实验三 近似抽取报告

2011011239　计算机系 13 班　王庆

使用的方法：Single Heap Based Method

步骤：

1. 用 entity.txt 里面的字符串 dictionary 构建 q_gram 划分子字符串的倒排列表。
2. 对于每一个文档 document 构建一个 single heap。
3. 通过记录 dictionary 每一个字符串的子字符串在 document 中的位置，来调整 single heap，获得可能相似的字符串。
4. 验证可能相似的字符串是否相似。

参考一篇关于 Faerie Algorithm 的算法，伪代码如下：

Input: A dictionary of entities $E = \{e_1, e_2, \ldots, e_n\}$;
A document D;
A similarity function and a threshold;
Output: $\{\langle s, e\rangle |\ s$ and e are similar for the function and threshold$\}$, where s is a substring of D and $e \in E$.

```
begin
    Tokenize entities in E and construct an inverted index;
    Tokenize D and get inverted lists of tokens in D;
    Construct a heap H on top of inverted lists of D;
    e is the top element of H; /* keep the current entity*/
    Initialize a position list P_e = φ;
    while (⟨e_i, p_i⟩ = H.top)! = φ do
        if e_i == e then
            P_e∪ = {p_i}; /* e_i is the new top entity. */
        else
            Derive the threshold T_l for entity e;
            if |P_e| ≥ T_l then
                Find candidate windows using Algorithm 1;
                Get candidates using candidate windows;
            e = e_i; P_e = {p_i}; // update the current entity
        Adjust the heap;
    Verify candidate pairs;
end
```

Algorithm 1: Find Candidate Windows

Input: e: An entity; P_e: Position list of e on the heap;
T_l: Threshold; $\top_e$: The upper bound of token numbers;

```
1 begin
2     i = 1;
3     while i ≤ |P_e| − T_l + 1 do
4         j = i + T_l − 1;
5         if p_j − p_i + 1 ≤ ⊤_e then
6             BinarySpan(i, j);
7             i = i + 1; /* shift to the next window */
8         else i = BinaryShift(i, j);
9 end
```

Procedure BinarySpan(i, j)

Input: i: the start point; j: the end point;

```
1 begin
2     lower = j; upper = i + ⊤_e − 1 ;
3     while lower ≤ upper do
4         mid = ⌈(upper + lower)/2⌉;
5         if p_mid − p_i + 1 > ⊤_e then upper = mid − 1;
6         else lower = mid + 1;
7     mid = upper ;
8     Find candidate windows in D[i ··· mid];
9 end
```

Procedure BinaryShift(i, j)

Input: i: the start point; j: the end point
Output: i: the new start point;

```
1  begin
2      lower = i; upper = j;
3      while lower ≤ upper do
4          mid = ⌈(lower + upper)/2⌉;
5          if (p_j + (mid − i)) − p_mid + 1 > ⊤_e then
6              lower = mid + 1;
7          else upper = mid − 1;
8      i = lower; j = i + T_l − 1;
9      if p_j − p_i + 1 > ⊤_e then i = BinaryShift (i, j);
10     else return i;
11 end
```

在剪枝的过程中，对于 ED 和 Jaccard 分别计算 TL，TE， $\perp_e$

- Jaccard Similarity: $T_l = \lceil |e| * \delta \rceil$.
- Cosine Similarity: $T_l = \lceil |e| * \delta^2 \rceil$.
- Dice Similarity: $T_l = \lceil |e| * \frac{\delta}{2-\delta} \rceil$.
- Edit Distance: $T_l = |e| - \tau * q$.
- Edit Similarity: $T_l = \lceil |e| - \left((|e| + q - 1) * \frac{(1-\delta)}{\delta} * q \right) \rceil$.

- Jaccard Similarity: $\perp_e = \lceil |e| * \delta \rceil$ and $\top_e = \lfloor \frac{|e|}{\delta} \rfloor$.
- Cosine Similarity: $\perp_e = \lceil |e| * \delta^2 \rceil$ and $\top_e = \lfloor \frac{|e|}{\delta^2} \rfloor$.
- Dice Similarity: $\perp_e = \lceil |e| * \frac{\delta}{2-\delta} \rceil$ and $\top_e = \lfloor |e| * \frac{2-\delta}{\delta} \rfloor$.
- Edit Distance: $\perp_e = |e| - \tau$ and $\top_e = |e| + \tau$.
- Edit Similarity: $\perp_e = \lceil (|e| + q - 1) * \delta - (q-1) \rceil$ and $\top_e = \lfloor \frac{|e|+q-1}{\delta} - (q-1) \rfloor$.

参考网址：

http://dl.acm.org/citation.cfm?id=1989379

最后，感谢尚泽远同学对我的帮助，感谢助教辛苦的批改作业，感谢老师的教授和提供的相关资料。祝愿大家一切都好^_^