D-010

類似文字列検索における可変長 N-gram を用いたマージの効率化

An Efficent Merging Method Using Variable Length N-gram for Approximate String Matching

木村 光樹 † 高須 淳宏 ‡ 安達 淳 ‡
Mitsuki Kimura Atsuhiro Takasu Jun Adachi

1 はじめに

類似文字列検索は大量のオブジェクト中からクエリと似たオブジェクトを探す Similarity Search の対象オブジェクトとして文字列を扱うものである．その上，人が扱う情報は文字列として扱えることが多いため，その応用範囲は広く，様々な技術の基幹として用いることができるが，近年の情報量の増加とともにその高速さと正確性がより求められるようになってきた．

文字列間の類似度を表す指標は，編集距離，Jaccard Similarity, Cosine Similarity, Dice Similarity など様々に存在する．しかしながら，これらの文字列間の類似度を計算するのに一般的に時間がかかってしまうことが知られているために事前に解候補の削減を行う必要がある．近年の類似文字列検索では，類似した文字列同士は共有する gram を多くなるということに着目し，gram ベースで索引付けを行った転置索引を検索に用いることが多い．このとき，類似した文字列は類似度と求める閾値を決めれば，共有する gram の下限値を求めることができる．gram ベースの索引付けを用いた類似文字列検索では，クエリを gram に分割し各 gram を索引語とする転置索引中の各レコード内の文字列 id のリストをマージしていくときに，この下限値より多く出現する文字列 id を解候補とする．しかし，索引付けの際の gram 長 ($= N$) に性能が依存してしまうという問題がある．N の値が大きいとき，索引数はデータ中に存在する文字種の N 乗で増えていき，索引付けに膨大な時間がかかる一方で，長い gram を共有する文字列は一般的に少なくなっていくので，リストのマージが容易に行えるという利点がある．逆に，N が小さいと，索引数は少なくなるが，一つのリスト内に存在する文字列 id が多くなってしまうためにリストのマージに時間がかかってしまう．

このことを踏まえて筆者らは，編集距離を類似度に用いる類似文字列検索において gram の長いとき短いときの両方の利点を上手く利用するために可変長の N-gram を索引付けに用い，それを用いた新しい文字列 id のリストのマージ手法を提案する．索引付けでは，その文字列を代表的に表す部分文字列となる gram である rg(representattive gram) の概念を提案し，suffix tree を用いることで rg の抽出を行う．またこの rg の特徴を上手く利用した文字列 id のリストのマージの新しい手法を提案する．

可変長 N-gram の先行研究である VGRAM, 文字列 id のマージに関する先行研究である，MergeOpt のとの比較実験では提案手法の方が，検索時間の点で優位であることが示せた．

2 問題定義

本稿では，数ある類似度のうち最もよく使われる，編集距離を対象とする．編集距離はある文字列を別の文字列へと変換するために行う編集操作の最小回数で表される．編集操作とは，挿入・削除．置換の 3 つの操作のことであり，それぞれの操作を行うごとに距離が 1 ずつ増えていく．

文字列 s_1,s_2 間の編集距離を $ed(s_1, s_2)$ で表す．クエリを Q, 文字列データの集合を S としたとき本稿で対象とする類似文字列とは，ある閾値 k を用いて次式を満たす $s \in S$ となる文字列 s を全て見つけることである．

$$ed(s, Q) \leq k \tag{1}$$

3 提案手法

本稿ではある文字列 s の長さを $|s|$ で表し，s の i 番目の文字を $s[i](i \leq i \leq |s|)$ で表す．また $s[i]$ から始まり $s[j](i \leq i < j \leq |s|)$ で終わる文字列 s の部分文字列を $s[i;j]$ で表す．

提案する手法は，(1) 使用する可変長 gram の選択，(2) 文字列 id のリストのマージ，の 2 つに分かれる．

可変長 gram の選択と索引付け

次の手順により, 使用する可変長の gram を行う．まず, gram の最短のノードを N_{min}，頻度の閾値として τ を定める．

1. データ中の文字列を全て一つの suffix tree に表現し，各ノードが何回出現したかを数える
2. 深さが N_{min} 以下のノードに対して，親のノードが頻度が閾値の τ を超え，そのノードの頻度が τ 以下となるノードの子ノードを枝刈りし，そのノードをマークする
3. 残ったノードを連結したもののうち，マークしたノードを含むもののみを可変長の gram とする

このようにして抽出された gram を rg とする．rg の長さを M であるとすると，rg の最後の文字を削除した，長さ $M-1$ の gram は頻出であると言えるが，rg をもつ文字列はデータ中で出現頻度は少なくなる．このため，この rg はその文字列を代表的に表す部分文字列となると言える．

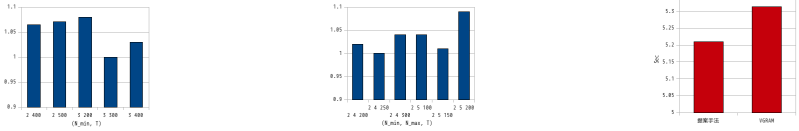
次に索引付けの手法について述べる．可変長の gram は tree 構造で保持しておく．まずは文字位置 $i = 1$ から始める．tree 上でルートノードから探索をし，それに連結するノードの中から $s[i]$ となる枝を探す．その枝が見つかったら，それに連結するノードの中から $s[i+1]$ となる枝を探す．探索に失敗するか，末端までたどりついたときに tree 上での探索をやめる．$s[j](i < j)$ の文字で探索が終了したとし，$s[j]$ が末端のノードであったなら部分文字列 $s[i;j]$ を索引語候補とする．$s[j]$ が末端のノードでなければ，部分文字列 $s[i;i+N_{min}]$ を索引語候補とする．そしてこの索引語候補が位置 i の時点で索引語と確定している gram の部分文字列となっていなければ，これを gram として確定する．以上の操作を $i = |s| - Nmin$ になるまで，i を 1 ずつ増やしていく．これにより得られた gram の集合をその文字列の索引語とする．

文字列 id リストのマージ

rg を用いた文字列リストのマージ手法について説明する．クエリと類似した文字列をデータセット中から gram ベースの索引付けを用いた転置索引を用いて検索するとき，長さが N の gram を索引語としたとすると，クエリ Q と編集距離が k 以下

† 東京大学大学院

‡ 国立情報学研究所



(a) 提案手法　(b) VGRAM　(c) 比較

図 1　パラメータとクエリ処理時間

のデータセット中の文字列 s の共有する gram の下限値 T は次式で与えられる [1].

$$T = \max(|s|, |Q|) - N + 1 - k \cdot N \quad (2)$$

一般的な手法では，クエリを gram に分割したときに，その各 gram を索引語とする転置索引中の各レコードの文字列 id のリストをマージしていく過程で出現回数が T 回以上となる文字列 id を探すことで解候補の削減を行う．

提案する手法では，rg を効果的に用いることでこのリストのマージを効率化する．簡単のために，クエリとの編集距離が 1 以内の文字列を探すこととする．

クエリ内に rg が含まれる

このとき次の 2 つの場合を考える．

1. rg に編集操作が行われていない
2. rg に編集操作が行われている

(1) のとき，rg を索引語とする文字列のリスト中に必ず解が存在する．そのため，このリスト内に含まれる文字列 id を残りのリスト内で二分探索を行い，$T - |rg(Q)|$ 回以上出現する文字列を解候補とする．ここで $|rg(Q)|$ はクエリ Q 中に存在する rg の個数を表す．

(2) のとき，rg は編集操作を受けているため，その他の gram のうち影響を受ける可能性があるのは rg と文字列を共有している gram のみである．rg 以外の gram 長は N_{min} であるので，このとき T_{rg} は次式のようになる．

$$T_{rg} = |G(Q)| - |rg| - (N_{min} - 1) \quad (3)$$

$|G(Q)|$ はクエリ Q 内の索引語 (gram) の総数である．よって rg 以外の gram を索引語とする文字列 id のリストをマージしていき，T_{rg} 回以上出現する文字列 id を MergeSkip[3] により検索する．

クエリ内に rg が含まれない

このとき，クエリの各 gram を索引語とする文字列 id のリストをマージするときには T 回以上出現する文字列 id を MergeSkip することで検索する．

4　評価実験

比較手法として，可変長 N-gram の先行研究である VGRAM[2] により可変長の N-gram を抽出し，MergeOpt により文字列 id のリストのマージをする手法を用いた．実験では，エントリー数が 69,069 個の英単語辞書を用いて実験を行った．単語長は平均 9.4 文字であった．全ての単語を用いて，Suffix Tree を用いた可変長 N-gram，VGRAM を生成した．また，全単語中から任意に選んだ 10,000 個の単語を編集距離 1 の文字列へと変換し，検索用のクエリとした．

まず，VGRAM，提案手法それぞれにおいて検索の速度が最も早くなるパラメータを求め，この値を用いて評価を行った．計測した時間は，クエリの索引語の分割からリストのマージに要した時間である．

図 1(a) に提案手法を用いて各パラメータを変化させたときの，クエリ処理に要する時間の変化を示し，図 1(b) に VGRAM で閾値を変化させたときの処理時間の変化を示す．それぞれの最適なパラメータのときのクエリ処理時間を比較すると，提案手法が優位であることが分かる (図 1(c))．

提案手法では，rg を用いることで一つのリスト内に含まれる文字列 id を削減することができた．このため結果としてリストのマージにかかる時間を削減することができた．

5　おわりに

編集距離を類似度に用いる類似文字列検索において，データセット中での出現頻度を考慮し，その文字列を代表する可変長 N-gram の抽出と，それを用いた文字列 id のリストのマージの高速化手法を提案した．実在のデータを用いた性能評価実験では，既存の可変長 N-gram で索引付けし，文字列 id のリストのマージに MergeOpt を用いた検索手法よりもよい結果を示すことができた．今回計測した時間は，リストのマージに要するもののみであった．gram ベースの問題の一つとして，索引付けに時間がかかることが指摘されている．今後は，索引付けの観点からも高速化を図りたい．

参考文献

[1] Sarawagi et al., Efficient set joins on similarity predicates. In SIGMOD, 2004.
[2] Li, et al., VGRAM: Improving Performance of Approximate Queries on String Collections Using Variable-Length Grams. In VLDB, 2007.
[3] Li et al., Efficient Merging and Filtering Algorithms for Approximate String Searches. In ICDE,2008.