程序初始计划：

这个项目是记录地铁线路并且加以计算，所以储存线路是其中的关键部分。

经过慎重的思考以及选择，最后决定采用链表的形式储存整个地图，因为地图的站点分布不均匀，采用邻接矩阵很不合适无法对齐，这样的方式也比邻接链表要方便。

最终的储存结构中，每个站点都有一个单独的station对象，并且储存了相关的信息，站与站之间按照线路排列，每一条线路的起点被专门放在一个数组里面方便访问。

文件说明：

保存的文件（beijing-subway.txt）里面都是中文，按照一行线路名一行该线路下所有站点名的方式排列，将所有的线路中的站点都这样排列在其中。如果是环线，则在线路行最后加上”CIRCLE”标识，如果是单行线，则在对应位置加上”ONEWAY”来标识。地铁路线名已经给好（请勿随便修改，线路名称以及数量不能变动 这样才能有效识别）

其他：

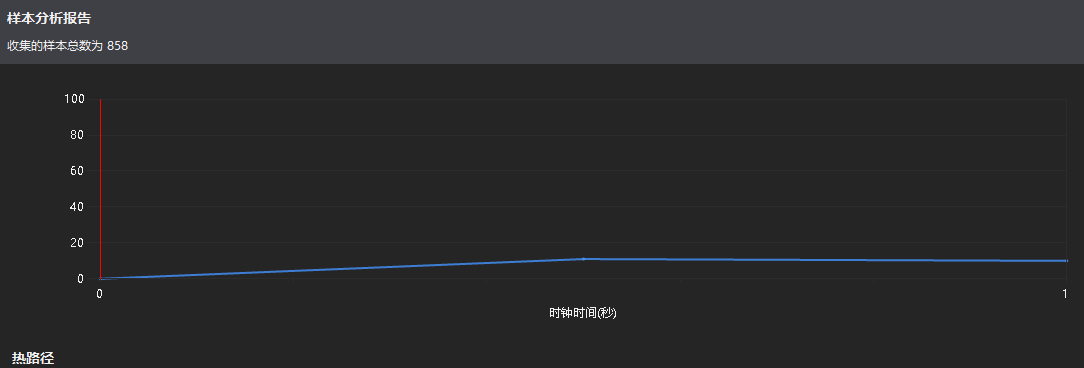
对于一般的错误命令行或者错误文件可以正确识别并且处理

代码分析工具的Warning都已经消除(是由于文件输入顺序不确定造成的潜在风险)

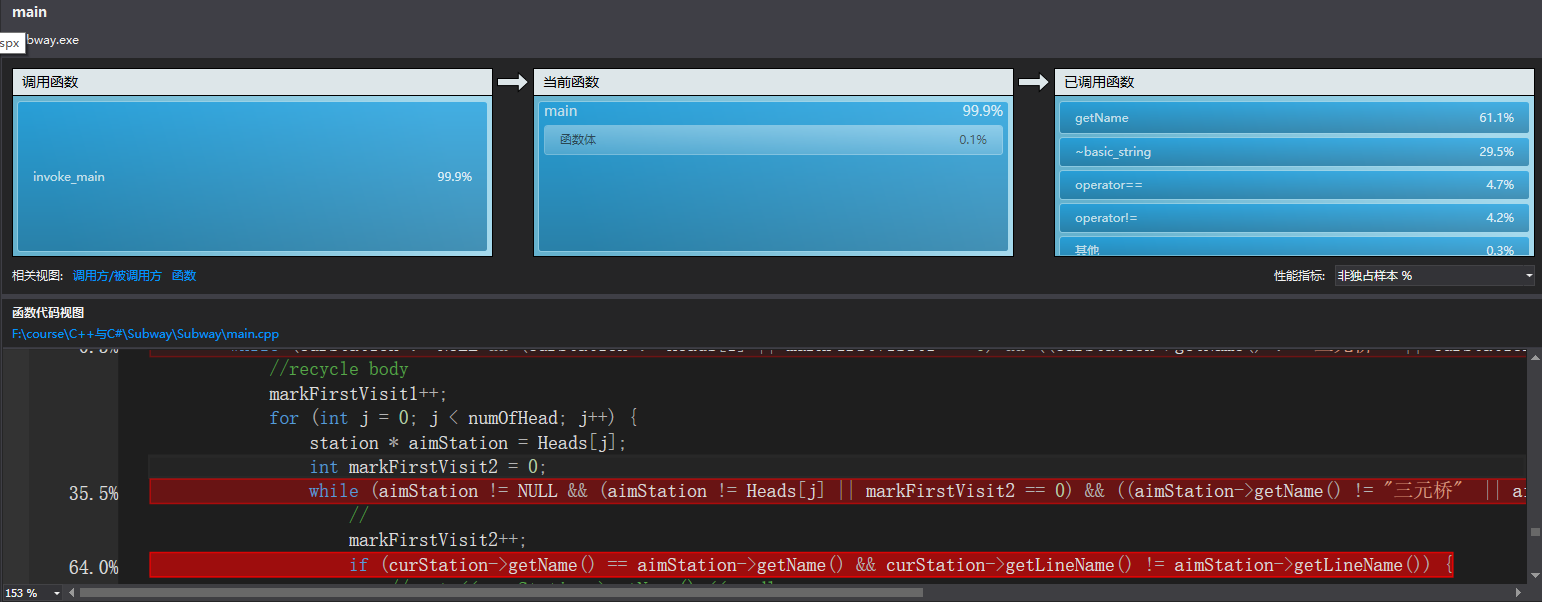
测试用例见另一文件

优化记录：

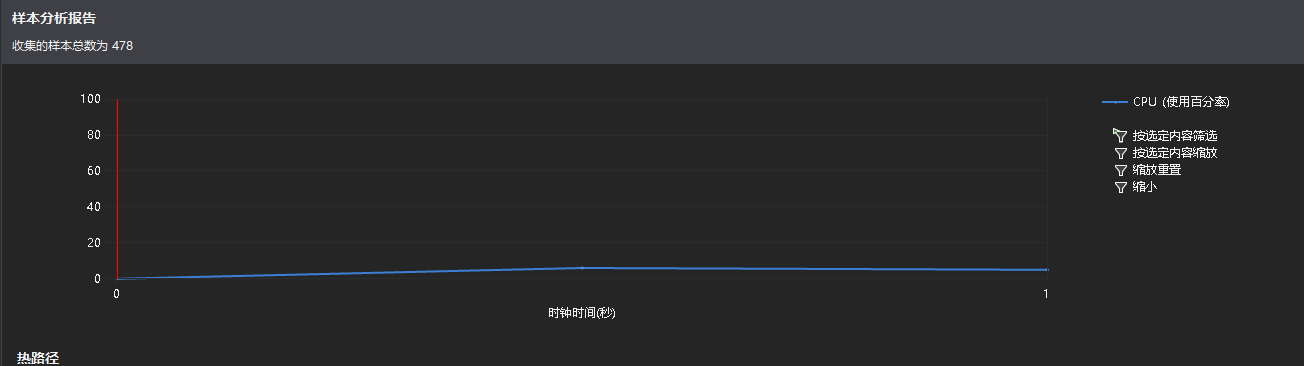
初始性能（样例 公主坟-公益西桥）

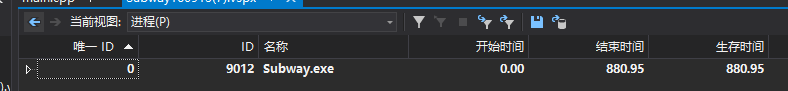


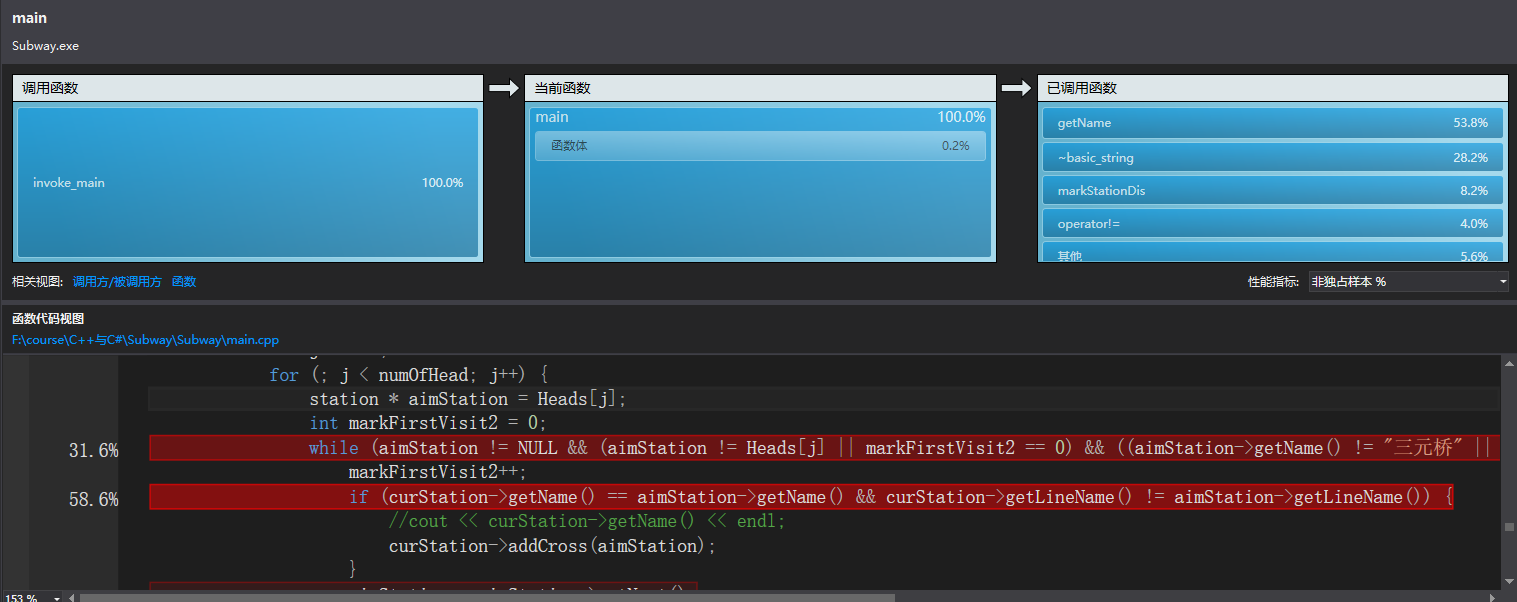




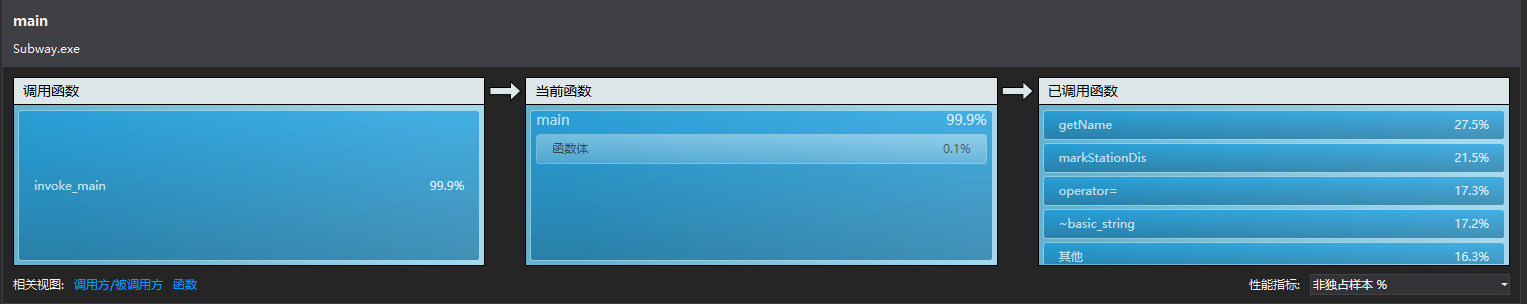
初步改进之后：

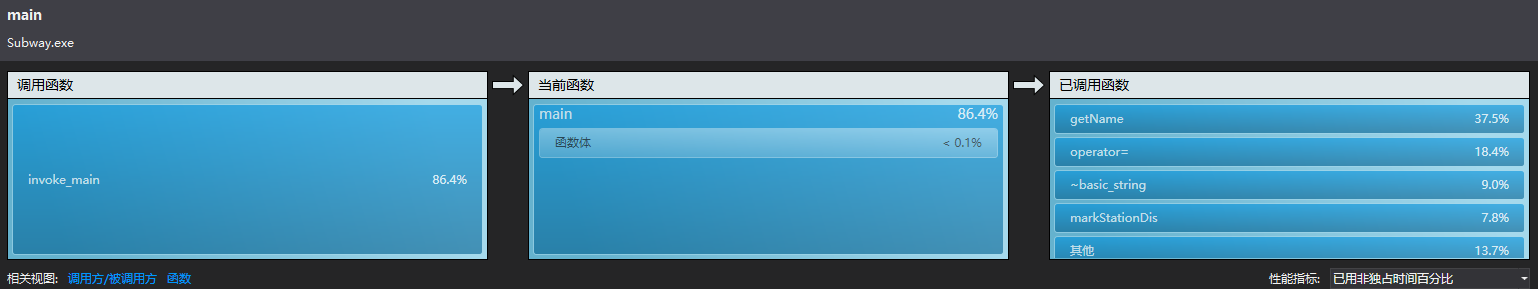




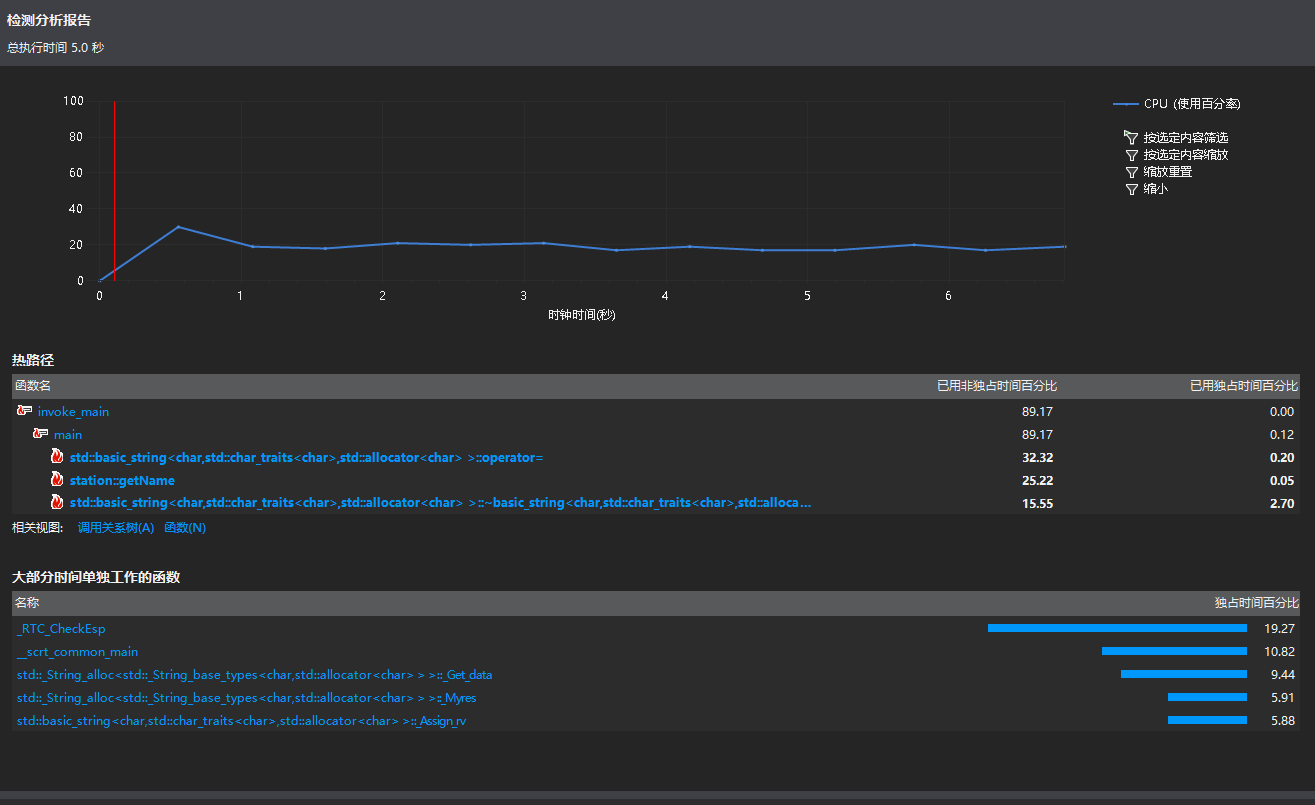


再次改进之后：





最终结果：



可以看到最初的代码运行效率极低，很多地方重复的比较严重。最初最大的问题就是GetName方法调用过于频繁，首先得做法是在集中调用的地方调用一次并且储存起来，以后就不需要重复调用这个方法。此外针对路线储存方式做出了极大的优化: 本来是在站点类里加入一个字符串属性专门用来储存线路，但这样导致每次访问都会更改，尤其在广度优先遍历的情况下更为严重。后来的优化方法是去除掉所有与存储路线相关的内容，改为最后得出结果后由目的地出发回溯得出所经历的路线，效果显著。

性能分析如上图所示，注意最后一张是采用检测的方法而非一般的CPU采样(因为CPU采样在某一时间段忽然开始鬼畜的一直显示“没有可用的调用树数据”)所以这种检测对于时间的影响非常的大。但是实际运行还是很快的。

模块开发耗时预计：

|  |  |  |
| --- | --- | --- |
| **PSP 2.1** | **Personal Software Process Stages** | **Time** |
| **Planning** | **计划** |  |
| · Estimate | · 估计这个任务需要多少时间 | 10m |
| **Development** | **开发** |  |
| · Analysis | · 需求分析 (包括学习新技术) | 1h |
| · Design Spec | · 生成设计文档 | 1h |
| · Design Review | · 设计复审 (和同事审核设计文档) | 30m |
| · Coding Standard | · 代码规范 (为目前的开发制定合适的规范) | 10m |
| · Design | · 具体设计 | 2h |
| · Coding | · 具体编码 | 3h |
| · Code Review | · 代码复审 | 1h |
| · Test | · 测试（自我测试，修改代码，提交修改） | 1h |
| **Reporting** | **报告** |  |
| · Test Report | · 测试报告 | 1h |
| · Size Measurement | · 计算工作量 | 30m |
| · Postmortem & Process Improvement Plan | · 事后总结, 并提出过程改进计划 | 10m |
|  | 合计 | 11.5h |

实际模块开发耗时：

|  |  |  |
| --- | --- | --- |
| **PSP 2.1** | **Personal Software Process Stages** | **Time** |
| **Planning** | **计划** |  |
| · Estimate | · 估计这个任务需要多少时间 | 10m |
| **Development** | **开发** |  |
| · Analysis | · 需求分析 (包括学习新技术) | 1.5h |
| · Design Spec | · 生成设计文档 | 0h |
| · Design Review | · 设计复审 (和同事审核设计文档) | 10m |
| · Coding Standard | · 代码规范 (为目前的开发制定合适的规范) | 10m |
| · Design | · 具体设计 | 1h |
| · Coding | · 具体编码 | 3h |
| · Code Review | · 代码复审 | 3h |
| · Test | · 测试（自我测试，修改代码，提交修改） | 3h |
| **Reporting** | **报告** |  |
| · Test Report | · 测试报告 | 1h |
| · Size Measurement | · 计算工作量 | 0 |
| · Postmortem & Process Improvement Plan | · 事后总结, 并提出过程改进计划 | 10m |
|  | 合计 | 12h10m |

收获：

1. 学习了VS代码分析工具的使用，真的非常的强大以及有用，也让我顺利的找到的代码的性能瓶颈并且可以直接做出针对性的优化（除了鬼畜的采样分析）
2. 提高了编码的能力，尤其是在优化的过程中，可以不断的发现自己之前代码中的不足，并且相处更优的算法。
3. 提高了系统开发软件的能力：虽然这次只是一个很小的系统，但是流程却非常的完整，从前期的准备思考规划预计，到代码的实际编写以及调试，中间穿插着与其他人的交流沟通，到对自己作品的改进直到最后的成品的产生。这一流程非常真实的模拟了实际开发过程，为我们以后的工作积累了经验。