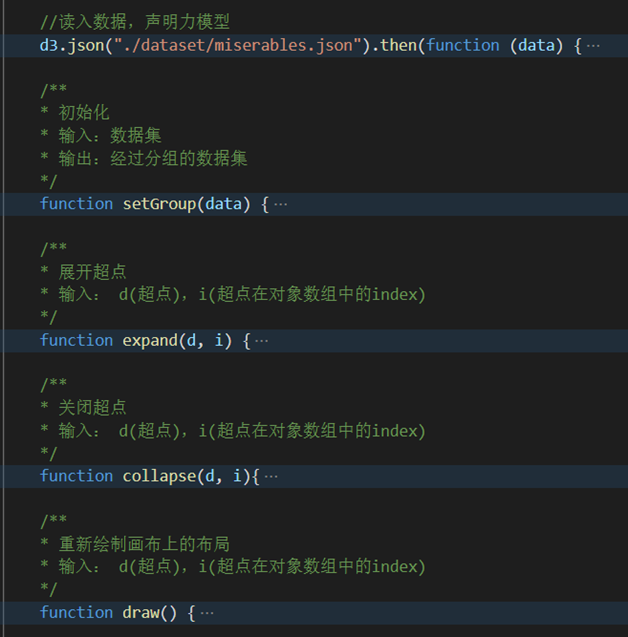
# 主体结构

包括一个主体：数据data的读入

以及四个函数：一个初始化数据函数，两个交互函数（拓展、收缩），一个布局刷新绘制函数。



## 初始化setGroup

对输入数据data进行分组，得到经过分组后的数据集groupedData，需要将初始节点信息和连边关联信息隐藏在组内（说明这个组里包含了哪些节点和连边）。

|  |  |
| --- | --- |
| 输入 |  |
| 输出： |  |

## 1.2 超点扩展操作

当用户点击一个超点时，则触发该操作。在这个过程中，需要通过一系列的计算判断此时有哪些节点和连边停留在画布上。

|  |  |  |
| --- | --- | --- |
| 触发前 |  |  |
| 触发后 |  |  |

## 1.3 节点集收缩操作

当用户点击一个非超点节点时，则触发该操作。在这个过程中，需要通过一系列的计算判断此时有哪些节点和连边停留在画布上。

## 1.4 布局刷新函数

布局刷新函数：在原来的布局基础上继续迭代，直到达到稳定。

1. 每次刷新需要重新绑定数据，否则会出错。
2. 对连边集进行更新
3. 对节点集进行更新，已知d3特性中存在update, enter, exit, 所以与之前的数据集相比存在两种情况：
   1. 当原始节点数<=当前节点数，存在update和enter两个部分，update的节点个数=页面上的circle元素个数，需要进行样式上的更新。enter部分属于向页面添加的circle元素个数。
   2. 当原始节点数>当前节点数，存在update和exit两个部分，update的节点个数=当前节点个数（两个节点集的交集），需要进行样式上的更新。exit部分属于两个节点集的非交集部分，需要将这些元素其移出页面。

# 二、函数逻辑

## 2.1 主体逻辑

1. 声明全局变量

2. 读入数据：将每条连边的source和target设置为节点对象本身

3. 为画布添加<g>节点，分别为节点<g>，连边<g>，便于后期对页面元素进行管理

4. 对数据进行初始化

5. 声明力模型 d3-force-simulation

## 2.2 初始化

1. 遍历所有节点，如果该节点所属的组还未被添加到组节点集中，则新建一个组，将该节点加入该组。如果该节点所属的组以及被添加到组节点集中，则将该加入该组，并计节点的size+1；

2. 遍历所有连边，判断节点两端所属组{group1, group2}，若该连边还未被添加到组连边集中，则新建一条边，如果已经存在，则连边的size++；

## 2.3 超点扩展

2.1.1 节点扩展：根据用户所点击的超点可知组内包含哪些节点，需要遍历这些节点，将其加入总节点集中。

1. 遍历组内节点，依次加入总结点集。
2. 将用户点击的这个超点移出总节点集。

2.1.2 连边扩展：根据用户所点击的超点可知组内包含哪些连边。需要遍历这个边集，将其加入总连边集中。

1. 在遍历边集的过程中，需要考虑以下几种情况：
   1. 连边e的两端均在本组内
   2. 连边e的一端在本组内，但另一端在一个已展开的组内，也就是与独立的节点相连
   3. 连边e的一端在本组内，但另一端在一个未展开的组内

其中情况a)和情况b)可以合并，因为当用户点击本组超点时，该超点已被判定为展开。所以需要判断连边另一端所属的组是否展开。由于无法直接知晓超点的连边集中连边两端（源节点、目标节点）哪一个属于本组，所以在每次遍历的时候需要预先判断。

1. 判断连边两端是源节点还是目标节点属于本组节点，每次遍历的时候新建一个对象slink，确保将连边slink的一端为e中属于本组内节点的那个节点，对另一端进行处理。
2. 针对情况a-b)，设置slink的另一端为连边e的另一端。（不进行处理，直接将连边加入总数据集）
3. 针对情况c)，设置slink的另一端为某个未展开的组节点。!注！：这里可能会出现一个问题，如下图所示，当节点1分别与节点4和节点5相连，存在两条边。但展开后只需要向节点集加入一条连边{node1, group2, size=2}。

图示

描述已自动生成

## 2.4 节点集收缩

有两种思路：思路1->类似于超点扩展，想办法把相关节点和连边从总节点集和总连边集中删除。思路2->新建一个节点和连边数组，从0开始，判断有哪些节点和连边相关，加入新数组，然后赋值给总节点集和总连边集。（目前采取的是这个思路，但是思路1不需要拷贝，效率可能会高一些）

2.4.1 节点收缩：根据用户所点击的节点可知该节点属于哪个组，将属于该组的节点都移出总节点集，然后再向节点集中添加组节点。（目前采取的方法还是遍历总节点集，将与当前组无关的节点都加入到临时的newNodes数组中）

2.4.2 连边收缩：根据用户所点击的超点可知组名，遍历所有连边，判断连边两端是否有节点在组内，如果没有，直接添加到新连边集中，如果没有：

1. 判断连边两端是源节点还是目标节点属于用户点击的这个组，设置start为本组节点，end为另一端节点。

1.1 当end展开时，情况a的连边直接丢掉，情况b需要统计，需要计算group1里有多少个节点与node3相连，相加后，最后加入的是{group1, node3，size=2}。利用map，如果没有就直接加入，如果相同的连边已经存在了，就直接对连边size++；

1.2 当end未展开时，此时需要添加{group1, group2, size=x}，这个连边及其size可以直接从初始化时保留的最原始的数据中获取，并且只添加一次。

图示

描述已自动生成

## 2.5 布局刷新函数

布局刷新函数：在原来的布局基础上继续迭代，直到达到稳定。

1. 每次刷新需要重新绑定数据，否则会出错。
2. 对连边集进行更新
3. 对节点集进行更新，已知d3特性中存在update, enter, exit, 所以与之前的数据集相比存在两种情况：
   1. 当原始节点数<=当前节点数，存在update和enter两个部分，update的节点个数=页面上的circle元素个数，需要进行样式上的更新。enter部分属于向页面添加的circle元素个数。
   2. 当原始节点数>当前节点数，存在update和exit两个部分，update的节点个数=当前节点个数（两个节点集的交集），需要进行样式上的更新。exit部分属于两个节点集的非交集部分，需要将这些元素其移出页面。

交互绑定：为每个节点绑定一个交互函数，当用户进行点击，先判断该节点是否展开，如果已展开则触发collapse()，如果未展开则触发expand()。