稿

大家好，下面由我们组来为大家介绍我们的大作业项目：实时文本协作插件colive。这是我们小组的成员。

首先由我来介绍我们的项目背景。众所周知，多人同时编写latex文档是一件很麻烦的事情。传统的解决方案是使用overleaf这一网站进行在线编辑，或者装vscode插件。但overleaf访问速度较慢，而且两人以上协作要收费；而live share的连接是不稳定，且必须依赖于vscode。我们参考了2018年的项目，同时查阅了语言服务器这一新技术的相关资料，决定编写支持协作实时编辑的插件，并使之可以跨编辑器工作。

这是我们的项目简介。

这是我们的技术路线：首先我们通过查阅资料学习CRDT这种能实现协作编辑的算法，然后我们编写vscode插件，再然后我们使之符合Language Server的标准，从而容易移植到其他文本编辑器上，最后我们通过自己搭建的服务器实现多人协作功能。

下面由我来为大家介绍CRDT算法。一个普通的文本编辑器至少要实现以下功能：对字符串的增删，对操作的撤销和重做；然而在2018年的项目中，他们只实现了对单个字符的操作。

这是他们实现的文本编辑器，注意到我将这个字符串复制到这个字符串后面，应该出现这样的结果，但是中间六个字符的顺序是错乱的，说明他们的算法不能正确处理字符串操作。

我们认识到，基于字符串的操作是复制粘贴选择删除查找替换等文本操作的基础，而撤销操作的实现在多人协作的情况下又很复杂，这是我们需要解决的难题。通过查阅资料，我们了解了较早的实现方法。

首先是操作变换这种方法，我们直接来看这个例子，一个用户在s[1]之前插入字符'd'，这样的操作导致整个字符串变长，另一个用户并发地删除s[2]时，程序必须能知道删除的是第3个字符而不是第2个字符。这种算法依赖于将操作历史排序，并将后面的操作变换使之与前面的操作相容，它的问题是遇到字符串这样的复杂情况时，很难找到一个不被举出反例的变换规则。

然后CRDT这种算法，CRDT的全称是无冲突可复制数据类型。它通过数据结构而非操作变换来处理并发请求。CRDT为每一个字符都设计了一个标识符，当客户对字符操作时，后台转化为对标识符操作，这样就不会因为操作冲突而产生不一致了。我们通过这张图可以看出，CRDT的突出特点是在每一个客户端都有一个merge程序，这些merge程序的结果是相同的。而OT是通过服务器端的一个merge程序处理冲突，然后把结果分发到每一个客户端。CRDT的问题是，由于它没有保存操作的历史，撤销操作很难实现。

所以我们计划将两种方法结合。首先用户只能看见view上的字符串，他对view进行的操作被记录到Qv队列里，其他用户的操作由广播到传递到Qin队列里，这些操作由model通过CRDT算法整合并渲染回view，然后将整合的结果通过Qout队列广播给其他用户。

这是解决插入冲突的一个例子。我们可以看到，Peer3想要在a和h之间插入字符串X，但Peer4有一个并发的ah间插入操作f，我们通过规定Peer序号大的放在右边，所以Peer3只能在af之间插入，然后我们再判断有没有并发的af间插入操作，这样递归地确定字符串插入的位置。

下面请hwz同学为大家介绍vscode插件开发的相关内容。