doi: 10.12068/j.issn.1005-3026.2019.11.005

一种新颖的编辑距离限制下的相似性确认算法

于长永, 李森森, 赵 楚, 马海涛
(东北大学秦皇岛分校 计算机与通信工程学院, 河北 秦皇岛 066004)

摘 要: 针对相似性确认步骤中编辑距离计算的高复杂性问题,提出了一种在编辑距离限制下的基于鸽笼原理的字符串相似性确认算法. 首先找到满足编辑距离片段映射的片段,以此片段为基准,将长度为500 bp的read分段. 然后对满足编辑距离片段映射的左右部分递归地进行编辑距离计算,将各段得到的编辑距离相加即为最后结果. 最后根据最长公共子串的下限将需要验证的片段数目降到最低,得到优化方案. 实验结果表明,基于鸽笼原理的分段递归计算编辑距离的确认算法减少了验证步骤的时间,并能保证假阳率和假阴率都为零.

关 键 词: 读取映射;编辑距离;相似性查询;鸽笼原理;确认算法
中图分类号: TP 311.131 **文献标志码**: A **文章编号**: 1005-3026(2019)11-1543-06

A Novel Similarity Verification Algorithm Under Edit Distance Limitation

YU Chang-yong, *LI Miao-miao*, *ZHAO Chu*, *MA Hai-tao*
(School of Computer & Communication Engineering, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China. Corresponding author: LI Miao-miao, E-mail: lmm_neu@126.com)

Abstract: For the high complexity problem of edit distance calculation in the similarity confirmation step, a string similarity confirmation algorithm based on pigonhole principle under the limitation of edit distance is proposed. Firstly, it found a segment satisfies the edit distance segment mapping, then based on this segment, the read length of 500 bp was segmented. Then the edit distance of right and left parts was calculated and all parts sum to the final result. Finally, based on the longest common substring, the number of segments to be verified is minimized, and an optimization scheme was obtained. The experimental result show that this verification algorithm reduces the time of the verification step, and ensures the zero false positive and false negative rate.

Key words: read mapping; edit distance; similarity search; pigonhole principle; verification algorithm

字符串在计算机系统中无处不在,因此字符串处理得到了广泛的研究. 字符串处理中最重要的问题是如何有效地评估两个字符串间的相似性[1]. 相似性查询,即给定一个字符串集合和一个查询字符串,在这个集合中找到所有与查询串满足相似性度量的字符串[2]. 相似性查询的应用场景非常广泛,在基因序列比对、拼写检查、复制检测等方面都有涉及[3-6]. 因此,字符串的相似性查询成为数据领域的一个关注点.

现有的大多数相似性查询都要求具有一个特定的相似性度量函数,例如,Jaccard相似性、余弦相似性和编辑距离[7-11]. 本文只依据两个字符串间的编辑距离来判断它们是否相似,但是,当数据集很大时,穷举所有字符串对的编辑距离是不现实的. 为了解决这一问题,使用过滤加验证的框架进行相似性查询[12]. 在验证阶段,对候选集合进行编辑距离验证,得到满足阈值条件的结果集[13]. 很显然,高效的编辑距离计算方法对算法

收稿日期: 2019-03-13
基金项目: 国家自然科学基金资助项目(61772124,61401080,61402087); 河北省自然科学基金资助项目(F2015501049); 河北省教育厅项目(QN2014339); 中央高校基本科研业务费专项资金资助项目(N150402002).
作者简介: 于长永(1981-),男,辽宁海城人,东北大学副教授.

的时间性能有很大的提升.

本文研究基于鸽笼原理的分段递归计算编辑距离的字符串相似性查询方法. 为了证明本文所提算法的有效性,与现有的方法进行了比较. 实验结果表明,该方法能够在时间上达到更优的效果.

1 问题定义及相关工作

定义 1 (编辑距离限制下的相似性查询问题):设 $S=\{s_1,s_2,\cdots,s_n\}$ 是一个字符串集合,q 表示一个查询字符串,τ 代表编辑距离阈值,式(1)表示在编辑距离限制下的相似性查找结果.

$$R(S,q,\tau)=\{\mathrm{ed}(s_i,q)\leqslant\tau\mid i=1,2,\cdots,n\}. \quad (1)$$

其中,$\mathrm{ed}(s_i,q)$表示将 q 转换为 s_i 所需的最少操作次数,包括插入、删除、替换操作[14]. 字符串相似性查询问题是字符串研究方向的基础问题. 最原始的查询算法是将查询串和数据库中每一个字符串两两计算编辑距离,很显然,这是非常耗时的. 为了减少相似性查询所用的时间,采用过滤加验证的框架. 大量实验数据表明,验证占据了大部分的时间. 目前可以用来优化验证的方法主要分为两类:①减少候选集合的规模; ②提高编辑距离计算速度. 对于方法①,由于候选集合的规模大小会有一个下限,当达到一定规模时,再减小候选集合规模,会导致过滤成本大大增加. 对于方法②,大部分算法是基于动态规划进行优化. 例如,基于范围的计算[1]、分层计算、前缀树[1,15-16]等方法. 但是,对于大规模数据而言,如何进行验证步骤的优化、实现快速地相似性查询仍然具有挑战性.

1.1 动态规划方法

设 s 和 r 是两个字符串,利用 $|r|+1$ 行 $|s|+1$ 列的矩阵 $\boldsymbol{D}$ 计算 $\mathrm{ed}(s,r)$. 其中,$|r|$ 和 $|s|$ 分别表示字符串 r 和字符串 s 的长度. $s[i,j]$ 表示 s 中从 i 到 j 的子串. $\mathrm{ed}(s,r)$ 表示将 r 转换到 s 所需要的最少操作次数(插入、删除、替换). $\boldsymbol{D}[i,j]$ 表示前缀 $r[1,h]$ 和前缀 $s[1,j]$ 的编辑距离,很显然,对于 $0\leqslant i\leqslant|r|$,$\boldsymbol{D}[i,0]=i$ 成立;对于 $0\leqslant j\leqslant|s|$,$\boldsymbol{D}[0,j]=j$ 成立. 按照式(2)迭代计算 $\boldsymbol{D}[i,j]$:

$$\boldsymbol{D}[i,j]=\min(\boldsymbol{D}[i-1,j]+1,\boldsymbol{D}[i,j-1]+1,\boldsymbol{D}[i-1,j-1]+\&). \quad (2)$$

当 $s[j]=r[i]$ 时,$\&=0$,否则 $\&=1$. 该方法时间复杂度为 $O(|r|\times|s|)$. 给定数据集字符串 $s=$ series,查询串 $r=$ seraji,图 1 给出了利用原始方法计算编辑距离的距离矩阵.

		s	e	r	i	e	s
	0	1	2	3	4	5	6
s	1	0	1	2	3	4	4
e	2	1	0	1	2	3	4
r	3	2	1	0	1	2	3
a	4	3	2	1	1	2	3
j	5	4	3	2	2	2	3
i	6	5	4	3	2	3	3

图 1 动态规划计算编辑距离

Fig. 1 Computing edit distance by dynamic programming

1.2 范围

给定字符串 s 和 r,$|s|=m$,$|r|=n$,编辑距离阈值为 τ,那么只需要计算矩阵中的 K 条对角线即可,K 的值可以由式(3)确定. 同时,当主对角线上的值出现 $\tau-|m-n|+1$ 时,即可判断字符串 s 和 r 不相似.

$$K=\begin{cases}\tau+1,(\tau-(m-n))\%2=0;\\ \tau,(\tau-(m-n))\%2=1.\end{cases} \quad (3)$$

例如,$s=$ series,$r=$ seraji,如果给定阈值 $\tau=2$,那么 $(\tau-(m-n))\%2=0$,最多需要计算 $\tau+1$ 条对角线.

1.3 层级

令 $<i,j>$ 表示距离矩阵中第 i 行第 j 列的实体,E_x 表示矩阵中值为 x 的实体集合. 给定字符串 s 和 r,编辑距离阈值 τ,为了计算 r 和 s 的编辑距离,需要递归的计算 E_x,$x=0,1,\cdots,n$. 如果 $<|r|,|s|>\in E_x$,那么 $\mathrm{ed}(s,r)=x$. 当 $\mathrm{ed}(s,r)\leqslant\tau$ 时,加入结果集中. 例如,$s=$ series,$r=$ seraji,逐步计算编辑距离的过程如下:

首先,$E_0=\{<1,1>,<2,2>,<3,3>\}$,$E_1=\{<2,1>,<1,2>,<3,2>,<2,3><4,3>,<3,4>,<4,4>\}$,$<|r|,|s|>$不属于 E_1,根据 E_1 计算 E_2,$E_2=\{<3,1>,<1,3>,<4,2>,<5,3>,<6,4>,<2,4>,<3,5>,<4,5>,<5,5>,<6,4>\}$,$<|r|,|s|>$不属于 E_2,根据 E_2 计算 E_3,$E_3=\{<6,5>,<5,6>,<6,6>\}$,$<|r|,|s|>\in E_3$,因此 $\mathrm{ed}(s,r)=3$. 计算过程见图 2,时间复杂度为 $O(\tau\times\min(|r|,|s|))$.

这种方法需要枚举数据集中每个字符串,计算复杂度也很大. 除此之外,时间复杂度和 $\min(|r|,|s|)$ 以及字符串的长度有关. 很显然,

当字符串很长时,时间性能会有所下降. 由于暴力枚举数据集中的每个字符串存在困难,可以利用不同字符串间的共同前缀共享计算.

		s	e	r	i	e	s
	0	1	2	3	4	5	6
s	1	0	1	2			
e	2	1	0	1	2		
r	3	2	1	0	1	2	
a	4		2	1	1	2	
j	5			2	2	2	3
i	6				2	3	3

图 2　分层计算编辑距离

Fig. 2　Leveled edit distance calculation

1.4　前缀树

由于数据库中的字符串类似于基因序列、Web 网址等具有高度重复性,因此,对集合中的字符串动态构建一个前缀树结构,有利于解决具有相同前缀的字符串重复性问题.

给定字符串数据集 S 和查询字符串 q,$S=\{s_1,s_2,\cdots,s_n\}$,对 S 中的字符串构建前缀树,相同字符串具有相同的前缀,在前缀树上表示为具有相同的节点. 例如,s_1 = AGGAA,s_2 = AGGAC,q = AGGCT,那么 s_1 和 s_2 共享 4 个前缀树节点. 因此,可以直接利用 s_1 的子串 AGGA 与 q = AGGCT 的编辑距离快速得到 s_2 和 q 的编辑距离.

但是,对候选集合中存在的较长字符串而言,前缀树结构也不能很好地解决复杂的编辑距离计算问题.

通过对上面三种编辑距离计算方法进行分析,相对于最初的利用动态规划来计算编辑距离,虽然在时间上有所提升,但是,当 $\mathrm{ed}(s,r)>\tau$,或者 $\max(|s|,|r|)$ 很大时,以上方法并不能达到很好的效果. 为了解决该问题,本文提出一种在片段映射上递归地利用鸽笼原理计算编辑距离的算法,并对该算法提出了优化方案.

2　Recursive – ed 算法

2.1　编辑距离片段映射

定义 2　(编辑距离片段映射):给定两个字符串 r 和 s. $P(r)=\{\mathrm{seg}^r(i)\}$ 和 $P(s)=\{\mathrm{seg}^s(i)\}$,$i=1,\cdots,n$,分别表示 s 和 r 的分段集合. 如果编辑距离 $\mathrm{ed}(r,s)$ 满足式(4),那么称 $f(\mathrm{seg}^r(i))=\mathrm{seg}^s(i)$ 为编辑距离分段映射(ed – mapping). 设 $\mathrm{ed}_l,\mathrm{ed}_r$ 表示 $\mathrm{seg}^s(i)$ 与 $\mathrm{seg}^r(i)$ 之间左、右两部分的编辑距离,如果满足式(5),则称 $\mathrm{seg}^r(i)$ 与 $\mathrm{seg}^s(i)$ 之间是 ed – mapping,记为 $f^{\mathrm{ed}}(*)$.

$$\mathrm{ed}(s,r)=\sum_{i=0}^{n}\mathrm{ed}(\mathrm{seg}^s(i),\mathrm{seg}^r(i)),\quad(4)$$

$$\mathrm{ed}(s,r)=\mathrm{ed}_l+\mathrm{ed}_i(s,r)+\mathrm{ed}_r.\quad(5)$$

例如,给定数据集中的一个字符串 s = abcd,查询串 r = acxd,根据式(4),可以列出从字符串 s 到字符串 r 的编辑距离片段映射,ε 表示空字符.

1) $f(\mathrm{a})=\mathrm{a}$,$f(\mathrm{b})=\varepsilon$,$f(\mathrm{c})=\mathrm{c}$,$f(\varepsilon)=\mathrm{x}$,$f(\mathrm{d})=\mathrm{d}$,

2) $f(\mathrm{ab})=\mathrm{a}$,$f(\mathrm{c})=\mathrm{c}$,$f(\mathrm{d})=\mathrm{xd}$,

3) $f(\mathrm{ab})=\mathrm{a}$,$f(\mathrm{c})=\mathrm{cx}$,$f(\mathrm{d})=\mathrm{d}$,

4) $f(\mathrm{abc})=\mathrm{ac}$,$f(\mathrm{d})=\mathrm{xd}$.

根据定义 2,如果知道字符串 r 和 s 之间的 1 个编辑距离片段映射,那么 2 个字符串间复杂的编辑距离的计算可以转换为多个子字符串编辑距离的求和计算,减少长字符串编辑距离计算的复杂度. 但是使用定义 2 去确定两个字符串中的分段是否满足 ed – mapping 是一个复杂的过程,下面通过定理 1 和定理 3 来介绍如何巧妙地运用 ed – mapping 解决加速计算编辑距离的问题.

定理 1　给定两个字符串 s 和 r,将 r 分成 n 段,如果 s 与 r 在给定的编辑阈值 τ 下相似,那么至少存在一个 $\mathrm{seg}^r(i)$ $(1\leqslant i\leqslant n)$ 和一个 s 中的子串 s_m,满足:

1) $\mathrm{ed}(\mathrm{seg}^r(i),s_m)\leqslant\lfloor\frac{\tau}{n}\rfloor$;

2) $f^{\mathrm{ed}}(\mathrm{seg}^r(i))=s_m$.

证明　由于 $\mathrm{ed}(s,r)\leqslant\tau$,即对 r 进行不多于 τ 次的编辑操作就可以将 r 变为 s,因此,将 r 分为 n 段后,至少有一段在变换的过程中发生的编辑操作次数小于 $\lfloor\frac{\tau}{n}\rfloor$,设该段为 $\mathrm{seg}^r(i)$,同时设该段在变换中变为 s 的子串 s_m,则有 $\mathrm{seg}^r(i)$ 和 s_m 满足条件 1 和条件 2,证毕.

在定理 1 中,给出了快速判断两个字符串分段中是否存在 ed – mapping,同时以下推论也成立:

1) 如果 $\tau<n$,那么 $\mathrm{seg}^r(i)=s_m$.

2) $\mathrm{seg}^r(i)$ 在字符串 r 中的位置与 s_m 在 s 中的位置差小于等于 τ.

2.2 Recursive – ed 算法

定理 2 给定两个字符串 s 和 r,将 r 分成 n 段,如果 s 与 r 在给定的编辑阈值 τ 下相似,那么可以推断出定理 1 中的结论. 并且以下结论也成立:

3) 如果找到编辑距离映射 $\mathrm{ed}(\mathrm{seg}_i^r, s_m)$,那么左半部分的 r_l 和 s_l 必须满足定理 1,其中左半部分的编辑距离阈值为 $\tau_l = \tau - (|r_r| - |s_r| + \mathrm{ed}(\mathrm{seg}_i^r, s_m))$.

4) 右半部分同理,r_r 和 s_r 必须满足定理 1,此时的编辑距离阈值为 $\tau_r = \tau - (|r_l| - |s_l| + \mathrm{ed}(\mathrm{seg}_i^r, s_m))$.

其中,$\mathrm{seg}^r(i)$ 和 s_m 满足 $f^{\mathrm{ed}}(\mathrm{seg}^r(i)) = s_m$,$r_l$,$r_r$ 和 s_l,s_r 分别表示 $\mathrm{seg}^r(i)$ 和 s_m 的左、右两部分,τ_l,τ_r 分别表示 r_l 和 s_l,r_r 和 s_r 之间的编辑距离阈值. 通过上式可递归的计算编辑距离.

定理 3 如果两个字符串 r 和 s 满足 $\mathrm{ed}(s,r) \leqslant \tau$,$r$ 的分段集合为 $P(r) = \{\mathrm{seg}^r(i)\}$,那么有 $\mathrm{ed}(s,r) = \min_{i=1}^{n}\{\mathrm{ed}_i(s,r) \mid f(\mathrm{seg}^r(i)) = s_m^i\}$.

其中,$f(\mathrm{seg}^r(i)) = s_m^i$ 表示 $\mathrm{seg}^r(i)$ 和 s_m^i 在编辑距离 $\mathrm{ed}(\mathrm{seg}_i^r, s_m) \leqslant \lfloor \frac{\tau}{n} \rfloor$ 限制下是分段映射. 值得注意的是,这样的映射是否为编辑距离映射是不必要的. 只要满足片段 $\mathrm{seg}^r(i)$ 能够映射到 s_m^i,那么就可以用 $\mathrm{ed}_i(s,r) = \mathrm{ed}_l + \mathrm{ed}_r + \mathrm{ed}(\mathrm{seg}_i^r, s_m)$ 来表示总的编辑距离. 根据定理 1,定理 2 和定理 3 显然成立.

2.3 Recursive – ed 算法的优化

定义 3 (最小覆盖字符串):给定字符串 s 和 r,如果 $c(s,r)$ 满足:

1) $\mathrm{ed}(s,r) = \mathrm{ed}(s,c) + \mathrm{ed}(c,r)$;

2) $s \to c$ 过程中只有插入和替换操作,$c \to r$ 过程中只有删除和替换操作.

那么称 $c(s,r)$ 为 s 和 r 的最小覆盖字符串. 很显然,$c(s,r)$ 一定存在并且不唯一,并且 $|c(s,r)| > |s|$,$|c(s,r)| > |r|$. $|c(s,r)|$ 可以通过以下定理得到:

定理 4 给定字符串 s 和 r,令 F 是从 s 到 r 编辑距离映射的集合. 那么,

$$|c(s,r)| = \sum_{i=1}^{n} |c(\mathrm{seg}^r(i), f(\mathrm{seg}^r(i)))|, \tag{6}$$

$$|c(s,r)| = \max_{f \in F}\left(\sum_{i=1}^{n}(\max\{|\mathrm{seg}^r(i)|, |f(\mathrm{seg}^r(i))|\})\right). \tag{7}$$

由于 $|c|$ 不能直接计算出来,因此用 $|c|*$ = $\max\{|s|,|r|\}$ 代替 $|c|$,并随着假设的映射片段的改变,$|c|*$ 逐步改变. 令 $|c|* = \sum_{i=1}^{n} \max\{|\mathrm{seg}^r(i)|, |f(\mathrm{seg}^r(i))|\}$ 作为 $|c|$ 的近似值.

证明 $c(s,r)$ 表示 r 变为 s 的过程中,先对 s 进行所有的插入操作,得到字符串 c,c 的长度为 $|c(s,r)|$. 因此,将 s 分段后,每段变为 r 的一段,这变换中每个对应段的 $\sum_{i=1}^{n} |c(\mathrm{seg}^r(i), f(\mathrm{seg}^r(i)))|$ 表示每段都只进行插入,得到字符串 c_i,c_i 的长度和就是 $\sum_{i=1}^{n} |c(\mathrm{seg}^r(i), f(\mathrm{seg}^r(i)))|$. 因此整体变换和分段变换的结果是相同的,所以式(6)成立.

取 s 和 r 的分段映射,每段 (s_i, r_i) 至多包含一个插入或一个删除,那么有 $c(s_i, r_i) = \max\{|s_i|, |r_i|\}$,所以式(7)成立.

引理 1 (共享 q – gram 的下限):给定字符串 s 和 r,满足 $\mathrm{ed}(s,r) \leqslant \tau$. 那么 s 和 r 至少共享 $N_{\mathrm{q-gram}} = |c(s,r)| - q + 1 - \tau \times q$ 个公共 q – grams. 当 $s \to r$ 只有删除或插入操作时,$\mathrm{LB} = \max\{|s|,|r|\} - q + 1 - q \times \tau$ 是共享 q – grams 的最少数量.

引理 2 (最长公共子串的下限):给定字符串 s 和 r,满足 $\mathrm{ed}(s,r) \leqslant \tau$. 那么必然存在长度至少为 $L_1 = q + \lceil \frac{N_{\mathrm{q-gram}}}{\tau+1} \rceil - 1 = \lceil \frac{|c| - \tau}{\tau+1} \rceil = L_2$. 并且,当共享 q – gram 的数量大于 $N_{\mathrm{q-gram}}$ 时,$L^* = q + \lceil \frac{N_*}{\tau+1} \rceil - 1$.

在优化算法中,通过减少被验证片段的数目,达到优化目的. 由引理 1,引理 2 可知,共享 q – gram 的下限和最长公共子串的下限都与 $|c(s,r)|$ 有关,因此,当一个新的片段映射被假设之后,$|c|*$,$N_{\mathrm{q-gram}}$,L^* 的值都被更新.

由鸽笼原理可知,字符串 r 被分为 $\tau+1$ 个片段,每一个片段都有可能 ed – mapping 到 s 的子串,因此,为了验证 $\mathrm{ed}(s,r) \leqslant \tau$,最坏的情况下需要对 $\tau+1$ 种情况进行验证. 实际上,根据引理 2,只有较长片段才会被验证.

定理 5 给定字符串 s 和 r,如果 $\mathrm{ed}(s,r) \leqslant \tau$,那么必定存在 s 的子串 sub^s 和 r 的子串 sub^r 满足:

1) $\mathrm{sub}^s = \mathrm{sub}^r$;

2) $F_{\mathrm{ed}}(\mathrm{sub}^r) = \mathrm{sub}^s$;

3) $|\mathrm{sub}^r| \geqslant L^*(L_2)$.

根据定理 5，只需要考虑映射片段长度大于 L_2 的情况. 为了找到长度大于 L_2 的所有映射片段，需要将字符串 r 分成 $N* = \lceil \frac{|r|}{\lceil \frac{L_2}{2} \rceil} \rceil$ 个相同长度的片段. 同时，利用 q - gram 的计数过滤，$q = L_2$.

3　实验结果与讨论

3.1　数据集和实验环境

本文实验在 Linux 操作系统下，g + + 编程环境上利用 C + + 语言实现，本文实验使用真实的测序 read 数据集，具体见下表 1，该数据集来自于文献[14]，从该文献的实验数据集中选择两个数据作为本文的实验数据，read 长度为 100 bp，在实验过程中，将长度为 100 bp 的 read 扩展到 500 bp.

表 1　数据集
Table 1　Database

数据集	大小/Mb	长度/bp
ERR240726_1	984	100
ERR240726_2	880	100

3.2　实验结果

根据定理 1 ~ 4，本文提出的算法的假阳性和假阴率均为 0. 当两个字符串 s 和 r 的编辑距离小于 τ 时，Recursive - ed 返回真实的编辑距离，否则 Recursive - ed 返回 $\tau + 1$. 因此，本文提出的算法的准确性是 100% .

对于上述两个数据集，分别随机产生编辑距离 $\tau \leqslant x(x = 3,5,7)$ 的 10 000 条和 30 000 条基因序列，序列数目可以根据实验要求进行更改.

表 2　验证候选集
Table 2　Candidate database

候选集	大小/Mb	基因序列/条
Pairdata_1_3	784. 9	10 000
Pairdata_1_5	783. 5	10 000
Pairdata_1_7	782. 3	10 000
Pairdata_2_3	784. 9	30 000
Pairdata_2_5	783. 5	30 000
Pairdata_2_7	782. 3	30 000

在实验中，为了验证本文所提算法的有效性，将序列的长度定义为 500 bp. 对每个候选集利用本文列举的几种不同编辑距离计算方法，实验结果见表 3.

表 3　Pairdata_1_*x* 各函数所用时间
Table 3　Time spent on Pairdata_1_*x* functions　s

τ	原始	层级	范围	Recursive - ed
3	31. 37	2. 37	2. 59	0. 14
5	31. 84	2. 43	2. 68	0. 22
7	32. 11	2. 51	2. 74	0. 32

为了测试候选序列数目对算法的影响，对验证集 Pairdata_2_*x* 选取 Number = 30 000 的情况，实验结果见表 4.

表 4　Pairdata_2_*x* 各函数所用时间
Table 4　Time spent on Pairdata_2_*x* functions　s

τ	原始	层级	范围	Recursive - ed
3	95. 96	7. 28	7. 88	0. 40
5	96. 73	7. 59	8. 14	0. 69
7	95. 79	7. 66	7. 94	0. 43

4　结　　语

本文提出的 Recursive - ed 算法用来快速计算编辑距离. 在 Recursive - ed 算法中，利用鸽笼原理，通过将序列分段递归计算编辑距离，有效地完成相似性验证步骤. 和已有的其他 4 种算法相比较，本文提出的方法更加适用于较长字符串的相似性比较，能够克服因 $\max(|s|,|r|)$ 很大而造成的计算复杂的问题. 现如今序列数据呈指数模式增长，使得序列间的相似性查询越来越困难. 本文提出的算法可以作为该问题的潜在解决方案.

参考文献：

[1] Deng D, Li G L, Feng J H, et al. Top-k string similarity search with edit-distance constraints [C]// IEEE 29th International Conference on Data Engineering (ICDE). Washington D C, 2013: 925 - 936.

[2] Zhang Z J, Hadjieleftheriou M, Ooi B C, et al. Bed-tree: an all-purpose index structure for string similarity search based on edit distance [C]// Proceedings of the ACM SIGMOD International Conference on Management of Data. Indianapolis, 2010: 915 - 926.

[3] Deng D, Li G L, Feng J H. A pivotal prefix based filtering algorithm for string similarity search [C]// SIGMOD Conference. Snowbird, 2014: 673 - 684.

[4] Hadjieleftheriou M, Koudas N, Srivastava D. Incremental maintenance of length normalized indexes for approximate string matching [C]// SIGMOD Conference. Rhode Island, 2009: 429 - 440.

[5] Li, C, Lu J H, Lu Y M, et al. Efficient merging and filtering algorithms for approximate string searches [J]. *International Conference on Data Engineering*, 2008, 7(12): 257 - 266.

[6] Wandelt S, Wang J, Leser U, et al. State-of-the-art in string similarity search and join [J]. *ACM Sigmod Record*, 2014, 43(1): 64 - 76.

[7] Jiang Y, Li G, Feng J, et al. String similarity joins: an experimental evaluation [J]. *International Conference on Very Large Data Bases*, 2014, 7(8): 625 – 636.
[8] Xiao C, Wang W, Lin X M, et al. Efficient similarity joins for near-duplicate detection [J]. *ACM Transaction*, 2011, 36(3): 131 – 140.
[9] Jiang Y, Deng D, Wang J N, et al. Efficient parallel partition-based algorithms for similarity search and join with edit distance constraints [C]// EDBT/ICDT. New York, 2013: 341 – 348.
[10] Bi F, Chang L, Zhang W, et al. Efficient string similarity search: a cross pivotal based approach [C]// International Conference on Database Systems for Advanced Applications. Hanoi, 2015: 545 – 564.
[11] Levenstein V. Binary codes capable of correcting spurious insertions and deletions of ones [J]. *Problems of Information Transmission*, 1965, 1(1): 8 – 17.
[12] Yu M, Wang J, Li G L, et al. A unified framework for string similarity search with edit-distance constraint [J]. *International Conference on Very Large Data Bases Journal*, 2017, 26(2): 249 – 274.
[13] 米琳. 基于 q – gram 的字符串相似性查询研究[J]. 现代计算机(专业版), 2014(6): 12 – 16.
(Mi Lin. Research on string similarity query based on q-gram [J]. *Modern Computer (Professional Edition)*, 2014(6): 12 – 16.)
[14] Mishra S, Gandhi T, Arora A, et al. Efficient edit distance based string similarity search using deletion neighborhoods [C]// EDBT/ICDT. New York, 2013: 375 – 383.
[15] Rheinländer A, Knobloch M, HochmuthN, et al. Prefix tree indexing for similarity search and similarity joins on genomic data [C]// International Conference on Scientific and Statistical Database Management. Berlin, 2010: 519 – 536.
[16] Xin H, Greth J, Emmons J, et al. Shifted hamming distance: a fast and accurate SIMD-friendly filter to accelerate alignment verification in read mapping [J]. *Bioinformatics*, 2015, 31(10): 1553 – 1560.

(上接第 1542 页)

[3] Zink M D, Brüser C, Winnersbach P, et al. Heartbeat cycle length detection by a ballistocardiographic sensor in atrial fibrillation and sinus rhythm [J]. *Biomed Research International*, 2015(19): 1 – 10.
[4] Etemadi M, Inan O T. Wearable ballistocardiogram and seismocardiogram systems for health and performance [J]. *Journal of Applied Physiology*, 2017, 124(612): 452 – 461.
[5] Lim Y G, Hong K H, Kim K K, et al. Monitoring physiological signals using nonintrusive sensors installed in daily life equipment [J]. *Biomedical Engineering Letters*, 2011, 1(1): 11 – 20.
[6] Bruser C, Diesel J, Zink M D H, et al. Automatic detection of atrial fibrillation in cardiac vibration signals [J]. *IEEE Journal of Biomedical and Health Informatics*, 2013, 17(1): 162 – 171.
[7] Acharya U R, Fujita H, Lih O S, et al. Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network [J]. *Information Sciences*, 2017, 405: 81 – 90.
[8] 金晶晶, 王旭, 杨丹. 基于体震信号的心率测量方法[J]. 东北大学学报(自然科学版), 2009, 30(2): 176 – 179.
(Jin Jing-jing, Wang Xu, Yang Dan. Heart rate measurement based on fluttering signal from human body [J]. *Journal of Northeastern University (Natural Science)*, 2009, 30(2): 176 – 179.)
[9] Fan X, Yao Q, Cai Y, et al. Multi-scaled fusion of deep convolutional neural networks for screening atrial fibrillation from single lead short ECG recordings [J]. *IEEE Journal of Biomedical and Health Informatics*, 2018, 22(6): 1744 – 1753.
[10] Lecun Y L, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition [J]. *Proceedings of the IEEE*, 1998, 86(11): 2278 – 2324.
[11] Golkov V, Dosovitskiy A, Sperl J I, et al. Q-Space deep learning: twelve-fold shorter and model-free diffusion MRI scans [J]. *IEEE Transactions on Medical Imaging*, 2016, 35(5): 1344 – 1351.
[12] Van Grinsven M, van Ginneken B, Hoyng C, et al. Fast convolutional neural network training using selective data sampling: application to hemorrhage detection in color fundus images [J]. *IEEE Transactions on Medical Imaging*, 2016, 35(5): 1273 – 1284.
[13] Kiranyaz S, Ince T, Gabbouj M. Real-time patient-specific ECG classification by 1D convolutional neural networks [J]. *IEEE Transactions on Bio-medical Engineering*, 2015, 63(3): 664 – 675.