Assignment 3
Flat to Hierarchical Data Conversion
and the D3 Sunburst Layout

Introduction

The purpose of this assignment is twofold: 1) To give you experience in using JavaScript to convert flat data to hierarchical form, and 2) To familiarize you with the D3 sunburst layout.

The Consumer Price Index measures the changes in the prices paid for a basket of consumer goods and services. In this assignment, you will be provided a data set representing this set of goods and their weights in the form of a flat CSV text file. Although the CSV file is flat, there is enough information in the records for you to construct a hierarchical JavaScript object containing the expenditure categories and their sub categories. The origin of the data is the Bureau of Labor Statistics. You can view the information in human readable text form here: http://www.bls.gov/cpi/usri_2015.txt

CSV Format

The file **cpi2015.csv** contains all the input data you will need. The first 9 categories/subcategories in the file are shown below. You can compare this with the text form referenced above.

```
ID,Level,Parent,CPI_U,CPI_W,"Expenditure_category"
0,1,-1,100.000,100.000,"All items"
1,2,0,14.973,16.204,"Food and beverages"
2,3,1,14.015,15.318,"Food"
3,4,2,8.230,9.374,"Food at home"
4,5,3,1.098,1.258,"Cereals and bakery products"
5,6,4,.371,.425,"Cereals and cereal products"
6,7,5,.043,.049,"Flour and prepared flour mixes"
7,7,5,.192,.214,"Breakfast cereal"
8,7,5,.135,.161,"Rice, pasta, cornmeal"
```

The first column is the ID of the entry. The second column is the category level. It is not really needed, but is provided for your convenience. It is the depth in the category tree where the entry resides. Level 1 is “All Items” with ID 0, which is the root of the tree. The third column is the ID of the *parent* category of the entry. The root category has no parent so its parent ID is set to -1. Take note that this is the reverse of the hierarchical JSON format we’ve been dealing with in class (e.g. flare.json) where each category identifies the *children* objects. The next two columns are the CPI-U and CPI-W weights in percent. For the purposes of this exercise, use the wage earner value CPI-W in your chart. The last column is the description of the expenditure category. You should notice the CPI-W value of every category is the sum of the CPI-W values all its direct descendants. The value of the root category is of course 100%.

Reading and Converting the CSV File

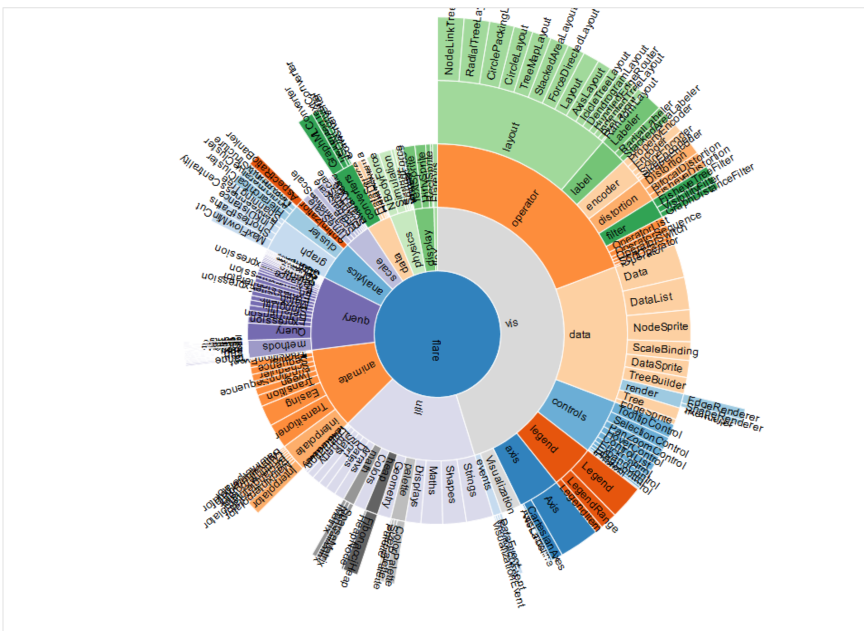
You can use the d3.csv() function to read and parse the CSV file. To convert to a hierarchical JavaScript tree object, you will have to write short piece of JavScript code. This is not a CS course on Algorithms, so you can choose to scour the web for any (in)elegant solution you can borrow or steal and use as your

own. You can start with http://stackoverflow.com/questions/11934862/d3-use-nest-function-to-turn-flat-data-with-parent-key-into-a-hierarchy and maybe dig a little deeper later.

Visualization

Recall that a sunburst is just one in a family of D3 Hierarchy Layouts (see: https://github.com/mbostock/d3/wiki/Hierarchy-Layout). Once you have your nested JSON object, you can probably just as easily implement any one of the listed layouts. However, here we're explicitly asking for a *sunburst* partition layout. Specifically, the requirement of this assignment is to implement a "*Zoomable Sunburst with Labels*". Google that phrase. You might find one of many implementation examples to your liking. In a sunburst, the length of each arc corresponds to the value. Because it is a zoomable sunburst, clicking on any arc promotes that category to be the root of the tree, and only that subtree will be visible in the layout. Clicking on the root of the subtree (middle disc) will move the root of the sunburst layout to the parent of the current root.

Because the depths of the leaves of the CPI Weights tree are not the same, your sunburst will look similar to the following layout representing flare.json. It will not look like a full round disc.



Requirements

- Implement the Zoomable Sunburst with Labels as yet another reusable object in your library file `d3wrapper.js`. The constructor name should be D3WRAP.ZoomableSunburst. Just like the other reusable assets which you have written so far, it should take three arguments: 1) the DOM object to which to append the layout. 2) The data to be bound to the layout. In this case a JSON object in flare.json style format. 3) A parameters object that specifies the size, margins and so on of the layout. You may include a string length limit to truncate the name (labels).
- Use our standard HTML template. Name the file sunburst.html, and the corresponding JavaScript file sunburst.js

- Write the CSV to hierarchical JSON format converter code. The code may be in a named function or in an anonymous callback function. When the DOM is loaded, use the d3.csv() function to read cpi2015.csv, then use your converter function to generate the JSON object. At the minimum, use the same properties as flare.json: `children`, `name` and `size` (taken from `CPI_W`). You may include other properties like the `ID`, `Level` and the unused `CPI_U`. You MUST create the global object named **`cpiFlare`** that references the object so it can be inspected when the assignment is graded.
- Finally, create a new D3WRAP.ZoomableSunburst() object in the chart1 DIV of the template

Point Allocation

This is a 10 point assignment

- Correct construction of cpiFlare from cpi2015.csv – 3 points
- Correct initial representation of the data in sunburst form – 3 points
- Ability to zoom in and out on click (descend or ascend tree) – 2 points
- Visually pleasing layout (choice of colors, margins, dividers) – 2 points