图可视化工具(GraphVisualizer)说明文档

清华大学软件学院 软件41班 李肇阳 2014013432

程序设计实训 2015.8

**一、开发环境**

* OS
  + Windows 7 Ultimate 64 Bit
* IDE
  + Visual Studio 2012 Premium (MSVC++ 11.0)
  + Qt5 Visual Studio Add-in 1.2.3
* External Libraries
  + Qt 5.3 32-bit for Desktop (MSVC 2012 OpenGL) [GPL]
  + OGDF v. 2015.05 [GPL]

Bin目录下是可执行文件。附有所需的MSVC11.0运行库的安装包，以及Qt所需的一系列dll文件。已用虚拟机测试，在新安装的纯净Win10系统中能够正常运行。

由于采用标准C++语法、采用的图形框架Qt是跨平台的、采用的外部库OGDF是C++的且是开源的，本软件理论上可跨平台编译运行。

程序执行时默认从**工作目录下**的PaperConferenceAuthorGraph、TopicGraph子目录内读取输入数据。

**二、要求完成情况**

以下是已完成的项目，依大作业要求和评分标准列出。

基本要求：

✓使用QT实现2个数据集的不同数据类型、不同Layout方法的绘图，能够在不同方法数据间切换

✓能够展示节点和边的所有属性

✓能够对视图进行放大、缩小*（可使用鼠标滚轮）*

进阶要求：

✓可以用鼠标拖动节点

✓不同Layout之间用动画进行变换

✓鼠标在移到可以点击的部分时变成手的形状

✓可以针对点的属性*（连通度）*进行过滤

✓可以利用拉索选择一部分点进行高亮，并且拖动这一些点*（鼠标右键进入多选模式）*

其他交互：

✓自定义视图的几何样式：半径、线宽、字号、配色主题

✓为悬停和选中的节点提供视觉提示

✓点击节点后高亮由其相邻节点构成的子图

✓鼠标拖拽可进行视图平移

✓鼠标滚轮可以当前指针位置为中心进行视图缩放

自己写的Layout算法：

✓Circular*（简单匀布在圆周上。这是一个平凡的算法，库里本来就有）*

✓Circular x2*（选取一部分点放在中间的小圆圈内）*

✓Grid*（简单平铺在平面格点上。这是一个平凡的算法，库里本来就有）*

✓LZYLayout*（所有点分布成LZY字样。这是一个娱乐性的算法）*

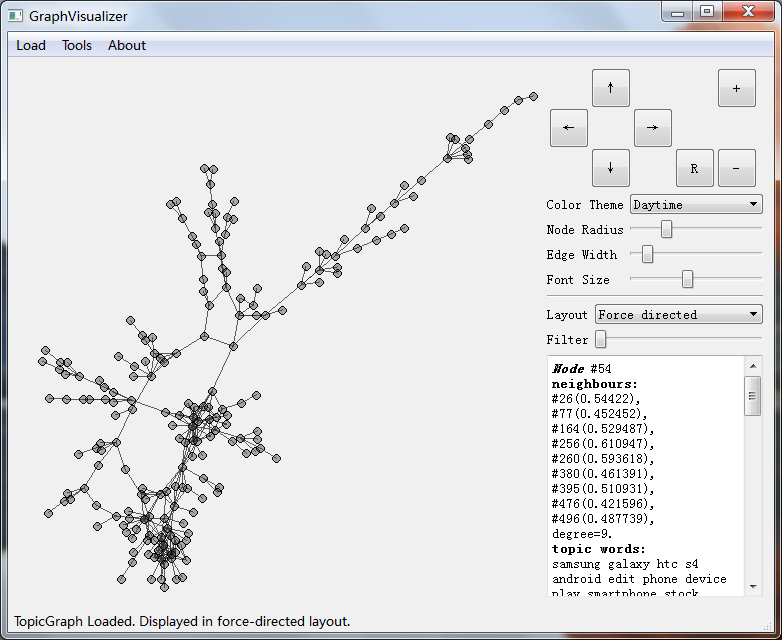
水平实在有限，自己写的布局算法都比较水。感觉亮点在于细腻流畅的前端交互。

除使用右侧面板中的按钮、滑动条、下拉列表框之外，主要的交互都可以由各种鼠标动作（左键、右键、滚轮，悬停、点击、拖拽）完成。

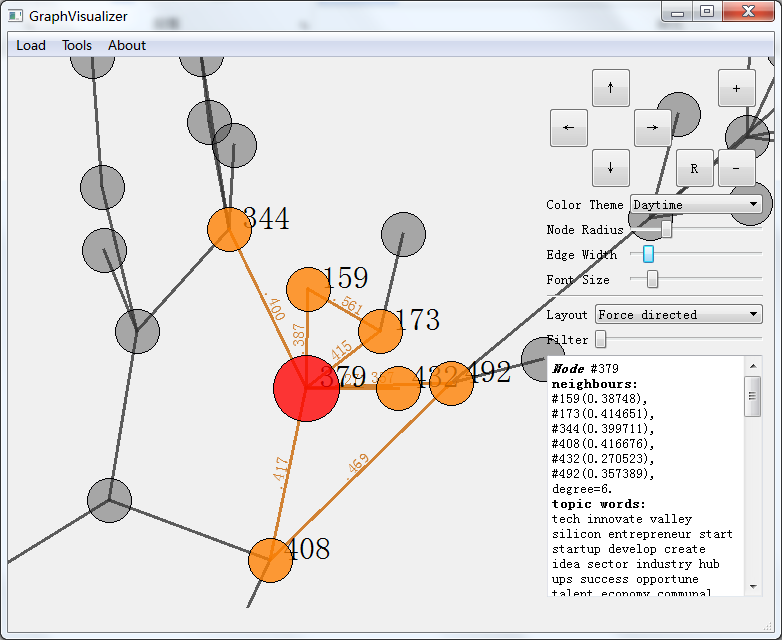
工程总代码量不大（1.6k行，不计generated files），因为用了些面向对象的技巧精简节约代码量。

数据结构和逻辑上做了精心的优化。据我自己测试，程序占用内存的峰值不超过12MB，交互和动画毫无卡顿感。

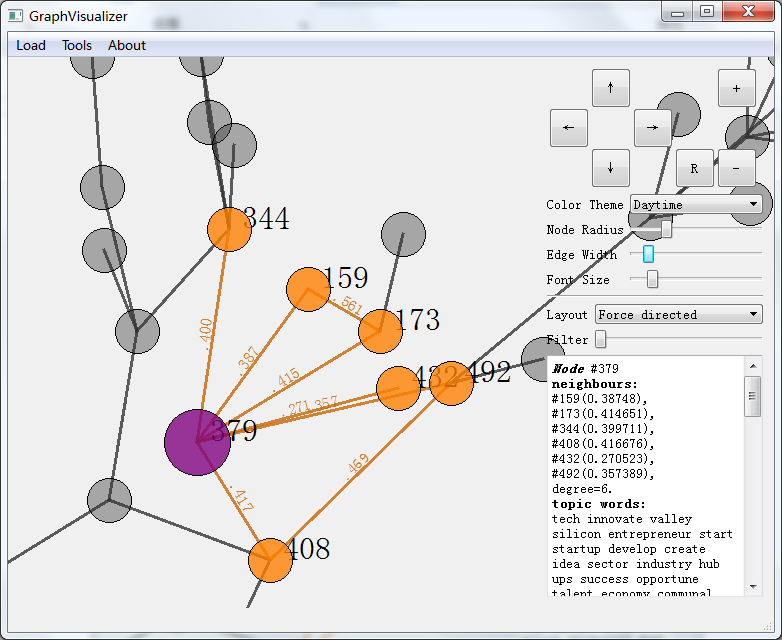
**三、简要操作说明和运行时截图**



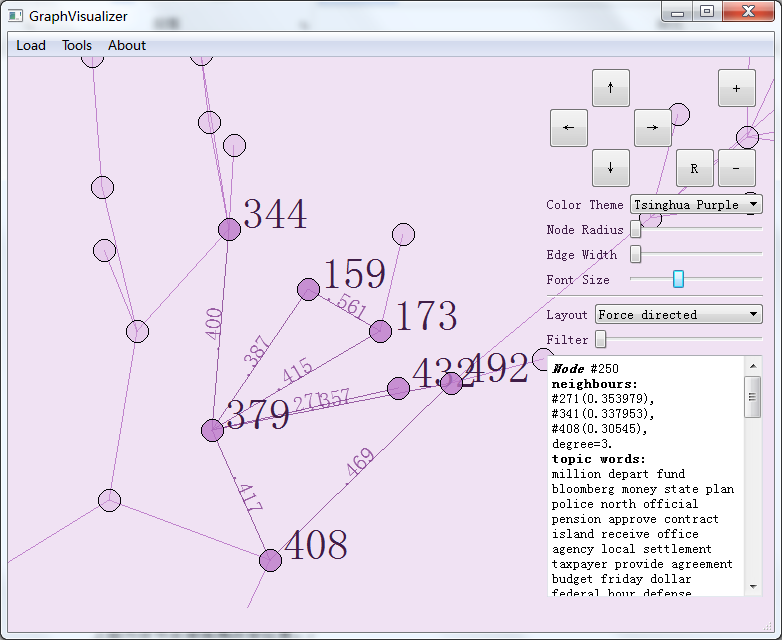
上图为点击菜单项“Load>TopicGraph”后从文件加载TopicGraph后默认展示Force-directed布局。



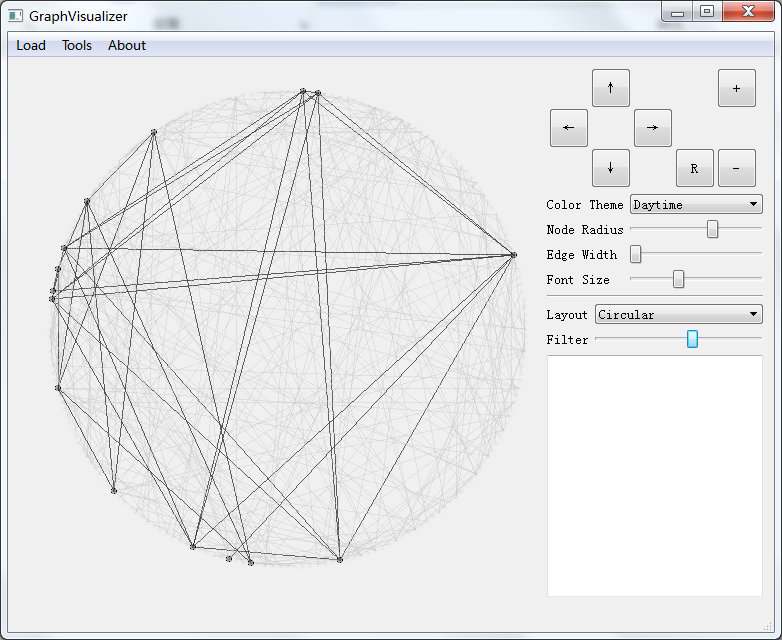
上图为放大到局部（可用鼠标滚轮和拖拽，也可用右上角导航按钮）。鼠标悬停于节点上时节点会变色，同时指针变为手形，右侧面板下方会分字段显示该节点详细信息；鼠标按下时节点会再次变色，为用户提供充足的视觉提示。点击一个节点后高亮出其所有相邻节点，显示出相关边的信息（权重）。



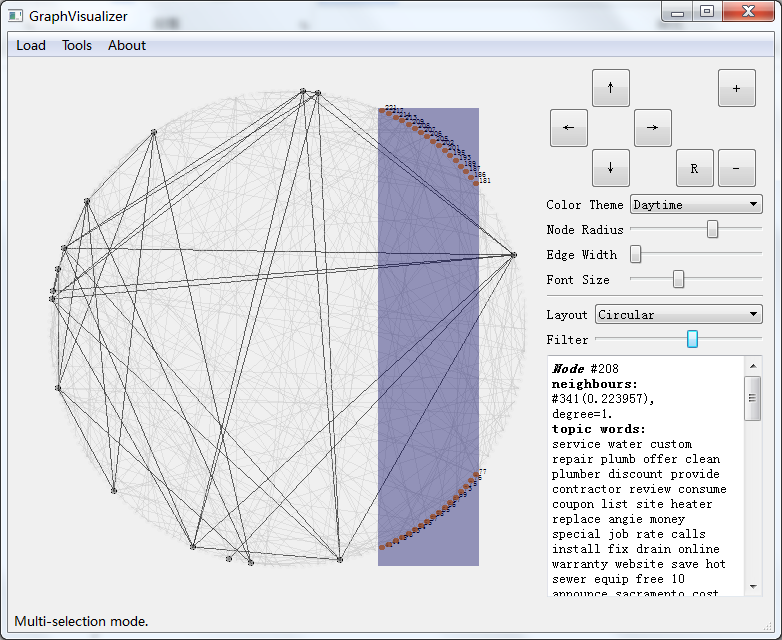
使用鼠标拖拽该节点可改变其位置。与其相邻的边亦实时更新。



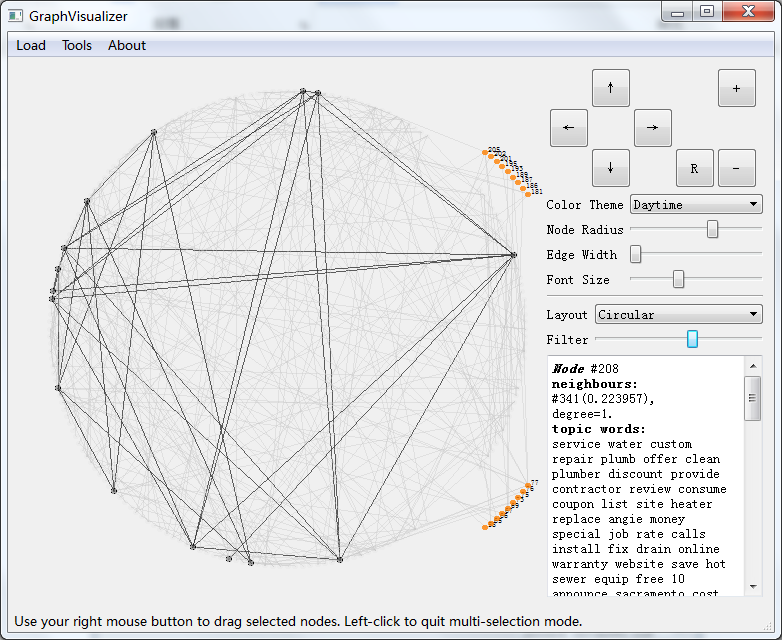
用右侧面板可更改几何样式（半径、线宽、字号、配色）。图为半径最小、线宽最小、字号默认、清华紫配色。



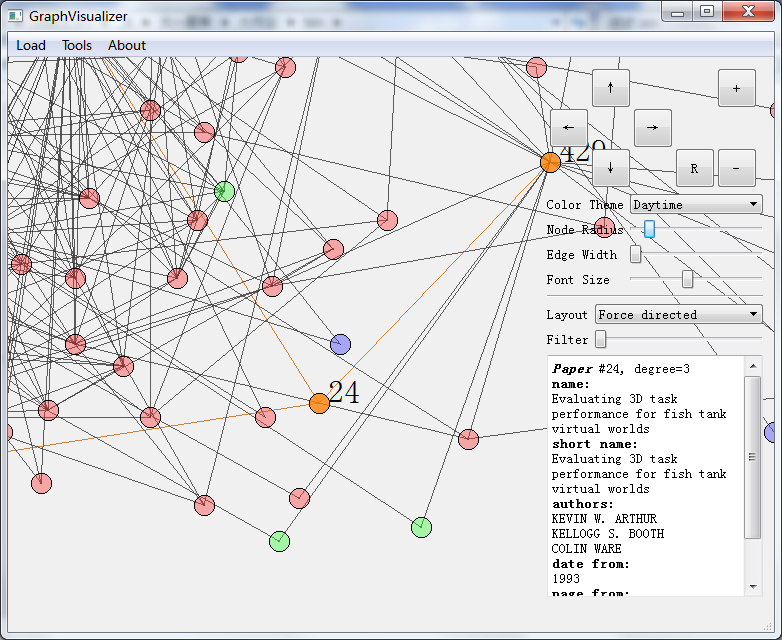
用右侧面板上的Layout下拉列表框可切换布局方案。切换时有动画过渡。图为Circular布局。用右侧Filter滑块可按节点的连通度进行过滤，被过滤的点和边变为浅色。



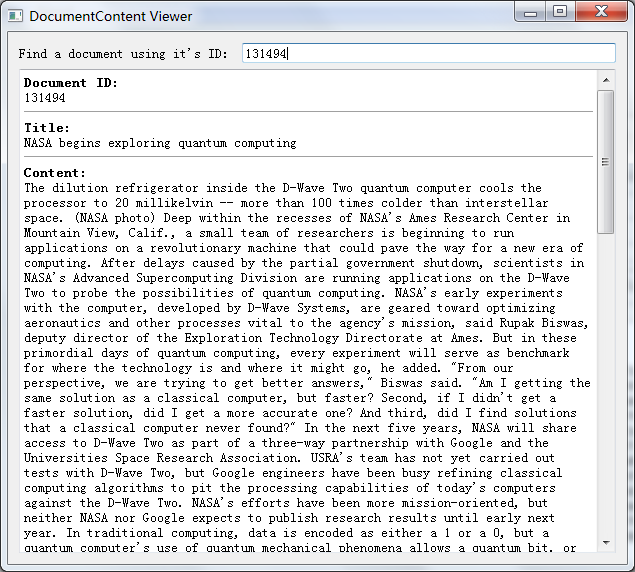
矩形选框模式，可多选节点。按住鼠标右键画出矩形即可多选。下方状态栏提供操作指引。



用矩形框选中多个节点后，用鼠标右键拖拽，可改变它们的位置。



另一数据集合。图为从文件加载PaperAuthorConferenceGraph后，不同类型的点以不同颜色显示。



Tools菜单下的文档内容查看器。输入ID可快速查看文档内容。

**四、实现说明**

1. 数据的存储

1.1类的结构：完全按照大作业要求。

* 两种类型的图（PaperConferenceAuthorGraph和TopicGraph）都继承自同一个Graph基类；
* 共四种类型的点（PaperNode, ConferenceNode, AuthorNode, TopicNode)都继承自同一个Node基类；
* 共两种类型的边（DirectedEdge，UndirectedEdge）都继承自同一个Edge基类。
* 点类中存有布局信息（坐标）。

1.2文件读取

* 由于输入文件格式工整，直接一行一行读。
* 图类的void importEdgesFromTXT(string fileName)，void importNodesFromTXT(string fileName)两个成员函数用于读入顶点集、边集。
* 对于DocumentContent.txt，考虑到该文件较大、不宜全部读入内存，我使用std::map<>建立了“文章id->该文章在文件中的位置”的映射（用到std:: ifstream 的ignore(), tellg(), seekg() 等函数进行随机读取），随用随读。

1.3点和边的关联

* 顶点集、边集分别是图类的成员（std::vector<>）。节点id的统一性保证可根据节点寻找其邻居、根据边寻找其端点。为提高效率，在每个边中增设Node\*成员，保存指向其端点的指针，以在绘制边时迅速得到端点坐标。在调用两个importFrom()之后调用bind()成员函数可维护上述结构。
* 对于PaperConferenceAuthorGraph，由于其顶点分为三种，建立了三个成员vector<PaperNode> papers; vector<ConferenceNode> conferences; vector<AuthorNode> authors;。考虑到每当需要遍历所有节点时都需要些三个for(){}循环，代码量巨大，另建立了vector<Node\*> nodes;成员，统一保存所有节点的指针（基类型）。基类型有type字段，需要时据其强制转换为派生类指针以读取派生类特有字段。调用makeUnionNodeVector()成员函数维护上述结构。
* calcDegree()成员函数据顶点集、边集计算各顶点的连通度并存入内存，同时返回图中最大的连通度，用以设置界面上degree filter滑块的范围。
* 以上列举的图类的诸成员函数存在相互依赖关系，调用时须明确先后。

2.布局算法

* 使用了OGDF提供的1种力导向算法，自己写了4种布局算法。调用图类的doLayout()函数进行布局计算。各种布局结果均直接存入每个节点。考虑到计算布局耗时较长，为避免用户在切换布局时的等待，本程序在载入文件后立即计算全部5种布局并保存在内存中。

3.图形绘制和交互

* 主窗口class GraphVisualizer继承QMainWindow。成员变量除图外，主要是绘图和交互所需的各种参数，如图形的几何参数、用于视图平移缩放的坐标转换的参数、颜色值、鼠标的坐标、与鼠标产生交互的节点的id或下标、动画的开关和相位等。成员函数主要是处理各种用户事件的槽函数。
* 绘图纯QPainter，没有用QGraphicsScene。除沿边的方向斜着打印文字外，没有用到QPainter::setTransform，QPainter::translate。视图的平移缩放等全部是自己计算的，自己编写的，自己维护参数。
* 鼠标交互全都靠处理mouseMoveEvent，mousePressEvent，mouseReleaseEvent，wheelEvent事件。为此在窗口类中设立了不少布尔型变量，维护这些flags以保证悬停、点击、拖拽等各种复杂鼠标事件的正确响应。遍历节点、寻找与鼠标动作产生交互的节点的操作是在处理鼠标事件时进行的，更新相关数据结构（当前与鼠标产生交互的节点的id或下标或指针（单个或集合），节点显示状态（高亮与否）的flag）以供绘制。
* 动画是QTimer逐帧播放的。为播放动画，每个节点增设成员变量记录动画的终点。利用差分方程根据当前坐标、终点坐标和当前的相位计算下一帧时的坐标。

**五、反思和感言**

见ppt

（完）