visualisation allowed us to study the evolution of the used strategies and to study match by match the generation.
Due to lack of time, we could not make all the wanted visualisation presented in the Project Description's part. This include the bubble chart representing each individual per generation and a genealogical tree for each individual to represent their ancestor and children.
We think that with all the planned visualisations, we could made our data understandable for the non-developers of our GA.

5 Conclusion

This project allowed us to have a visualization of a work we did. With it, we have been able to see the issues of our algorithm, and we can now fix them. One of the major problem is that we did not stopped the algorithm when we reach the optimal solution. It explains the randomness of the line chart after we passed the optimal generation. We also have a problem with the line chart, the mutation rate is not showing the fluctuation of the characteristics, because it is a average value. The standard deviation and a boxplot for each characteristics for each generation would have been a better indicator. However, the piechart about the repartition of the used strategies showed us that the ”dumb strategy” disappears around the end of the algorithm. This was a expected result, and it confirm our first hypothesis on the algorithm. A further work could be to improve the visualizations and to develop the missing ones in order to keep improving the GA.

REFERENCES

[1] Data-Drive Documents. https://d3js.org/.

[2] Genetic Car. http://rednuht.org/genetic_cars_2/.

[3] Genetic Walkers. http://rednuht.org/genetic_walkers/.

[4] D. E. Goldberg and J. H. Holland. Genetic algorithms and machine learning. *Machine Learning*, 3(2):95–99, 1988.

[5] E. Hart and P. Ross. GAVEL - a new tool for genetic algorithm visualization. *IEEE Trans. Evolutionary Computation*, 5(4):335–348, 2001.

[6] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[7] K. Sims. Evolving virtual creatures. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 15–22, New York, NY, USA, 1994. ACM.