Data Vizualisation : Pokemon fight

Ecole Polytechnique
INF552

Charles de Malefette - Marc-Antoine Oudotte

Introduction and presentation of the database

The objective of this project was to succeed in visualizing the power relationships between the different pokemons. We decided to place all the pokémon in a circle and to draw links between each pokémon. A link connects two pokémon and corresponds to a confrontation between these pokémon. The color of the link will indicate who won. The database we used was divided into three parts that we divided into three separate .csv files :

The first file *pokemon_formated.csv* listed all the pokemons in the following form : index, types, name, generation, legendary

— The index is the number of the pokemon in the Pokemon universe

— The types define the strengths and weaknesses of a pokemon in relation to the attacks it can suffer or inflict. The existing types are 18 and are the following : Fire, Water, Grass, Bug, Electric, Normal, Ice, Fighting, Rock, Ground, Flying, Psychic, Poison, Ghost, Fairy, Dragon, Dark, Steel. A pokémon can have more than one type at the same time.

— Name is the name of the pokémon according to the English version of the franchise

— There have been 6 generations of pokémon over the years, each generation contains about 150 pokémon. The feature *generation* indicates the number of the generation.

— Some pokémon are legendary pokémon, meaning they are very rare and often very powerful. The feature *legendary* is a boolean that indicates if the pokémon is a legendary pokémon. This feature is useful for our project since legendary pokémon will probably be of great interest in terms of battles.

Here is an example for a pokémon : 7,"['Fire', 'Flying']",Charizard,1,False

The second file of the database, *combats.csv* contains the results of clashes between pokémon and is presented in this form : First_pokemon, Second_pokemon, Winner The pokemons are considered by their indexes. For example, the line "3, 13, 13" tells us that pokemon number 13 wins in a battle against pokemon number 3.

The last file is a folder named *imagepokemon* which contains the images of all the pokemons studied. This file will be used to improve the graphic rendering of our data display.

We had a bad surprise when we noticed that the indexes of the pokemons in the file *pokemon_formated.csv* were not the exact indexes of the franchise and thus not the same as those of the file *imagepokemon*. We had to perform a reindexing to correct this problem. Without going into details, this problem came from the presence of the mega-evolutions of some pokemons which caused a shift in the indexes.

In this report, we will first present the visual results we were looking for and then talk about the method of using the database to obtain these results.

1 Visualisation des affrontements

1.1 Chord Diagram

The choice of the diagram to display was an important choice for the final rendering. After searching for several different types of diagrams, we found the chord diagrams. Chord diagrams visualise links (or flows) between a group of nodes, where each flow has a numeric value. For example, they can show migration flows between countries.

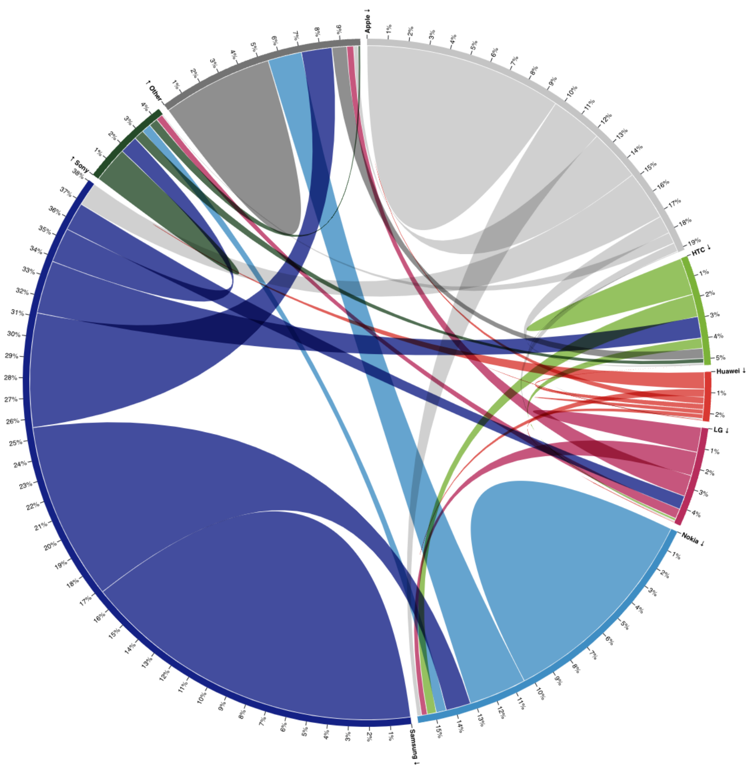


FIGURE 1 – An example of a chord diagram based on the result of a phone survey

We were convinced by the graphical rendering that such a diagram offered as well as the multitudes of functionalities that we could add to it. The creation of such a diagram is done using a square matrix of size n where n is the total number of data to analyze. In short, the index element (i,j) of the matrix measures the link from i to j. In our case, each pokemon will occupy one row of the matrix and the link between one pokemon and the others will depend on the outcome of their fights. In our study, the diagonal of the matrix will be composed of 0 because a pokemon cannot self-fight. The use of such a diagram allows to visualize easily many links between data and to see global dynamics emerging and not local ones. Indeed, the initial goal of our project was not to know specifically if a pokémon wins or loses but rather to know which types are dominant over others.

This choice of diagram led to some difficulties in displaying the results. Initially we wanted to display all types and generations at once but the number of links and pokémon was much too large so the links could not even be displayed. These were actually too thin. So we decided to use filters to display only certain generations and types so that the results are visible, readable and interpretable. We will explain the use of filters in more detail in a later section.

1.2 Design of the results

In this section. We will explain all the graphical details that we have added to facilitate the visualization of the data.

First of all, we put a plain sky blue background and an image of three mythical pokemons of the franchise. The objective was to add an explicit reference so that the uninformed user can quickly understand in which universe this data was taken. However, we did not want to overload the graph, so we opted for a rather uncluttered and discreet image.

FIGURE 2 – The background of the site

Un second point qui a été très important dans notre projet a été l'attribution de couleurs à chaque type. Comme précisé plus tôt, nous avons ordonné les pokémons par type dans la matrice. Ainsi, les pokémons de même type seront proches dans le diagramme. L'avantage de cela est qu'en ajoutant une couleur par type, on obtiendra un regroupement des couleurs ce qui créera des pôles propres à chaque type. Nous avons ajouté des arcs de couleurs autour de chaque type pour améliorer le rendu. We have also indicated the names of the types to avoid any ambiguity.

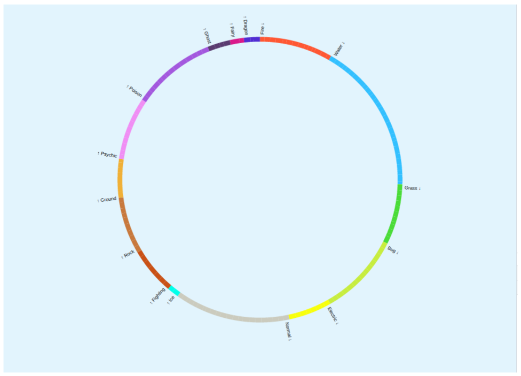


FIGURE 3 – Grouping of pokemons by type, one color arc for each typee

The link between two pokémon will be the color of the one who wins the fight. Since the power relations between pokémon are very much linked to type, the dynamics will be much more readable by grouping pokémon by type. For example, ground pokémon tend to beat electricity pokémon, so by grouping by type, we will see a brown dynamic invade the yellow camp which is much more interpretable than if we had not grouped the types. Below is the result if we leave only the soil and electricity types. We can clearly see that the ground type invades the yellow camp while the yellows have difficulties to infiltrate the browns.

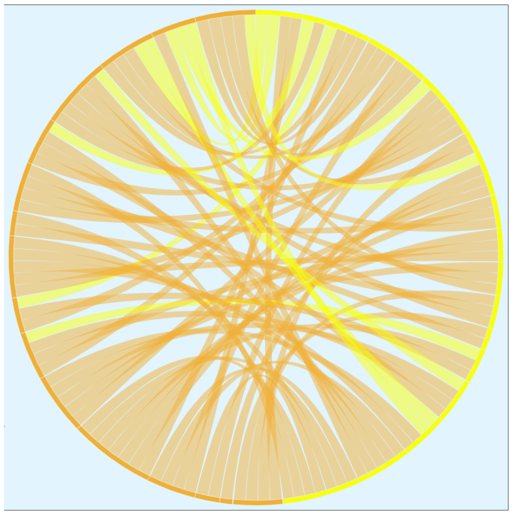


FIGURE 4 – Confrontation between two types : ground versus electricity

We have also added the possibility to have a live image of the pokemons when we go through the circle. We will talk about this in more detail in the last part.

2 Exploitation of the dataset

2.1 Creation of the matrix

To create the diagram, you have to convert the database into a matrix. This matrix is a square of the same size as the number of pokémon to study. Each line corresponds to a pokemon. The algorithm accesses the database *combats.csv* which contains the results of the clashes and updates the matrix after each clash. Thus for each i, j combination of pokémon, if the clash did not take place, the matrix is 0 in position i, j. If i won, the matrix will be k in position i, j and p in position j, i where p and k are strictly positive integers with $k > p$. If i has lost we do the reverse. p and k characterize the thickness of the link connecting i and j. k is the thickness at the beginning and p the thickness at the end. This database does not contain all the clashes for the sake of relevance. All the fights are not interesting to simulate because in the pokemon universe they would not have the possibility to fight each other. At the beginning, we did not understand why this database did not list all the results of the clashes. Indeed, once the algorithm that decides the winner is implemented, it doesn't cost much to test it on all possible clashes. We then hesitated to generate this database ourselves and it is while doing so that we understood the relevance of having only a part of the results. It was also a good way to limit the number of links to display on the screen which was a limiting factor for our project.

2.2 Use of filters

The diagram can only display a certain amount of data. We had to give up the idea of displaying all pokemons, all generations and all battles. We then created filters in the form of buttons that the user can click directly on the site. With these buttons, the user can select or deselect the types he wants. He can also choose the generation of pokémon to display. He also has the possibility to remove the legendary pokémon. We created this filter because legendary pokémon are often stronger than others and in some cases, their presence can distort the interpretations that can be made of the diagram. A last filter has been added to remove clashes between pokémon of the same type. Indeed, when two pokémon of the same type clash, the result is not very interesting in the case of our study since we are mainly interested in the relations between types. Below the rendering of the buttons allowing to activate or deactivate the filters on the page. This also allowed us to explicitly show the color code used for the types.

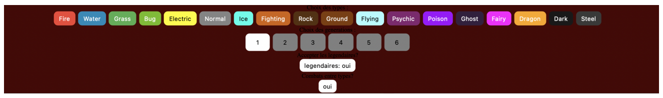
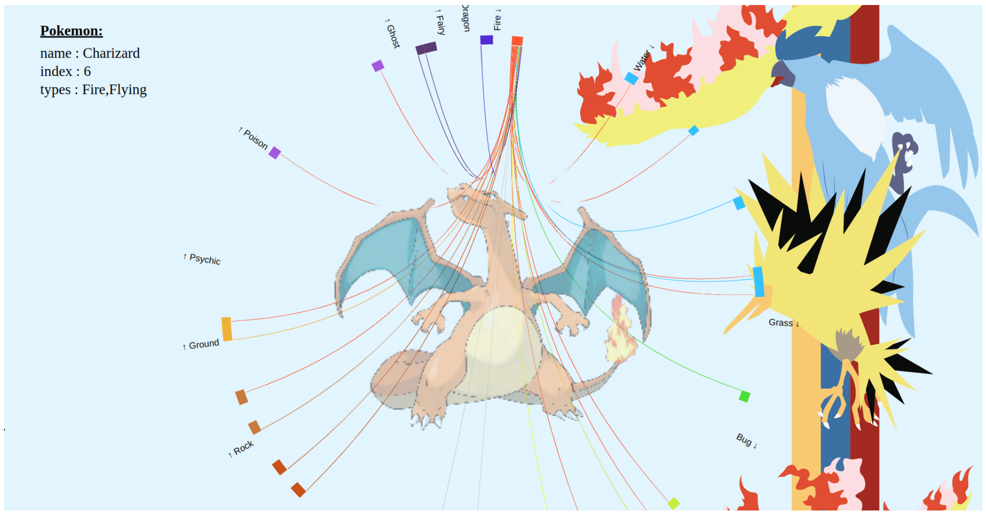


FIGURE 5 – The filters buttons

2.3 Dynamic display of results

At this stage of the project, we were satisfied with the result and especially satisfied with the global interpretation we could make, especially on the power relations between types. However, we wanted to add the possibility for the user to analyze each pokémon independently of their type. We added another feature : when the user moves the mouse over a pokemon, the graph only considers this pokemon and removes all the others to display only this one. A text box is then displayed to indicate the information about this pokemon and only the links that concern it will be displayed on the screen. In addition, an image of the pokemon in question will be displayed so that the user can locate the exact pokemon he is studying.



Pokemon:
name : Charizard
index : 6
types : Fire,Flying

FIGURE 6 – Analysis of one specific pokemon

In the example above, we are analyzing the pokemon **Charizard**. We can see the information about it as well as the results of all the clashes in which it has been involved. The colors allow us to see that he dominates the pokemons of water type but that he has a weakness against the pokemons of fire type. This is how we understood the real interest of this project for a user. For fans of pokemon games, it is common to be stuck in the game against a pokemon that you can't beat. By doing some research, we manage to get its statistics and the method to beat it, but there is no real solution to know the ideal team to beat it. With this project, if you are looking for how to beat a particular pokemon, you can easily see the pokemons to use to beat it by going to the pokemon in question.

3 Overall code structure

We will now quickly present the structure of the code. The file *pokemon.html* creates the interface with the filters and then calls the function *launch()* of the file *pokemon.js*. The function *launch()* initializes the canvas and launches the function *createGraph()* which generates the graph. We have dissociated the function *launch()* and *createGraph()* because we reset the graph with each modification on the filters and the dissociation of the two functions allows to create the canvas only once. The function *createGraph()* generates the clash matrix by calling the function *fillmatrix()* and displays it according to the design rules we have set.

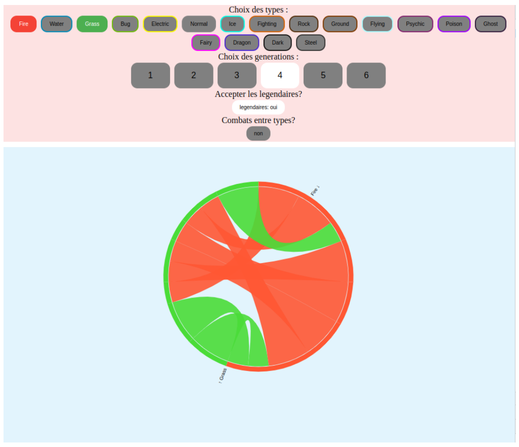
The variables *setValue, setgeneration, setlegendary* and *setsametype* are updated each time the user modifies a filter and call *createGraph()* again to update the graph which will be modified accordingly. All global variables are stored in a constant variable *ctx*.

The function *onmouseovered()* runs as soon as the user uses the mouse to analyze a pokemon and updates the graph display by setting the opacity of all useless links to 0. The function *onmouseouted()* allows to redisplay the whole graph once the mouse goes away. The functions *color()* and *colorribbon()* set the colors of the links and arcs according to the types of pokemons.

Conclusion

We have achieved our goals with this project. We even added new features like filters or pokemon specification. We tried to reproduce the pokemon atmosphere by playing with a precise color code and adding images. The biggest difficulty was to combine the fact of having a lot of data with its visualization in a graph. The filters helped us a lot for that. This project was very satisfying to do since it offers an interface that we both would have liked to have as kids when we played pokémon.

Below is a final rendering of our project.


Choix des types :

Choix des generations :

Accepter les legendaires?
legendaires: oui
Combats entre types?
non

FIGURE 7 – Final result of our project with two types

References

[1] Database for pokemon images : `https://www.kaggle.com/datasets/kvpratama/pokemon-images-dataset`

[2] Database for pokemon battles : `https://www.kaggle.com/code/jonathanbouchet/pokemon-battles`

[3] Site utilisé pour la création de notre diagramme : `https://observablehq.com/collection/@d3/d3-chord`