WEB服务搭建在本地机器上在linux命令行输入python3 server.py运行。同时调用在线分词接口的程序也在本地运行，在命令行中输入python3 client.py [进程数] [每个进程的线程数] [语料位置]，运行时输入参数来设置程序运行的进程数和线程数以及需要划分的语料文件。

编译过程：Python作为解释型语言仅在运行时翻译成机器语言，因此导入相应的包，直接运行不需要额外进行编译操作。

添加新词方法：

LAC内部实现了load\_customization()接口，参数为：

①path：新词词典位置

②sep：词典文件采用的分隔符，sep=None时，默认空格与制表符为分隔符

如果需要按特点的词划分，则在词典中输入需要相应的内容，可以对一个词语之间添加分隔符，在分词时遇到该词语则默认将其分开，其他词语不受影响。

例如：当词典如图1时，服务器返回的结果如图2所示。

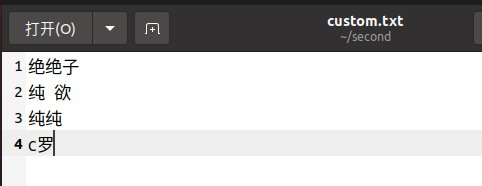


图1 新词词典

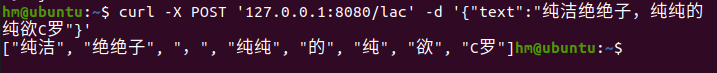


图2 分词结果

本地配置为：4核CPU，8G内存的linux虚拟机。

对一个138Mb的语料文件进行分词，根据多组测试得到的结果如下表1所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 并发进程数 | 分词线程数 | 分词时间 | 分词速度 |
| 2 | 5 | 32分钟 | 0.071mb/s(73kb/s) |
| 2 | 8 | 28分钟 | 0.082mb/s(85kb/s) |
| 2 | 10 | 31分钟 | 0.072mb/s(74kb/s) |
| 8 | 5 | 27分钟 | 0.086mb/s(88kb/s) |
| 8 | 8 | 30分钟 | 0.076mb/s(77kb/s) |
| **8** | **10** | **26分钟** | **0.088mb/s (90kb/s)** |

表1 语料分词在不同进程与线程下结果

根据结果可知，当进程数和线程数提升时，分词所需要的时间下降，但差距不是特别明显，可能原因如下：

①调用在线分词的代码在调用发送POST请求时，实际会阻塞等待服务器结果返回，因此异步效果较差。

②对于获取的结果立即进行写入，导致数据较多时IO次数也随之线性增长。

③服务端和客户端都在本地，导致实际性能没有达到要求。

④WEB服务传输需要一定的时间。