Network dataset visualisation using D3.js

Want to move from being a D3.js visualisation user to being a big data visualisation ninja? Let’s do one more example using D3.js. As before, you are encouraged to *do* this activity, but if you prefer then you are also welcome to read over the code and comments in order to gain a broad understanding of how such a visualisation can be created, or skip the exercise and move to the next section.

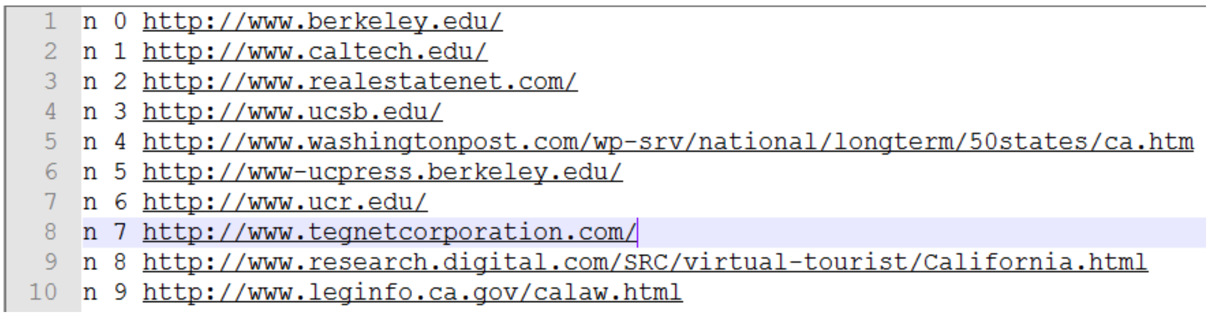
This example uses D3.js force layouts to visualise a dataset containing approximately 10000 websites and 15000 links between them. The circles representing the nodes have a radius and opacity based on their importance, or "rank", determined in mini MOOC 3. For an excellent overview of force layouts, see this blog post: https://medium.com/@sxywu/understanding-the-force-ef1237017d5#.micq7nqwl

This source code is modified from code available here: https://bl.ocks.org/mbostock/4600693

Data files

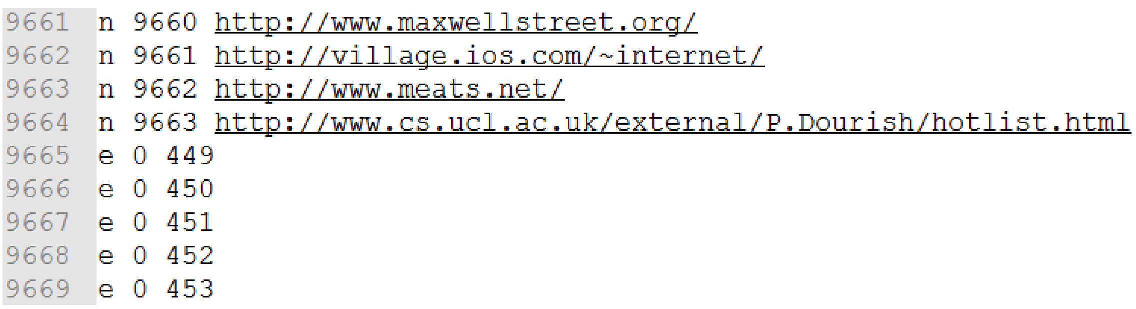
Original graph data file gr0.California

The data file was obtained from http://www.cs.cornell.edu/Courses/cs685/2007fa/data/gr0.California and contains a subset of the world wide web generated by the search query “California”. Note that this data set is quite old so some links may no longer exist.



```
n 0 http://www.berkeley.edu/
n 1 http://www.caltech.edu/
n 2 http://www.realestatenet.com/
n 3 http://www.ucsb.edu/
n 4 http://www.washingtonpost.com/wp-srv/national/longterm/50states/ca.htm
n 5 http://www-ucpress.berkeley.edu/
n 6 http://www.ucr.edu/
n 7 http://www.tegnetcorporation.com/
n 8 http://www.research.digital.com/SRC/virtual-tourist/California.html
n 9 http://www.leginfo.ca.gov/calaw.html
```

Figure 1: Dataset gr0.California showing the node numbers and name (URL)



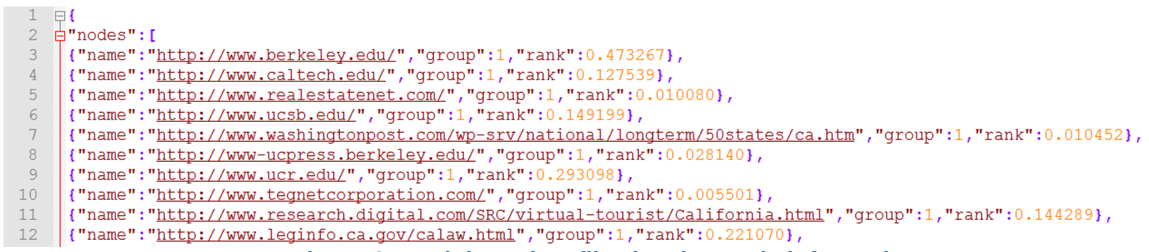
```
n 9660 http://www.maxwellstreet.org/
n 9661 http://village.ios.com/~internet/
n 9662 http://www.meats.net/
n 9663 http://www.cs.ucl.ac.uk/external/P.Dourish/hotlist.html
e 0 449
e 0 450
e 0 451
e 0 452
e 0 453
```

Figure 2 Dataset gr0.California showing the links (edges) given by a starting node and target node

JSON data file graph.json

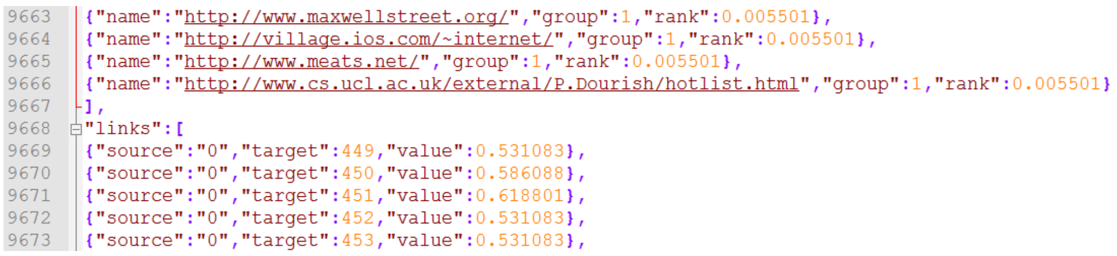
The original dataset was post-processed to generate a valid JSON data file. The figures below show the data stored in JSON format.

© Queensland University of Technology | CRICOS No. 00213J



```
{
"nodes":[
{"name":"http://www.berkeley.edu/","group":1,"rank":0.473267},
{"name":"http://www.caltech.edu/","group":1,"rank":0.127539},
{"name":"http://www.realestatenet.com/","group":1,"rank":0.010080},
{"name":"http://www.ucsb.edu/","group":1,"rank":0.149199},
{"name":"http://www.washingtonpost.com/wp-srv/national/longterm/50states/ca.htm","group":1,"rank":0.010452},
{"name":"http://www-ucpress.berkeley.edu/","group":1,"rank":0.028140},
{"name":"http://www.ucr.edu/","group":1,"rank":0.293098},
{"name":"http://www.tegnetcorporation.com/","group":1,"rank":0.005501},
{"name":"http://www.research.digital.com/SRC/virtual-tourist/California.html","group":1,"rank":0.144289},
{"name":"http://www.leginfo.ca.gov/calaw.html","group":1,"rank":0.221070},
```

Figure 3 graph.json data file showing node information



```
{"name":"http://www.maxwellstreet.org/","group":1,"rank":0.005501},
{"name":"http://village.ios.com/~internet/","group":1,"rank":0.005501},
{"name":"http://www.meats.net/","group":1,"rank":0.005501},
{"name":"http://www.cs.ucl.ac.uk/external/P.Dourish/hotlist.html","group":1,"rank":0.005501}
],
"links":[
{"source":"0","target":449,"value":0.531083},
{"source":"0","target":450,"value":0.586088},
{"source":"0","target":451,"value":0.618801},
{"source":"0","target":452,"value":0.531083},
{"source":"0","target":453,"value":0.531083},
```

Figure 4 graph.json data file showing link information

HTML + javascript

We now present the source code for generating a network visualisation using D3.js. Feel free to copy and paste this code into an editor to create the HTML page, or try coding it yourself!

d3_graph.html

```
//setup
<!DOCTYPE html>
<meta charset="utf-8">
<style>

.link {
  fill: none;
  stroke: #555;
}
.node {
  stroke: #fff;
  stroke-width: 1.5px;
}

</style>
<body>
<script src="scripts/d3.js"></script>
<script>
// setup drawing size
var width = 5500,
    height = 5500;

// setup the colour scheme used for colouring the nodes
var color = d3.scale.category20();


// Create a new force layout object by specifying the link distance and
```

© Queensland University of Technology | CRICOS No. 00213J

```
strength, and the size of the domain.
var force = d3.layout.force()
    .linkDistance(10)
    .linkStrength(5)
    .size([width, height]);

// Add svg drawing with specified width and height attributes. This is the
canvas where the graph will be drawn.
var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

// load the graph dataset in json format
d3.json("../data/graph.json", function(error, graph) {
  if (error) throw error;
// get the nodes from the graph dataset. Can you think of a way to remove
the unimportant nodes at this step?
  var nodes = graph.nodes.slice(),
      links = [],
      bilinks = [];
// build the links between nodes, using an intermediate node to create
curved edges
  graph.links.forEach(function(link) {
    var s = nodes[link.source],
        t = nodes[link.target],
        i = {}; // intermediate node
    nodes.push(i); // add the intermediate nodes to the set of nodes
    // create links between the source node and the intermediate node,
    // and the intermediate node and the destination node.
    links.push({source: s, target: i}, {source: i, target: t});
    bilinks.push([s, i, t]);
  });
  // add the nodes and links then start the force layout simulations
  force
      .nodes(nodes)
      .links(links)
      .start();
  // setup style of links
  var link = svg.selectAll(".link")
      .data(bilinks)
    .enter().append("path")
      .attr("class", "link");
  // setup tooltip for displaying url
  var tooltip = d3.select("body")
      .append("div")
      .attr("class", "tooltip")
      .style("position", "absolute")
      .style("z-index", "10")
      .style("font-size", 50)
      .style("visibility", "hidden");

/* This section of code specifies how the nodes will be drawn.
All of the nodes are selected and are represented by a circle,
with a radius proportional to the page ranking that is stored
in the graph.json data file. The opacity is also mapped directly
to the page rank, so unimportant web pages are not visible in the
visualisation.
*/
```

© Queensland University of Technology | CRICOS No. 00213J

```
  var node = svg.selectAll(".node")
      .data(graph.nodes)
    .enter().append("circle")
      .attr("class", "node")
      .attr("r", function (d) {return 100 * d.rank;}) // scale the radius
by the calculated page ranking
      .style("fill", function(d) { return color(d.group); })
 .style("opacity", function(d) { return d.rank; }) // opacity specified by
page ranking
 // Add tooltips to display url on mouseover
 .on("mouseover", function(d) {
      tooltip.transition()
               .style("left", d.x)
           .style("top", d.y)
               .text(d.name)
               .style("visibility", "visible");
 })
 .on("mouseout", function(d) {
      tooltip.transition()
               .style("left", d.x)
           .style("top", d.y)
               .style("visibility", "hidden");
      })
      .call(force.drag);


// This section updates the drawing of the graph as the force calculations
are processed.
// The links and nodes are transformed based on x and y values
  force.on("tick", function() {
    link.attr("d", function(d) {
      return "M" + d[0].x + "," + d[0].y
          + "S" + d[1].x + "," + d[1].y
          + " " + d[2].x + "," + d[2].y;
    });
    node.attr("transform", function(d) {
      return "translate(" + d.x + "," + d.y + ")";
    });
  });
});


</script>
</body>
```

Running the code

To be able to run this example you will also need a simple HTTP server as described in the previous example. On Windows, if you have Python installed you can run the HTTP server from a command window. Open a command window in the directory where you have the d3_graph.html file saved (or whatever filename you chose) and type the command

```
python -m http.server 8080
```

Then, navigate to http://localhost:8080/d3_graph.html in your web browser to run the script. Note that it may take some time for the graph to equilibrate due to the large amount of data being processed on your desktop computer!

© Queensland University of Technology | CRICOS No. 00213J

The Result

This visualisation demonstrates the complex linking structure contained in just a small subset of the World Wide Web. We've utilised the design elements of intensity (via opacity) and size to demonstrate the relative importance of each node (or webpage) in the dataset. This importance was calculated using a page ranking algorithm similar to that discussed in the previous course - Big Data: Mathematical Modelling. Why not try modifying the parameters controlling the size and opacity of the nodes (circles)? Are there ways you can think of to improve this visualisation?

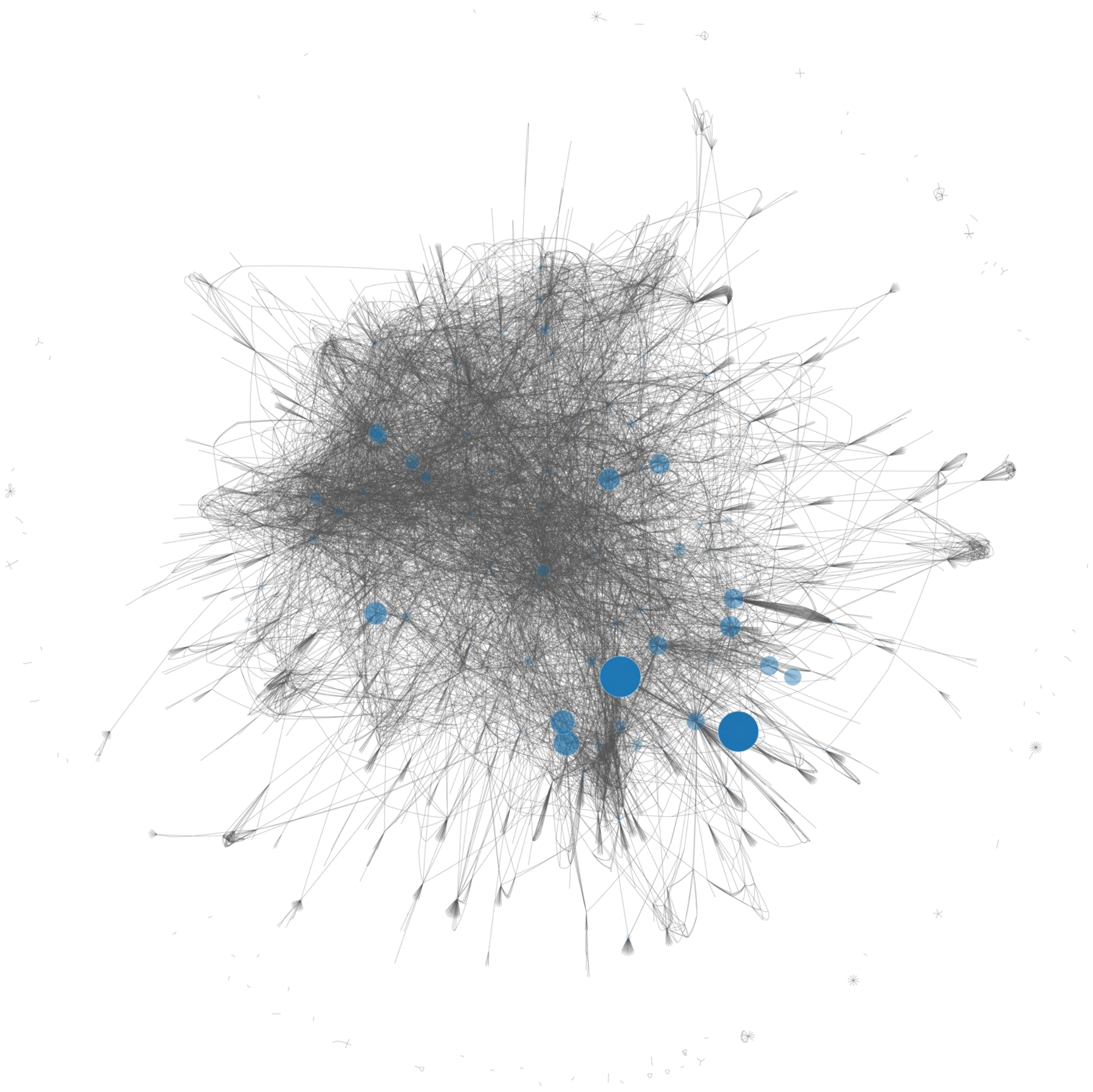


Figure 5 Visualisation of the gr0.California dataset, with page rankings shown by element size and intensity

© Queensland University of Technology | CRICOS No. 00213J