Marvel Character Screen-time Viz - CMPM 290

Abhiram M. Venkatesha
amadenur@ucsc.edu

Albert Garcia
awgarcia@ucsc.edu

Cassia Artanegara
cartaneg@ucsc.edu

Abstract

This paper presents an adjacency matrix representation of the on-screen appearance durations of 99 most popular characters in the Marvel Universe by analyzing a set of movies. The horizontal axis in the adjacency matrix representation is the set of characters and the vertical axis corresponds to the set of movies. Data is extracted from an IMDb webpage using parsing tools in python. D3.js in conjunction with HTML DOM and CSS styling is used to provide a rich webpage display of the adjacency matrix.

Index Terms: HTML Webpage—Visualization techniques—Adjacency Matrix—;

1 Introduction

Marvel Comics have been around since 1939, featuring a broad range of characters in a myriad of movies. Most of the movies have multiple character appearances other than the movie's protagonist. It is also the case that the protagonist in one of the movies might have a cameo role in a different Marvel movie. The goal of creating this visualization was to help the user discover relationships between the characters and movies, such as the relationships between characters within a certain movie, how those relationships change as the story within the Marvel Universe changes, or how actors fluctuate within the Marvel franchise.

2 Related Works

We built our visualization based off of preexisting data visualization techniques, combining elements of an adjacency matrix with small multiples.

3 Methodology

The project was divided into two parts. The first objective was to extract the data from an IMDb webpage and the second was to use D3 to visualize the data in an adjacency matrix representation.

The data extraction was done using Python 3. Data from an IMDB list of characters and their respective screen times were parsed with the Python package Beautiful Soup. A dictionary of characters and their corresponding descriptions was constructed using the find_all method on the Beautiful Soup object. Each dictionary value contained html tags and other characters that were then filtered out using Python's regular expression package "re."

The web_to_dict() method in the WebToDict.py file returned the dictionary containing the clean data with keys as the character names and values as their movie and on-screen time information. (Example entry in the dictionary: "Col. James 'Rhodey' Rhodes": actor: 'Terrence Howard', IRON MAN: '8:15', IRON MAN 2: '18:45'). Once the dictionary was obtained, DictToCSV.py file converted the dictionary into a csv file, which was later manipulated on-screen using D3.

In addition to the character name, actor name, and screen times (with their respective movie titles), the csv file also includes each characters' total screen time, the release order of all the movies, and the chronological order of the movies within the Marvel universe.

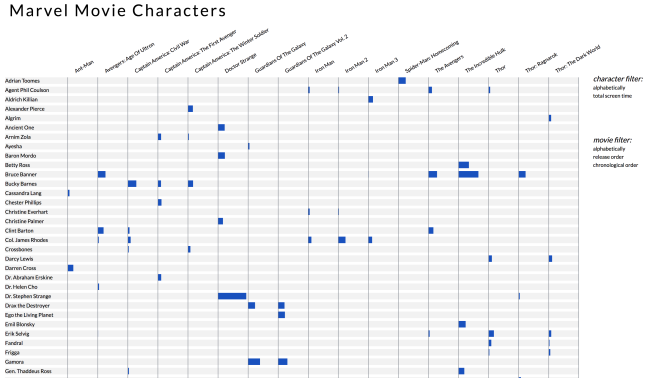

Figure 1: Visualization upon initial load.

Total screen time was calculated by adding up all the screen times for a particular character across all movies. To determine release order, each movie was assigned a number from 1 to 17 with 1 representing the earliest release and 17 representing the most recent release. Chronological order was determined in a similar fashion.

In order for the data to be utilized in a useful manner, the d3.csv function put the data into an array that was then sorted alphabetically by movie name and alphabetically by actor name. This is the default filter for displaying the data when it is initially loaded (see Figure 1), but the user may select one of two character filters and one of three movie filters. When the user clicks on a new filter, a new sorting algorithm rearranges the data stored in the array. The newly ordered data is then displayed in the new filtered order.

Each horizontal line pertains to a character while each vertical line pertains to a movie. The screen time for a particular character in a particular movie is drawn as a blue rectangle (orange on hover). The width of the rectangle is determined by how much time that character is on-screen in relation to the character with the most screen time in a movie (which is Tony Stark, clocking in at 77.25 minutes in Iron Man). The equation to calculate box size is as follows:

$$box\_size = \frac{screen\_time\_for\_current\_character}{longest\_screen\_time}$$

Because there is a large amount of data to represent, the primary design goal was to create a simple and streamlined user interface. Users can select how they wish the data to be organized from the sidebar on the right. When the user hovers on a data element, the entire row for that character is highlighted so that the user can easily identify other movies that character appears in. Additionally, the movie name, character name, and screen time for that character in that movie are written in the sidebar. Figure 2 demonstrates these design choices with a new filter applied to the data.

4 Discussion

The Marvel Movie Characters visualization primarily focuses on screen time of characters, but it ultimately allows us to derive interesting relationships between characters and movies. For example, Tony Stark is, unsurprisingly, the character with the most screen

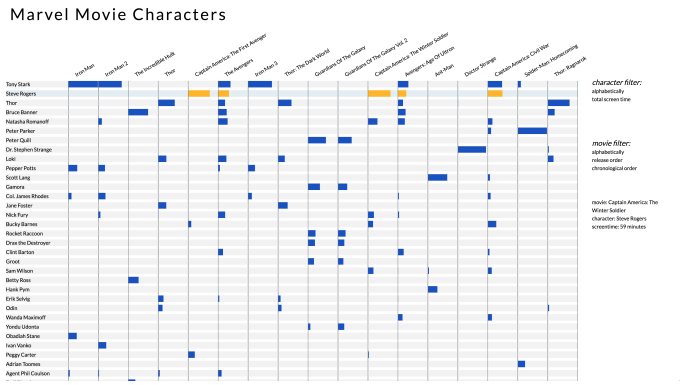

Figure 2: Visualization filtered by total character screen time and chronological movie order. The user is currently hovering on the data point for Steve Rogers in Captain America: The Winter Soldier (cursor hidden).

time in all three of the Iron Man movies. He also happens to hold the highest total screen time out of all other characters. From the visualization, it is easy to see that Tony Stark is seen on screen at a much higher proportion than his supporting costars. In contrast, the highest screen times for the two Avengers movies have a more well-balanced cast in terms of screen time.

5 Conclusion

This visualization of characters' screen time in Marvel movies can help display trends that can only be found while looking at the data in this manner. This visualization could also be easily adapted to other other movie, TV, or book series, or even new entirely new data. There are many ways we would like to improve this visualization in the future, some of which are listed below:

- Modify some visual aspects for easier and faster data reading (specifically, highlighting movie columns in addition to character rows on hover and exploring new color variations with increased contrast)
- Create new filters to explore relationships and patterns in new ways
- Increase user activity by allowing them to narrow down the data set (e.g. only look at the Iron Man movies or allow user to create their own subset)
- Add photographs of characters to the sidebar for more visual information
- Implement animations during data redrawing for enhanced visual effect