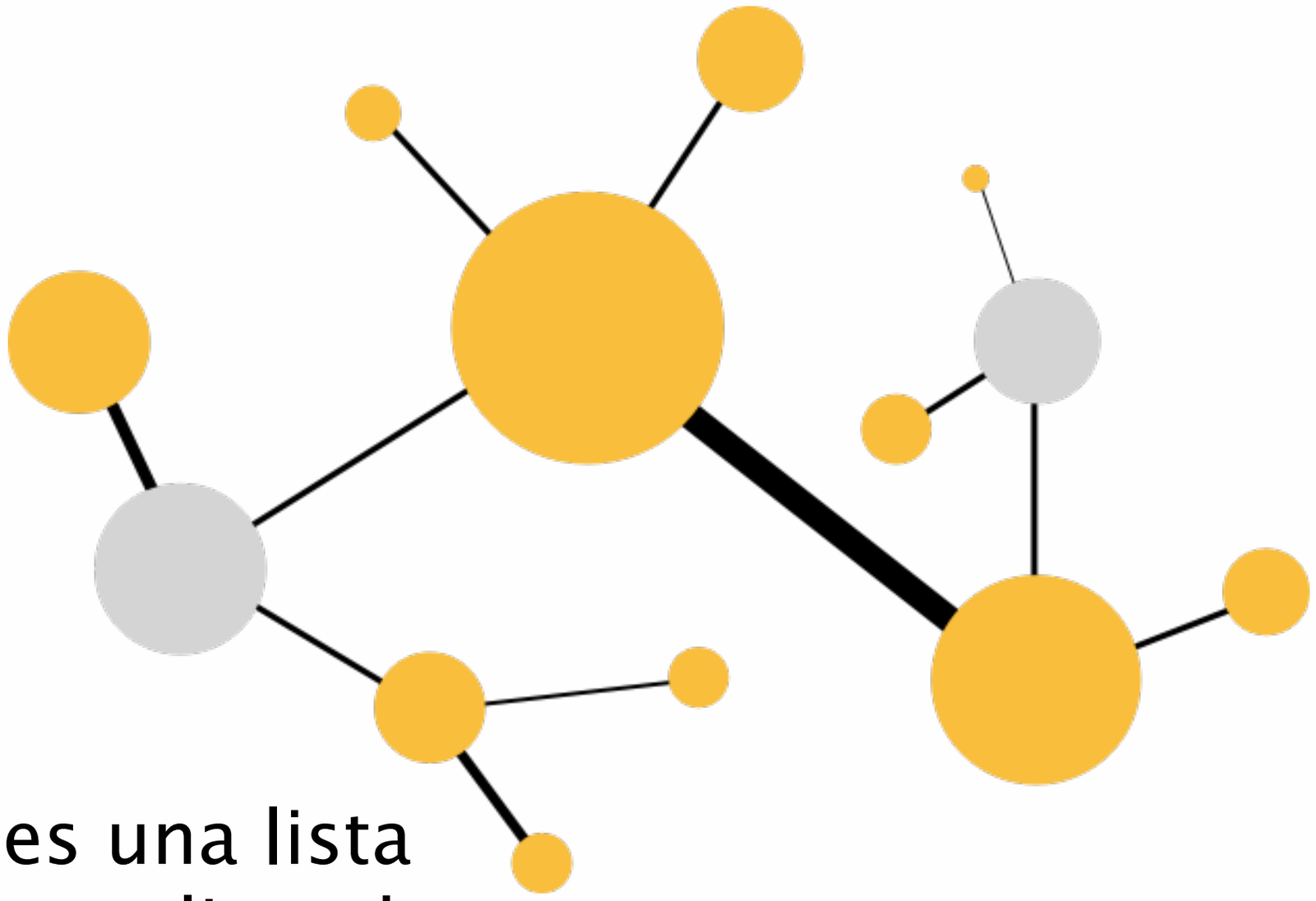
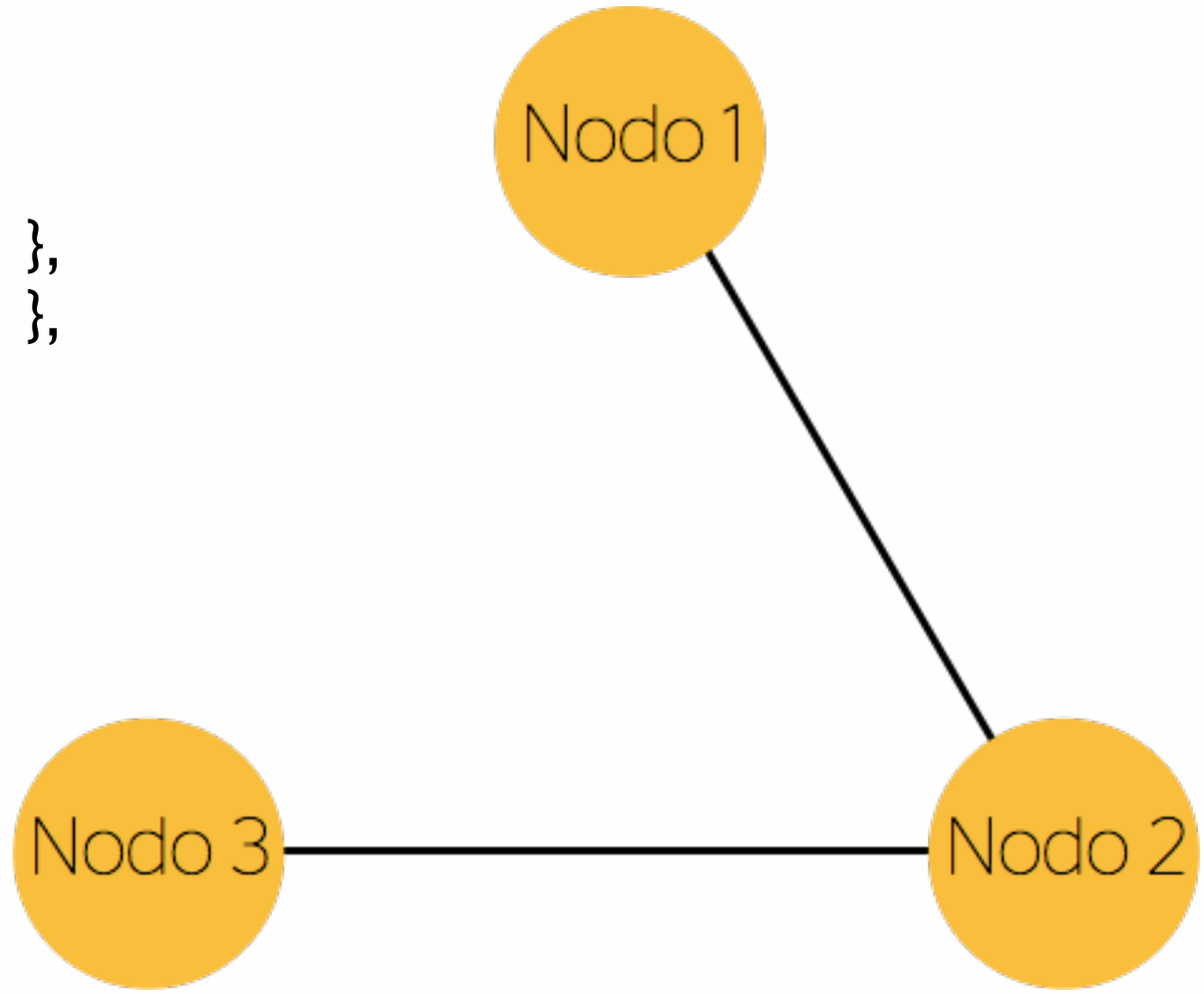


La entrada es una lista de nodos y una lista de aristas

https://github.com/mbostock/d3/wiki/Force-Layout

```
{
    “nodes”: [
        { “name”: “Nodo 1” },
        { “name”: “Nodo 2” },
        { “name”: “Nodo 3” }
    ],
    “edges”: [
        { “source”: 0, “target”: 1 },
        { “source”: 1, “target”: 2 },
    ]
}
```



Inicialización del layout:

```
var force = d3.layout.force()
.size([<ancho>, <alto>])
.charge(<fuerza a aplicar entre los nodos>)
.linkDistance(<longitud deseada de las aristas>)
.gravity(<fuerza de gravedad a utilizar>);
```

Aplicación del layout a los datos:

```
force.nodes(<lista de nodos>)
    .links(<lista de aristas>)
    .start();
```

Cada vez que se recalculan posiciones se produce un evento tick:

```
force.on("tick", function() {
    ...
})
```

Para cada nodo se calcula:

index, x, y, px, py, weight

Abrir dia4/network.html

Ejercicio

Dibujar el radio del círculo en función del número de conexiones

Dibujar el stroke-opacity de cada vértice en función del número de conexiones del nodo

Ejercicio

Dibujar en lugar de un círculo una etiqueta text con el nombre del nodo

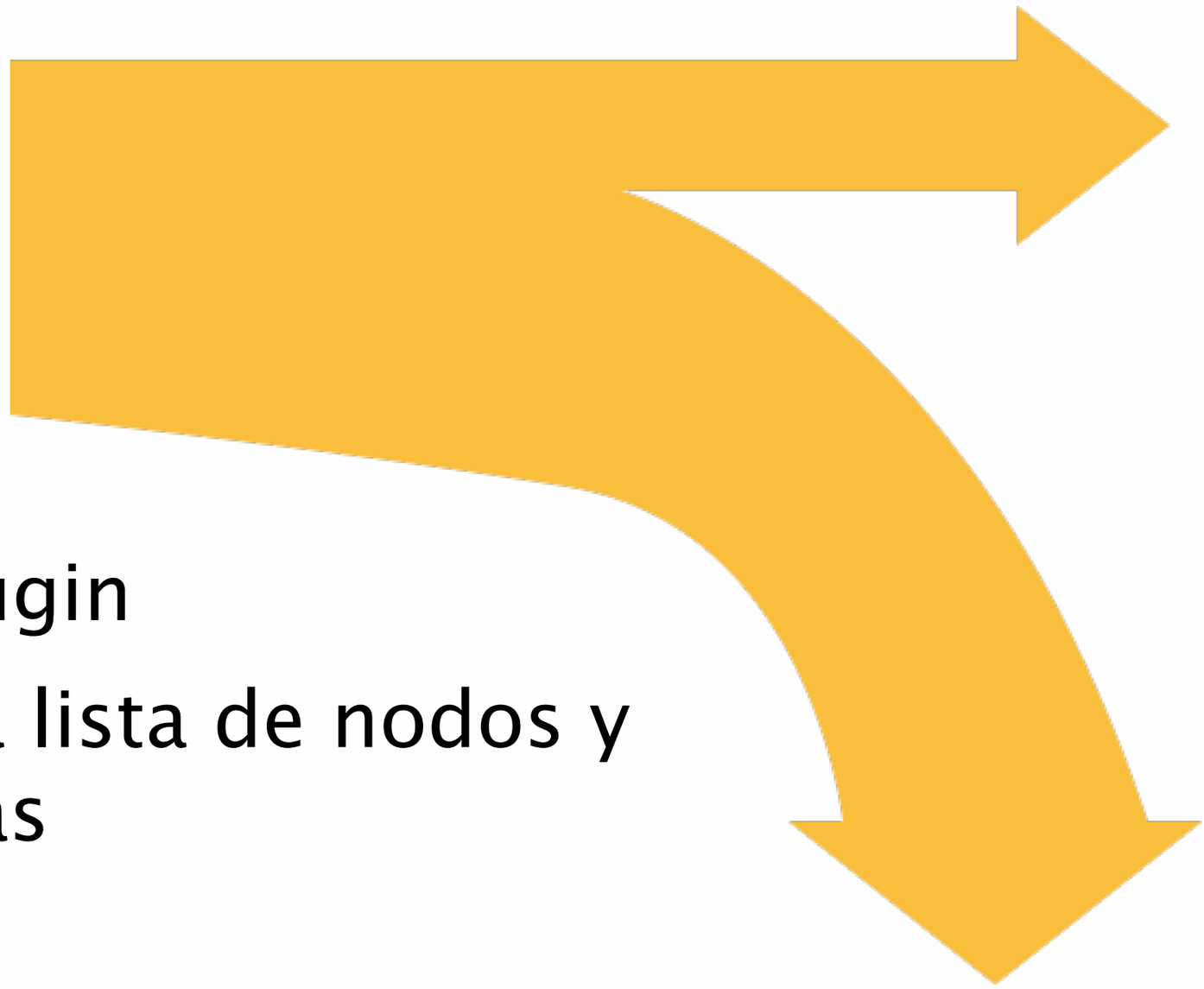
El tamaño de la fuente tiene que ser dependiente del número de conexiones del nodo

Ejercicio

Descargar de https://apps.facebook.com/netvizz/ la red de contactos de Facebook de cada uno

Visualizarla con un Force layout

En el caso de no tener cuenta en Facebook, utilizar el fichero flocker_network.json



Es necesario un plugin

- La entrada es una lista de nodos y una lista de aristas

https://github.com/d3/d3-plugins/tree/master/sankey

Inicialización del layout:

```
var sankey = d3.sankey()
.nodeWidth(<tamaño del nodo>)
.nodePadding(<padding entre los nodos>)
.size([<ancho>, <alto>]);
```

Aplicación del layout a los datos:

```
sankey.nodes(<lista de nodos>)
      .links(<lista de aristas>)
      .layout(<número de iteraciones>);
```

Para cada nodo se calcula:

x, y, dy

Ejercicio

- Añadir zoom y panning
- Resaltar las entradas y salidas de un nodo al hacer mouseover
- El tamaño de fuente es dependiente del número de conexiones

4. Layouts en D3.js

Metáfora de visualización para ver relaciones entre N entidades

Las cuerdas son elementos complejos

Inicialización del layout:

```
var chord = d3.layout.chord()
.padding(<d3.ascending o d3.descending>)
.sortSubgroups(<d3.ascending o d3.descending>)
.sortChords(<d3.ascending o d3.descending>);
```

Aplicación del layout a los datos:

```
chord.matrix(<matriz con los datos>)
```

Calcula para los datos:

groups, chords

Función para generar ticks dentro de un arco concreto

```
function groupTicks(d) {
    var k = (d.endAngle - d.startAngle) / d.value;
    return d3.range(0, d.value, 1000).map(function(v, i) {
        return {
            angle: v * k + d.startAngle,
            label: i % 5 ? null : v / 1000 + "k"
        };
    });
}
```

```
var matrix = [
        [11975,  5871, 8916, 2868],
        [ 1951, 10048, 2060, 6171],
        [ 8010, 16145, 8090, 8045],
        [ 1013,   990,  940, 6907]
      ];
```

Ejercicio

Añadir zoom y panning

Resaltar la cuerda sobre la que te pongas y los arcos que une

Cambiar la opacidad de las cuerdas según el arco donde tengamos el ratón