实验二　近似连接

计算机系 13 班　　2011011239　　王庆

一. 实验目的

1. 将两个文档进行连接，找出相似度满足阈值的字符串对。

2. 使用 joinED, joinJaccard 两种算法，两种评测标准找出满足阈值的字符串对。

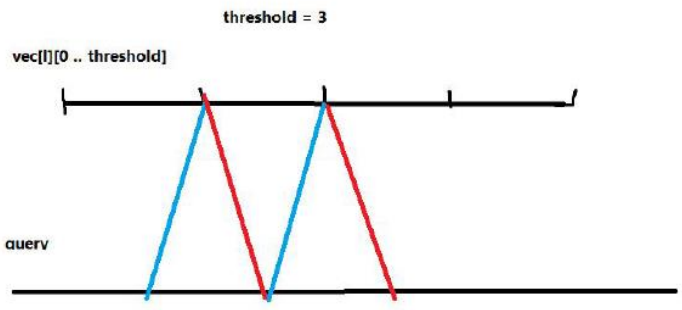
3. 要求运行时间尽可能快，使用内存尽可能小，在保证正确的情况下。

二. 设计思路

- JoinED----position based 算法

基本思想：将被查询的每一个文档分成 threshold+1 段，每一段字符串在查询文档中有一个可能存在区间，如果要求被查询文档的某文档与查询文档的编辑距离小于 threshold，则根据鸽巢原理，必然有一段与查询串的某字串相同。

1. 读取 file1 文件，根据长度建立索引，读取 file2 文件，根据长度建立索引，同时将 file2 中每一行字符串划分成 threshold+1 段,存储每一小段字符串的 hash 值。
2. 对于 file1 中长度为 m 的字符串 str1，找 file2 中长度为 j 的字符串 str2 为候选者， j 属于[threshold-m, threshold+m]区间。枚举 file2 中 str2 的分段在长度为 m 的字符串 str1 可能的区间是否有相同的子字符串，只要 str2 中存在一个分段在 str1 的可能区间有相同的子字符串，则 str1 和 str2 可能编辑距离小于等于 threshold，则将 str2 加入候选者。计算候选者和这行字符串的 ED 距离，如果小于等于阈值，则将相关的文档编号和 ED 距离加入 result。
3. 计算可能区间



设 pi 为第 i 段的起点，i 表示第 i 段(0 ≤ i ≤ threshold),则 str2 中的第 i 段可能

存在的区间：

li=len(str2)/(threshold+1)

pi= len(str2)*i/(threshold+1)

left = max(0, pi - i, pi + (len(str1)-len(str2)) - threshold + i)

right = min(len(str1) - li, pi + i, pi + (len(str1)-len(str2)) + threshold - i)

- JoinJaccard----前缀过滤

1. 统计 file2 中每行字符串经过 q_gram 划分后自字符串的出现频数，如果一行字符串划分的子字符串中有重复，比如 abbabbb，如果 q=2，则可以划分成自字符串 ab，bb，ba，ab，bb，bb，第一个 ab 和第二个 ab 是不相同的，第一个 bb 和第二个 bb 和第三个 bb 都是不相同的，所以我们在统计频数的时候，需要分别统计。
2. 建立 file2 中每行字符的前缀的倒排列表。每行字符进行 gram 划分成一堆子字符串，将这堆子字符串根据在 file2 中出现的频数由小到大排序，得到前 p 个 gram，根据这前 p 个 gram 构建倒排列表。

$$p = (1 - \text{threshold}) * |\text{gram}_{num}| + 1 \qquad \text{公式 1}$$

比如：file2 中的某一行为 abbabbb，q=2，则$|\text{gram}_{num}| = 6$

3. 枚举 file1 中的每 i 行字符串，进行 gram 划分成一堆子字符串，通过公式 1 计算出P_i，将这堆子字符串根据在 file2 中出现的频数由小到大排序，得到前P_i个 gram，将这P_i个 gram 倒排列表里面的 file2 对应的 ids 加入候选者。计算候选者和这行字符串的 Jaccard 距离，如果大于等于阈值，则将相关的文档编号和 Jaccard 距离加入 result。

三．实现要点

- JoinED

1. 根据长度快速得到对应长度的字符串，通过建立 vector<vector<string>>数据结果，通过长度索引，得到长度对应的字符串。
2. 比较子字符串是否相等，可以通过比较 hash 值是否相同。由于需要多次比较，则将子字符串的 hash 值记录在内存里面，只算一次，之后随取随用。
3. 为了减小程序运行的时间，调用函数传函数里不变的参时，传地址，这样参数不用 copy 一份。对于需要多次使用的 hash，可以计算后记录下来，而不是每次使用时

都计算一遍。

- JoinJaccard----前缀过滤

1. 对于一个字符串划分出来的子字符串重复的情况，我把这些重复的子字符串当作不同的子字符串进行处理，比如 abbabbb，如果 q=2，则可以划分成子字符串 ab，bb，ba，ab，bb，bb，我在进行统计子字符串出现频数时，将子字符串记录为：ab，bb，ba，ab1，bb1，bb2，即把重复出现的子字符串当成新的不同的子字符串记载。
2. 因为需要先统计所有 grams 出现的频数，然后根据频数和 file2 中的前缀，构建倒排列表，所以需要记录 file2 中每一行划分出来的 grams，不然需要读两次文件，而且每次读文件需要重新处理字符串。

四．遇到的困难

JoinED 和 JoinJaccard 都遇到了数据结构设置不合理的情况，比如说如何存储分离出来的有用数据，使得算法的运行时间尽可能的短。对于算法的理解，也有一定的偏差，在和同学进行深入讨论之后，才更加深入地理解了算法。因为 Jaccard 没有测试数据和测试平台，所以就找了一个大神，要了他的测试数据和结果，但我的 Jaccard 算法比他找出来的结果要多，我把多的结果单独拎出来，验证都是正确的，但也还有可能我也没有找全答案。

五．代码结构

函数名	作用
calc_hash(const string &s, int start, int end);	计算字符串从[start，end)的子字符串的 hash 值，使用 131 进制乘法。
joinJaccard(const char *filename1, const char *filename2, unsigned q, double threshold, vector<JaccardJoinResult> &result);	实现 joinJaccard 算法
getJaccardDistance(vector<string>& grams1, vector<PAIR>& grams2,double threshold);	获取两个 str1 和 str2 的 Jaccard 距离
joinED(const char *filename1, const char *filename2, unsigned q, unsigned threshold, vector<EDJoinResult> &result);	实现 joinED 算法

readData(const char * filename, vector<vector<string>>& data, vector<vector<int>>& data_num);	ED 算法中读取 file1 中的数据
readData2(const char * filename,vector<vector<vector<int>>>& datainfo,vector<vector<string>>& data,vector<vector<int>>& data_num,unsigned threshold);	ED 算法中读取 file2 中的数据，并记录算法需要的某些值
getEditDistance(const string &A, const string &B, int n, int m, unsigned threshold)	获取两个字符串的编辑距离

六．实验结果

Online 测试结果：同一份代码交了 5 次

ID	Homework	Upload Time	Result	Memory Usage(GB)	Time Usage(s)
3740	exp2: Similarity Join	Fri, 6 Jun 2014 23:42:25 +0800	Correct.	0.099	47.76
3739	exp2: Similarity Join	Fri, 6 Jun 2014 23:40:47 +0800	Correct.	0.098	47.458
3738	exp2: Similarity Join	Fri, 6 Jun 2014 23:39:20 +0800	Correct.	0.099	46.919
3737	exp2: Similarity Join	Fri, 6 Jun 2014 23:36:32 +0800	Correct.	0.098	47.607
3736	exp2: Similarity Join	Fri, 6 Jun 2014 16:12:49 +0800	Correct.	0.098	47.379

代码位置：解压后打包 exp2_2011011239 文件夹，直接提交即可。

七．感谢篇

非常感谢老师非常认真地教授给我们知识，通过数据库这门课，我了解了很多新的搜索算法，学习了很多 paper，收获很大。非常感谢助教帮我们搭建了测试平台，帮助我们更好的了解我们设计的算法的性能！

祝愿李老师身体健康，工作顺利，一切都好！

祝愿助教学业进步，多中 paper，一切顺利！