3.基于微观居民数据合成的高性能疫情传播仿真平台

3.1用户用例描述

构建人口相关

|  |  |
| --- | --- |
| **用例名称** | **初始化人口合成模型** |
| 前置条件 |  |
| 基本事件流 | 1. 用户在数据上传界面，选择本地文件上传； 2. 系统将数据文件发送至后端； 3. 系统后端运行贝叶斯网络训练算法，生成相应的贝叶斯网络模型； 4. 模型训练完成后，系统将初始化成功的信息反馈到前端界面 |
| 分支事件流 | 1. 若用户所选择的文件路径无效，反馈错误信息； 2. 若用户所上传的数据格式不符合规范，反馈错误信息； 3. 若数据库中已存在模型，询问用户是否替换模型 |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **更新人口合成模型** |
| 前置条件 | 模型已初始化 |
| 基本事件流 | 1. 用户在数据上传界面，选择上传本地数据文件（个体属性数据\个体-旅行调查数据\行程数据） 2. 系统将数据文件发送至后端 3. 系统后端利用新数据更新数据库中的模型 4. 模型更新完成后，系统将更新成功的信息反馈到前端 |
| 分支事件流 | 1.若用户所选择的文件路径无效，反馈错误信息；  2.若用户所上传的数据格式不符合规范，反馈错误信息；  3.若模型未初始化，反馈错误信息 |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **生成目标数量的城市人口** |
| 前置条件 | 模型已初始化 |
| 基本事件流 | 1. 在数据合成界面，用户点击“合成人口”按钮； 2. 系统前端弹出界面，要求用户输入目标数量N； 3. 用户输入目标数量N，点击“确认”按钮； 4. 系统将N发送至后端，利用已存在的模型合成目标数量的人口数据； 5. 系统前端提示合成完毕信息，并跳转至传染模型编辑界面 |
| 分支事件流 | 1. 模型尚未初始化，反馈错误信息； 2. 用户输入的N是无效值，反馈错误信息 |
|  |  |

构建传染模型相关

|  |  |
| --- | --- |
| **用例名称** | **编辑传染模型** |
| 前置条件 | 已生成目标城市人口 |
| 基本事件流 | 1. 系统前端提示操作说明； 2. 用户阅读说明，并点击“确认”按钮； 3. 系统前端隐藏提示框，显示编辑界面 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **新建结点** |
| 前置条件 | 用户处于编辑界面，并阅读操作说明 |
| 基本事件流 | 1. 用户点击“新建结点”按钮； 2. 系统前端显示结点图标，后端增加相应的数据； 3. 用户拖拽图标到指定位置； |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **删除结点** |
| 前置条件 | 结点已存在 |
| 基本事件流 | 1. 用户选中某结点； 2. 前端在所选中的结点附近弹出选项框； 3. 用户在选择框内选择“删除”； 4. 系统前端删除结点图标，后端删除相应的数据 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **编辑结点** |
| 前置条件 | 结点已存在 |
| 基本事件流 | 1用户选中某结点；  2前端在所选中的结点附近弹出选项框；  3用户在选择框内选择“编辑”，并编辑结点属性；  5.系统前端更改结点属性，后端更新相应结点的属性值 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **增加边** |
| 前置条件 | 至少已存在两个结点 |
| 基本事件流 | 1. 用户依次选中两个结点； 2. 系统前端弹出“增加边”； 3. 用户点击“增加边”； 4. 系统前端弹出权重编辑框； 5. 用户在编辑框内输入各权重值 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **删除边** |
| 前置条件 | 已存在一条边 |
| 基本事件流 | 1. 用户选中一条边； 2. 系统弹出选择框； 3. 用户在选择框内选择“删除”； 4. 系统前端删除所选边的图标，后端删除相应数据； |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **编辑边** |
| 前置条件 | 已存在一条边 |
| 基本事件流 | 1.用户选中一条边；  2.系统弹出选择框；  3.用户在选择框内选择“编辑”，并编辑权重；  4.系统前端更新所选边的权重，后端更新相应权重值； |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **确认完成模型编辑** |
| 前置条件 | 用户已完成模型的编辑 |
| 基本事件流 | 1. 用户点击“完成编辑”按钮； 2. 系统检查模型，并跳转至城市空间信息构建页面 |
| 分支事件流 | 1.用户所编辑的模型不符合规范，提示错误信息 |
|  |  |

构建城市空间场景相关

|  |  |
| --- | --- |
| **用例名称** | **提交城市空间图层的url（GeoServer）** |
| 前置条件 | 传染模型编辑完成 |
| 基本事件流 | 1. 用户按照系统前端的提示，依次输入各目标图层的url； 2. 系统将url发送至后端，后端调用WFS接口从GeoServer服务器获取shapefiles数据，并将数据组织为仿真所需的形式； 3. 数据处理完成后，系统前端跳转至仿真界面 |
| 分支事件流 | 1.用户所提供的url无效，或所提供的数据不符合要求，反馈错误信息 |
|  |  |

仿真相关

|  |  |
| --- | --- |
| **用例名称** | **启动仿真** |
| 前置条件 | 城市空间信息构建完成 |
| 基本事件流 | 1. 用户在仿真界面，点击“启动仿真”按钮 2. 系统将“启动仿真”的消息发送至后端，后端开始运行仿真逻辑； 3. 仿真过程中，后端将所计算出的数据发送至前端； 4. 前端以热力图、曲线图的形式对数据进行实时可视化 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **暂停仿真** |
| 前置条件 | 仿真已启动 |
| 基本事件流 | 1. 用户在仿真界面，点击“暂停”按钮； 2. 系统将“暂停”消息发送至后端； 3. 后端保存仿真状态信息，暂停仿真进度 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **开启医疗资源配置模式** |
| 前置条件 | 仿真已启动 |
| 基本事件流 | 1. 用户点击“医疗资源配置模式”； 2. 系统设置状态，将消息发送至后端，后端暂停仿真进程 |
| 分支事件流 | 1.仿真未启动，提示错误信息 |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **增加隔离点（医疗资源）** |
| 前置条件 | “医疗资源配置模式”已开始，且仿真处于暂停状态 |
| 基本事件流 | 1. 用户根据可视化的城市空间图层，在图层上点击某些位置，放置隔离点； 2. 系统在相应位置显示图标，并弹出输入框，要求用户输入隔离点的属性值； 3. 用户输入隔离点的属性值； 4. 系统前端更新相应数据，维护相关数据结构 |
| 分支事件流 | 1.用户所输入的属性值无效，提示错误信息 |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **撤销隔离点（医疗资源）** |
| 前置条件 | 图层上已存在隔离点 |
| 基本事件流 | 1. 用户选中某隔离点； 2. 系统前端在图标附近弹出选项框； 3. 用户在选项框内选择“撤销”； 4. 系统前端更新相应数据，维护相关数据结构 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **完成医疗资源配置** |
| 前置条件 |  |
| 基本事件流 | 1. 用户点击”编辑完成“按钮； 2. 系统将前端所维护的相关数据结构发送至后端，后端更新相应数据结构，重启仿真进程 |
| 分支事件流 |  |
|  |  |

|  |  |
| --- | --- |
| **用例名称** | **更改感染传播模型** |
| 前置条件 | 仿真处于暂停状态 |
| 基本事件流 | 1. 用户在仿真界面点击“更改传染模型”; 2. 系统将消息发送至后端，后端暂停仿真进度，前端跳转至传染模型编辑界面 |
| 分支事件流 |  |
|  |  |

3.2系统实现方案

3.2.1前端（OpenLayers, Web）

为实现轻量级访问，系统以Web端作为用户访问与交互的前端，来自GIS平台的城市空间数据（例如建筑、交通道路、用地等）和系统后端的计算结果（例如热力图、曲线图等）将在Web进行可视化展示；

系统主要借助 OpenLayers进行前端可视化开发。OpenLayers是一个用于WebGIS前端开发的JavaScript库, 为GIS数据的可视化、地图交互编辑、GIS数据获取提供了一系列API。

1. GIS数据可视化

OpenLayers采用层级化结构对GIS数据进行可视化：Feature->Layer->Map；

Feature指定每个地理图元的几何和属性；

Layer是由Feature组成的集合，支持用户指定Feature的显示样式；

Map是由Layer构成的集合，允许多个Layer叠加显示；

OpenLayers允许用户以GeoJSON格式数据构建Layer，并指定数据所采用的参考坐标系

1. 高性能渲染

对于GeoJson格式的数据，OpenLayers支持三种渲染方式：图像渲染、矢量渲染、Tile渲染。

图像渲染将Layer作为一张整体图像进行显示，优点是渲染速度快，缺点是放大时细节丢失严重；

矢量渲染将含有Features的Layer作为一张矢量图进行渲染，根据Layer当前的缩放等级，实时计算出一张图像，优点是缩放时保持显示精度，缺点是渲染速度慢；

Tile渲染预先计算出多个缩放等级下的网格化图像，在用户操作过程中，根据Layer当前的缩放等级和视窗中心位置确定所显示的图像，该渲染方式结合了图像渲染和适量渲染的优点，兼顾了高精度和高性能。

本系统中，对Feature数量大的Layer采用Tile渲染方式（如建筑分布、人口密度分布），其余Layer则采用矢量渲染

1. 交互编辑

Openlayers支持监听用户输入事件，并允许自定义相应的处理函数；

在本系统中，用户的基本输入事件（如地图缩放、拖拽等）、图层编辑（如设置隔离点）、图元属性查询（如区域感染人数）等功能均借助OpenLayers提供的API实现。

1. GIS数据获取

OpenLayers支持开发者以url路由的方式获取第三方地图底图

OpenLayers 支持Open GIS 协会制定的WMS（Web Mapping Service）和WFS（Web Feature Service）等网络服务规范，允许开发者通过[远程服务](https://baike.baidu.com/item/%E8%BF%9C%E7%A8%8B%E6%9C%8D%E5%8A%A1" \t "https://baike.baidu.com/item/openlayers/_blank)的方式，将以OGC 服务形式发布的地图数据加载到Web前端。

3.2.2后端（Flask，GeoServer，其他GIS平台）

本系统中，后端负责执行GIS数据的获取、数据合成算法和仿真逻辑，其中涉及到一些第三方接口和工具库（如与GeoServer交互的接口、CUDA计算环境），因此采用Python作为后端开发语言，借助Python的包管理模式方便地实现系统集成。

后端功能由三个主要模块构成：

1. 常规逻辑执行

负责执行常规逻辑，如数据读取、数据组织、仿真流程控制；

1. 并行化仿真

借助numba库提供的CUDA编程模型，实现大量Agent行为的并行化仿真；

numba所提供的CUDA编程模型允许开发者以python语言编写CUDA核函数，在程序运行时被编译为GPU代码。

1. 与GeoServer之间的数据访问接口

**GeoServer简介**

GeoServer 是 OpenGIS Web 服务器规范的J2EE实现，以开源的形式发布，用户可以将GeoServer部署在指定服务器上，为GIS应用提供持久化服务。GeoServer可提供以下主要服务：

**数据发布**

GeoServer允许用户以多种来源在服务器上发布GIS数据，如本地shapefiles文件、PostGIS数据库等；

单个数据被发布后将作为一个图层，每个图层拥有唯一的url可供访问；

**数据组织**

GeoServer允许用户以层次化的结构组织数据，即图层->图层组->工作区，每个工作区拥有唯一的url可供访问。在本项目中，要求用户将所需的各图层组织在同一个工作区内。

**数据查询**

GeoServer对基本的数据查询功能提供了高性能支持，例如查询所选择区域内的图元及属性。

**数据访问**

GeoServer提供了面向python的WMS和WFS访问接口，允许开发者通过接口从服务器下载特定的GIS数据（如shapefile）。本系统需要用户提供GIS数据的url，系统后端将调用WFS接口从GeoServer所部署服务器上获取shapefiles数据，用于构建城市空间场景。

3.2项目部署方案

（Flask, Ngnix, Docker镜像）