УДК 681.51:57

А.Л. СИДОРЕНКО[1], С.А. РАКОВ[1], А.С. КУЛИК[2], А.Г. ЧУХРАЙ[2], А.Ю. ЗАВГОРОДНИЙ[2]

[1]Украинский центр оценивания качества образования, Харьков, Украина
[2]Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ»,Украина

МЕТОДЫ БЫСТРОГО ПОИСКА ПОХОЖИХ СТРОК

Проблема обнаружения строк, не являющихся дубликатами, но представляющих одну сущность реального мира, часто встречается в системах повышения качества данных информационных систем. Такие похожие строки могут появляться в результате ошибок при вводе данных или использования аббревиатур. Исследуемая в работе проблема также тесно связана с проблемой слияния баз данных и является сложно решаемой, если требования высокой точности и скорости получения решения выдвигаются одновременно. В данной работе предлагаются два высокоточных метода поиска похожих строк для случаев, когда возможность использования аббревиатур может либо учитываться, либо нет. Показаны результаты применения методов на реальных данных, подтверждающие целесообразность их в системах повышения качества данных информационных систем.

Ключевые слова: *качество данных, критерий похожести строк, метод поиска похожих строк.*

Введение

Постоянное усовершенствование информационных технологий привело к накоплению современными организациями больших объемов данных в электронном виде. Руководители организаций осознали важность систем поддержки принятия решений в различных сферах деятельности. Вместе с тем эффективность использования таких систем во многом зависит от качества накопленных информационными системами данных. Известно множество примеров материальных потерь. Так, например, Л. Инглиш в своей монографии [1] оценивает потери предприятий из-за некачественных данных в 10 – 25% от их дохода.

Для достижения высокой достоверности и полноты данных информационных систем необходимо применение специализированных методов и средств повышения качества данных. В свою очередь, подобные средства должны разрабатываться с учетом возможностей современных вычислительных средств, специфики ввода данных в информационных системах и психофизиологических возможностей человека-оператора. В данной статье мы рассматриваем одну из ключевых проблем в системах повышения качества данных – обнаружение пар похожих строк в момент, когда информация поступает в буфер хранилища данных из различных источников. В свою очередь, упомянутая проблема тесно связана с задачей известной в литературе как слияние и очистка данных (merge/purge problem) [2, 3], семантическая интеграция, слияние записей или идентификация объектов [4].

В неформальном виде рассматриваемую проблему можно представить следующим образом. Дан последовательный список строк. Необходимо найти все пары строк, предположительно, описывающих одну и ту же сущность реального мира. Сложность проблемы обуславливается сложностью определения похожести двух строк. Следует отметить, что задача поиска данных, предположительно, представляющих одну и ту же сущность реального мира, достаточно давно привлекает внимание исследователей. Так в работе [5] представлен метод SoundEx, позволяющий путем фонетического кодирования обнаруживать похожие по звучанию английские слова. В работах [6, 7] исходные строки разбиваются на q-граммы, и в качестве меры похожести двух строк принимается количество q-грамм, присутствующих в обеих строках. Наконец, особого внимания заслуживает подход, базирующийся на расстоянии Левенштейна. В рамках данного подхода похожими считаются строки, для которых количество операций редактирования необходимое для преобразования одной строки в другую не превышает некоторого допустимого порога, что наилучшим образом соответствует ошибкам, допускаемым операторами на этапе ввода данных [8].

Говоря о безусловной полезности методов, представленных в упомянутых выше работах, следует отметить и ряд недостатков. Прежде всего, часть методов достигают высокого быстродействия за счет снижения точности получаемых результатов. Так, например, метод, представленный в [3], не найдет пару похожих строк, если первый и последний

символ одной из строк будет искажен, метод, опубликованный в [9], не находит пары похожих строк поступивших из одного источника и т.д. Более того, ни один из перечисленных методов не учитывает широко распространенную ситуацию, когда операторы используют аббревиатуры и сокращения, и две похожие строки представляют полное наименование и сокращение. Данная статья посвящена созданию методов, которые бы позволяли избежать указанных недостатков.

1. Метод быстрого поиска похожих строк

Для поиска похожих строк в качестве базового критерия похожести было выбрано расстояние редактирования Левенштейна, достоинство которого относительно других критериев заключаются в независимости от предметной области, представляемой строками и соответствии наиболее распространенным ошибкам ввода данных, допускаемыми операторами. Представим формальную постановку задачи. Пусть дан набор строк $ST=\{st_1,st_2,...,st_n\}$. Требуется найти все пары строк $st_i,st_j \in ST$, для которых выполняется условие:

$$d(st_i,st_j) \le \lambda, \qquad (1)$$

где $d(st_i,st_j)$ – расстояние редактирования Левенштейна между строками st_i и st_j;

λ – некоторый порог похожести.

Используя наивный подход, мы должны проверить каждую пару строк из набора на соответствие условию (1). Алгоритмическая сложность такого подхода будет квадратичной $O(n^2)$. Очевидно, что при достаточно большом n такой метод поиска пар похожих строк становится неэффективным. Следовательно, необходим метод, позволяющий достичь тех же результатов за существенно меньшее время.

Предлагаемое решение состоит из двух шагов. Вначале из набора ST случайным образом выбираются k представителей $o_1,o_2,...,o_k$, $(k<n)$, которые в дальнейшем будут ассоциироваться с осями k-мерного евклидова пространства E^k. Затем каждому элементу $st_i \in ST$ ставится в соответствие точка k-мерного евклидова пространства $P(st_i)$, координаты которой равны расстояниям Левенштейна до осей, т.е. $P(st_i)_j = d(st_i,o_j), i=\overline{1,n}, j=\overline{1,k}$. На втором шаге расстояния Левенштейна рассчитываются только для строк, соответствующих близко расположенным точкам k-мерного евклидова пространства. Для получения необходимых условий похожести строк введем ряд утверждений.

Утверждение 1. Расстояние Левенштейна между любыми двумя строками не меньше абсолютного значения разности расстояний от них до третьей строки,

т.е. $\forall st_i,st_j,st_g \;\; d(st_i,st_j) \ge | d(st_i,st_g) - d(st_g,st_j) |$.

Доказательство данного утверждения следует непосредственно из неравенства треугольника, выполняющегося для метрики Левенштейна.

Утверждение 2. Если st_i,st_j – строки, расстояние Левенштейна между которыми не больше некоторого порога λ, то точки $P(st_i)$ и $P(st_j)$ пространства E^k, соответствующие исходным строкам, удалены в E^k друг от друга на расстояние не более чем $\lambda\sqrt{k}$, т.е.

$$\forall i \forall j \ne i \; d(st_i,st_j) \le \lambda : \; \rho(P(st_i),P(st_j)) \le \lambda\sqrt{k}.$$

Доказательство. По определению метрика пространства

$$E^k \;\; \rho(P(st_i),P(st_j)) = \sqrt{(P(st_i)_1 - P(st_j)_1)^2 + ... + (P(st_i)_k - P(st_j)_k)^2}.$$

Согласно утверждению 1

$$| d(st_i,o_1) - d(st_j,o_1) | \le d(st_i,st_j), \ldots,$$

$$| d(st_i,o_k) - d(st_j,o_k) | \le d(st_i,st_j).$$

Отсюда, транзитивно,

$$| P(st_i)_1 - P(st_j)_1 | \le \lambda, \ldots, | P(st_i)_k - P(st_j)_k | \le \lambda,$$

а значит,

$$\sqrt{(P(st_i)_1 - P(st_j)_1)^2 + ... + (P(st_i)_k - P(st_j)_k)^2} \le \lambda\sqrt{k}.$$

Утверждение 3. Если st_i,st_j – строки, расстояние Левенштейна между которыми не больше некоторого порога λ, то точка $P(st_j)$ размещается в E^k в пределах гиперкуба с центром в точке $P(st_i)$ и стороной 2λ.

Доказательство. Согласно утверждению 1 получаем следующую систему.

$$\begin{cases} | d(st_i,o_1) - d(st_j,o_1) | \le d(st_i,st_j); \\ | d(st_i,o_2) - d(st_j,o_2) | \le d(st_i,st_j); \\ \qquad ... \\ | d(st_i,o_k) - d(st_j,o_k) | \le d(st_i,st_j). \end{cases}$$

Следовательно,

$$\begin{cases} P(st_j)_1 \ge P(st_i)_1 - \lambda; \\ P(st_j)_1 \le P(st_i)_1 + \lambda; \\ \qquad ... \\ P(st_j)_k \ge P(st_i)_k - \lambda; \\ P(st_j)_k \le P(st_i)_k + \lambda. \end{cases} \qquad (2)$$

Геометрический смысл системы неравенств (2) представляет собой гиперкуб с центром в точке $P(st_i) = (d(st_i, o_1), d(st_i, o_2), ..., d(st_i, o_k))$ и стороной 2λ.

Утверждение 4. Если st_i, st_j – строки, расстояние Левенштейна между которыми не больше некоторого порога λ, то абсолютное значение разности расстояний от точек $P(st_i)$ и $P(st_j)$ до начала координат в E^k не превышает $\lambda\sqrt{k}$, т.е.

$$|\rho(P(st_i),0) - \rho(P(st_j),0)| \le \lambda\sqrt{k}.$$

Доказательство. Согласно свойству метрики евклидова пространства (неравенство треугольника) $|\rho(P(st_i),0) - \rho(P(st_j),0)| \le \rho(P(st_i),P(st_j))$. С другой стороны, согласно утверждению 2

$$\rho(P(st_i),P(st_j)) \le \lambda\sqrt{k}.$$

Отсюда

$$|\rho(P(st_i),0) - \rho(P(st_j),0)| \le \rho(P(st_i),P(st_j)) \le \lambda\sqrt{k}$$

и, следовательно, $|\rho(P(st_i),0) - \rho(P(st_j),0)| \le \lambda\sqrt{k}$.

Утверждение 5. Пусть $u, w \in R$ и $u, w > 0$. Тогда из $[u] \le w$ следует $[u] \le [w]$, где $[u],[w]$ – целые части чисел u и w соответственно.

Доказательство. Рассмотрим два случая. 1. $[u] \le w$ и $[u] = [w]$. Поскольку $[w] \le [w]$, то $[u] \le [w]$. 2. $[u] \le w$ и $[u] < [w]$. Отсюда, $[u] \le [w]$, что и требовалось доказать.

Утверждение 6. Пусть $u, v, w \in R$ и $u, v, w > 0$. Тогда из $|u - v| \le w$ следует $|[u] - [v]| \le [w] + 1$, где $[u],[v],[w]$ – целые части чисел u, v и w соответственно.

Доказательство. Рассмотрим случай, когда $u \ge v$. Тогда из $u \le w + v$ следует $[u] \le w + v$, поскольку $u \ge [u]$, а также $[u] \le w + [v] + 1$, так как $v < [v] + 1$. Согласно утверждению 5 из $[u] - [v] \le w + 1$ следует $[u] - [v] \le [w] + 1$. Для случая $v > u$, который рассматривается аналогично, получаем $[v] - [u] \le [w] + 1$. Обобщая оба случая, имеем $|[u] - [v]| \le [w] + 1$, что и требовалось доказать.

Утверждение 7. Если абсолютное значение разности длин строк st_i, st_j больше λ, то и расстояние Левенштейна между st_i, st_j больше λ.

Доказательство данного утверждения очевидно и следует из факта, что для того, чтобы превратить строку st_i в строку st_j, необходимо выполнить как минимум $\lambda + 1$ операцию удаления.

Сформулируем суть предлагаемого решения в виде последовательности действий.

1. На первом шаге после случайного выбора из набора ST k строк-осей и вычисления координат точек $P(st_i)$ также вычисляем расстояния в E^k от точек $P(st_i)$ до начала координат: $\rho(P(st_i),0) = \sqrt{P(st_i)_1^2 + P(st_i)_2^2 + ... + P(st_i)_k^2}$. Кроме того, образуем матрицу D распределения расстояний в E^k от точек $P(st_i)$ до начала координат. Для этого введем множество $\Psi = \{[\rho(P(st_1),0)],[\rho(P(st_2),0)],...,[\rho(P(st_n),0)]\} = \{\psi_1, \psi_2, ..., \psi_z\}, z \le n$, где $[\rho(P(st_i),0)]$ обозначает целую часть от $\rho(P(st_i),0)$. Поставим в соответствие каждому $\psi_i \in \Psi$ множество целых чисел (индексов исходных строк с расстоянием до начала координат равным ψ_i), т.е.

$$IND_i = \{ind_{i1}, ind_{i2}, ..., ind_{iw}\}.$$

При этом выполняется следующее условие $\forall q \in \{1,...,w\}\, ind_{iq} \in \{1,...,n\}, \rho(P(st_{ind_{iq}}),0) = \psi_i$. Теперь перейдем непосредственно к построению матрицы D, размерность которой равна

$$(\max(\Psi) - \min(\Psi) + 1) \times (\max\{|IND_1|,...,|IND_z|\}).$$

Воспользовавшись вспомогательными множествами IND_i, присвоим $D_{\psi_i q} = ind_{iq}$. Таким образом, строка с индексом ψ_i матрицы D содержит индексы исходных наименований, для которых целая часть расстояния в E^k до начала координат равна ψ_i.

2. Для каждой st_i, $i = \overline{1,n}$ находим строку матрицы D с индексом $[\rho(P(sm_i),0)]$. После этого согласно утверждениям 4 и 6, просматриваем ближайшие к ней строки матрицы D с индексами из множества

$$\Psi 1 = \{ [\rho(P(st_i),0)] - [\lambda\sqrt{k}] - 1,$$
$$[\rho(P(st_i),0)] - [\lambda\sqrt{k}], ..., [\rho(P(st_i),0)] - 1,$$
$$[\rho(P(st_i),0)], [\rho(P(st_i),0)] + 1, ..., [\rho(P(st_i),0)] + [\lambda\sqrt{k}],$$
$$[\rho(P(st_i),0)] + [\lambda\sqrt{k}] + 1\} = \{\psi 1_1, \psi 1_2, ..., \psi 1_v\},$$

причем $\Psi 1 \subset \Psi$, $v \le z$.

Затем при просмотре элементов строки матрицы D с индексом $\psi 1_t$, т.е. $D_{\psi 1_{tq}}$, производим дальнейшее отсеивание "кандидатов" в похожие наименования: во-первых, путем проверки условия, сформулированного в утверждении 7, и во-вторых, проверяя условие из утверждения 3: лежит ли точка $P(st_{D_{\psi 1_{tq}}})$ в гиперкубе, построенном с центром в точке $P(st_i)$ и стороной 2λ. Наконец, если $P(st_{D_{\psi 1_{tq}}})$ находится в пределах заданного гипер-

куба, то вычисляем расстояние Левенштейна между строками st_i и $st_{D_{\psi 1 tq}}$.

2. Поиск похожих строк, содержащих сокращения и аббревиатуры

2.1. Критерий похожести строк

Для формирования критерия похожести двух строк, который бы позволял обнаруживать различные аббревиатуры и сокращения, необходимо, основываясь на вербальных описаниях, дать формальные математические определения некоторых понятий.

Пусть $\{c_1, c_2, \ldots, c_a\}$ – конечное непустое множество символов алфавита Σ_c. Будем называть *словом* любую цепочку символов алфавита Σ_c. Пусть $\{delim_1, delim_2, \ldots, delim_b\}$ – конечное непустое множество символов алфавита Σ_{del}, таких, что если $x \in \Sigma_{del}$, то $x \notin \Sigma_c$. Тогда любую строку st полученную путем конкатенации символов алфавита $\Sigma = \Sigma_c \cup \Sigma_{del}$, можно представить как *словосочетание* следующим образом:

$$st = w_1 \bullet z_1 \bullet w_2 \bullet z_2 \bullet \ldots \bullet z_{k-1} \bullet w_k, \quad (3)$$

где $w_1, w_2, \ldots, w_k$ – слова входящие в строку st;

$z_1, z_2, \ldots, z_{k-1}$ – произвольные цепочки символов алфавита Σ_{del};

$\bullet$ – обозначение операции конкатенации двух цепочек.

Будем считать *сокращением* слова $w = c_1 \bullet c_2 \bullet \ldots \bullet c_m$ такую строку p, для которой выполняется следующее условие:

$$F(p,w) = S_2[p \in \{\varepsilon, \varepsilon \bullet c_1 \bullet z, \ldots, \varepsilon \bullet c_1 \bullet c_2 \bullet \ldots \bullet c_m \bullet z\}], \quad (4)$$

где ε – пустой символ,

$z \in \sum_{del}^0 \cup \sum_{del}^1$ т.е. цепочка над алфавитом Σ_{del} с длиной равной 0 или 1.

Понятие *аббревиатуры* определим следующим образом: строка a является аббревиатурой или сокращением словосочетания st тогда и только тогда, когда строку a можно представить в виде конкатенации сокращений слов словосочетания s т.е.

$$a = p_1 \bullet p_2 \bullet \ldots \bullet p_k, \ \forall p_i F(p_i, w_i). \quad (5)$$

Для создания критерия похожести строк следует учитывать, что как полное наименование, так и аббревиатуры и сокращения, сознательно используемые оператором, могут быть подвержены искажениям, вызванным ошибкой человека. Поэтому представляется целесообразным ввести понятие *расстояния редактирования аббревиатур* между двумя строками определяемого минимальным количеством операций вставки или удаления символа, необходимых для такого преобразования одной из строк, после которого она будет являться аббревиатурой или сокращением второй строки. Тогда *критерием похожести двух строк* должно стать условие, при котором определенное выше расстояние между двумя строками не превышает некоторый заданный порог.

Таким образом, формальная постановка задачи поиска похожих строк с учетом сокращений и аббревиатур будет выглядеть следующим образом.

Пусть дан набор строк $ST = \{st_1, st_2, \ldots, st_n\}$. Требуется найти все пары строк $st_i, st_j \in ST$ для которых выполняется условие:

$$d_a(st_i, st_j) \le \lambda, \quad (6)$$

где $d_a(st_i, st_j)$ – расстояние редактирования аббревиатур между строками st_i и st_j;

λ – некоторый порог похожести.

2.2. Расчет расстояния редактирования аббревиатур

Для расчета расстояния редактирования аббревиатур в данной работе используется следующий подход. На первом этапе для полного наименования (или более длинного) st_1 длиной r символов строим недетерминированный конечный автомат (НКА), который допускает цепочки являющиеся аббревиатурой или сокращением наименования в соответствии с (5):

$$A_e = (Q, \Sigma, \delta, q_0, F),$$

где $Q = \{q_0, q_1, q_2, \ldots, q_r\}$ – множество состояний такого автомата, причем, если автомат находится в q_i, $i = \overline{1,r}$ это можно интерпретировать следующим образом: "Цепочка, поступившая на вход автомата, является аббревиатурой или сокращением строки $st_1' = s_1 \bullet s_2 \bullet \ldots \bullet s_i$ и, следовательно, строки st_1";

Σ – конечное множество входных символов (алфавит $\Sigma_c \cup \Sigma_{del}$);

q_0 – начальное состояние;

$F = \{q_1, q_2, \ldots, q_r\}$ – множество допускающих состояний; δ – функция переходов автомата, полученная в соответствии со следующими правилами:

а) необходимо обеспечить переходы, по которым, если на автомат подается цепочка равная полному наименованию, попадаем в состояние q_l, т.е. $q_i \in \delta(q_{i-1}, s_r)$;

б) необходимо обеспечить переходы, предусматривающие варианты сокращения слов в соответствии с (4).

Пример такого НКА, построенного для словосочетания "Привет мир", представлен на рис. 1.

На втором этапе, путем модификации построенного ранее НКА, получим автомат, допускающий все цепочки, для которых расстояние редактирования аббревиатур не превышает некоторого λ:

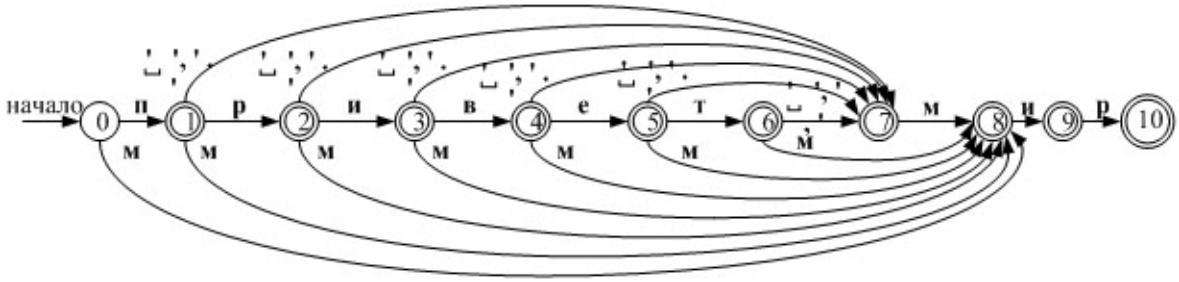

Рис. 1. Диаграмма переходов НКА, допускающего цепочки являющиеся сокращением или аббревиатурой словосочетания «привет мир»

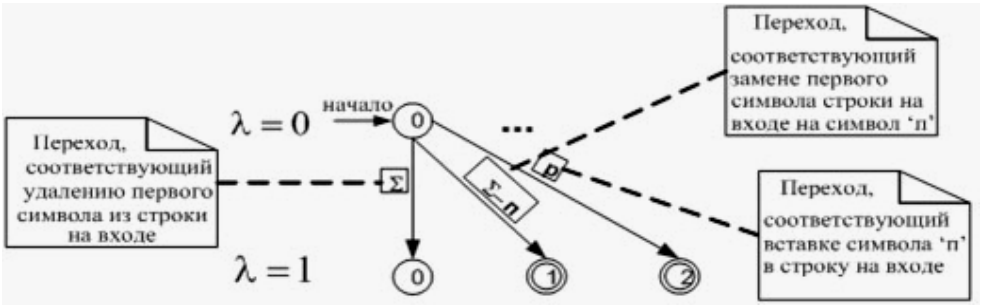

Рис. 2. Переходы, отражающие операции редактирования

A=(Q, Σ, δ, q_0, F), где Q={$q_{0,0}$, $q_{0,1}$, $q_{0,2}$, ..., $q_{0,r}$,..., $q_{\lambda,0}$, $q_{\lambda,1}$, $q_{\lambda,2}$, ..., $q_{\lambda,r}$} – множество состояний такого автомата, причем, если автомат находится в $q_{j,i}$, $j=\overline{0,\lambda}, i=\overline{1,r}$ это можно интерпретировать следующим образом: "Существует путь редактирования длиной в j операций, после выполнения которых, цепочка, поступившая на вход автомата, будет являться аббревиатурой или сокращением строки $st_1' = s_1 \bullet s_2 \bullet ... \bullet s_i$ и, следовательно, строки st_1". Соответственно если НКА находится во множестве состояний $States = \{q_{f1,s1}, q_{f2,s2}, ..., q_{fh,sh}\}$ то расстояние редактирования аббревиатур между исходной строкой s_1 и строкой, поданной на вход автомата, будет равным $d_a = \min(f1, f2, ..., fh)$; Σ – конечное множество входных символов (алфавит $\Sigma_c \cup \Sigma_{del}$); $q_{0,0}$ – начальное состояние; F={$q_{0,1}$, $q_{0,2}$, ..., $q_{0,r}$,..., $q_{\lambda,1}$, $q_{\lambda,2}$, ..., $q_{\lambda,r}$} – множество допускающих состояний; δ – функция переходов автомата, полученная в соответствии со следующими правилами: а) необходимо обеспечить переходы, аналогичные переходам автомата A_e между состояниями находящимися на одном уровне, т.е.

$$\exists\delta_{A_e}(q_i, s) = \{q_{x1}, ..., q_{xm}\}, i = \overline{0,r}, s \in \Sigma \Rightarrow$$
$$\Rightarrow \delta_A(q_{j,i}, s) \supseteq \{q_{j,x1}, ..., q_{j,xm}\}, \forall j = \overline{0,\lambda} \quad ;$$

б) необходимо обеспечить переходы, отражающие операции вставки, удаления и замены символов между состояниями находящимися на разных уровнях (см. рис.2).

Теперь, имея в распоряжении метод расчета расстояния редактирования аббревиатур, мы можем предложить «наивный» метод решения задачи поиска похожих пар строк в списке. В соответствии с «наивным» методом мы для каждой строки списка должны проверить, выполняется ли условие (4). Очевидно, что в связи с тем, что будет выполнено n×(n–1) таких проверок, при достаточно большом n алгоритм решения задачи будет чрезвычайно неэффективным по быстродействию. Поэтому необходимо создание такого метода, который бы значительно превышал по показателям быстродействия «наивный» метод, достигая при этом таких же результатов.

2.3. Метод поиска пар похожих строк

Подобно предыдущему методу предлагаемое решение состоит из двух шагов. На первом шаге все элементы набора ST отображаются в k -мерное Евклидово пространство E^k, с осями которого ассоциируются $o_1, o_2, ..., o_k$, $(k < n)$ – к выбранных случайным образом элементов набора ST, т.е. каждому элементу $st_i \in ST$ ставится в соответствие точка k -мерного Евклидова пространства $P(st_i)$, координаты которой равны простым расстояниям Левенштейна (операция замены символа трактуется как две операции) до осей: $P(st_i)_j = d_s(st_i, o_j), i = \overline{1,n}, j = \overline{1,k}$. Далее на следующем шаге рассматриваются лишь те пары строк, которые отвечают строго доказанным необходимым условиям похожести. Среди отличий предлагаемого метода от приведенного выше следует отметить, что в связи с отличием используемого критериев похожести строк полностью отличны

необходимые условия похожести. Поэтому, прежде чем приступить к изложению сути предлагаемого метода, сформулируем ряд строго доказанных математических утверждений.

Пусть ' – обозначение операции удаления из строки всех символов разделителей, т.е. s' – строка, получаемая из строки s путем удаления всех символов, принадлежащих алфавиту Σ_{del}.

Утверждение 8. Если строка st_1 является аббревиатурой или сокращением строки st_2, то наибольшая общая подпоследовательность (longest common subsequence) между строками st_1' и st_2' будет равна st_1'.

Доказательство данного утверждения очевидно и следует из определений наибольшей общей подпоследовательности и (5).

Утверждение 9. Если одну из строк st_1 или st_2, исказить одной операцией редактирования то величина MyLCS не увеличится более чем на 1, т.е.

$$\mathrm{MyLCS}(st_1^*, st_2^*) \le \mathrm{MyLCS}(st_1, st_2) + 1,$$

где $\mathrm{MyLCS}(x, y) = \min(|x'|, |y'|) - |\mathrm{lcs}(x', y')|$, st_1^*, st_2^* – значения исходных строк после произведенной операции редактирования.

Доказательство. Рассмотрим 3 случая.

а) Одна из строк искажена операцией замены символа. Тогда, если учесть тот факт, что операция замены символа при расчете простого расстояния Левенштейна интерпретируется как две операции редактирования, становится очевидным следующее неравенство: $d_s(st_1'^*, st_2'^*) \le d_s(st_1', st_2') + 2$. Используя широко известное соотношение (7):

$$d_s(x', y') = |x'| + |y'| - 2|\mathrm{lcs}(x', y')|, \qquad (7)$$

получим

$$|st_1'^*| + |st_2'^*| - 2|\mathrm{lcs}(st_1'^*, st_2'^*)| \le |st_1'| + |st_2'| - 2|\mathrm{lcs}(st_1', st_2')| + 2.$$

Тогда, т.к. при замене символа длина строки остается неизменной,

$$-2|\mathrm{lcs}(st_1'^*, st_2'^*)| \le -2|\mathrm{lcs}(st_1', st_2')| + 2,$$

теперь, разделив обе части неравенства на 2 и прибавив $\min(|st_1'|, |st_2'|) = \min(|st_1'^*|, |st_2'^*|)$, получим

$$\min(|st_1'^*|, |st_2'^*|) - \mathrm{lcs}(st_1'^*, st_2'^*) \le \min(|st_1'|, |st_2'|) - \mathrm{lcs}(st_1', st_2') + 1 \Rightarrow \mathrm{MyLCS}(st_1^*, st_2^*) \le \mathrm{MyLCS}(st_1, st_2) + 1.$$

б) Одна из строк искажена удалением символа. Тогда, исходя из определения функции d_s, получим $d_s(st_1'^*, st_2'^*) \le d_s(st_1', st_2') + 1$. Отсюда,

$$|st_1'^*| + |st_2'^*| - 2\mathrm{lcs}(st_1'^*, st_2'^*) \le |st_1'| + |st_2'| - 2\mathrm{lcs}(st_1', st_2') + 1,$$

следовательно,

$$-2\mathrm{lcs}(st_1'^*, st_2'^*) \le |st_1'| - |st_1'^*| + |st_2'| - |st_2'^*| - 2\mathrm{lcs}(st_1', st_2') + 1.$$

Исходя из условия, что только одна из строк была искажена, получим $|st_1'| - |st_1'^*| + |st_2'| - |st_2'^*| = 1$, $\min(|st_1'^*|, |st_2'^*|) \le \min(|st_1'|, |st_2'|)$. Тогда, составив систему неравенств

$$\begin{cases} -2\mathrm{lcs}(st_1'^*, st_2'^*) \le 1 - 2\mathrm{lcs}(st_1', st_2') + 1; \\ \min(|st_1'^*|, |st_2'^*|) \le \min(|st_1'|, |st_2'|), \end{cases}$$

разделив обе части первого неравенства на 2, и сложив левые и правые части неравенств получим $\mathrm{MyLCS}(st_1^*, st_2^*) \le \mathrm{MyLCS}(st_1, st_2) + 1$, что и требовалось доказать.

в) Одна из строк искажена вставкой символа. Для данного случая справедливы следующие отношения:

$$|st_1'| - |st_1'^*| + |st_2'| - |st_2'^*| = -1,$$

$$\min(|st_1'^*|, |st_2'^*|) \le \min(|st_1'|, |st_2'|) + 1.$$

Используя эти отношения и рассуждения аналогичные случаю б) не составляет труда доказать неравенство $\mathrm{MyLCS}(st_1^*, st_2^*) \le \mathrm{MyLCS}(st_1, st_2) + 1$.

Утверждение 10. Расстояние редактирования аббревиатур между строками st_1 и st_2 не превышает величину некоторого порога λ тогда и только тогда, когда величина $\mathrm{MyLCS}(st_1, st_2)$ также не превышает заданного порога, т.е. $d_a(st_1, st_2) \ge \mathrm{MyLCS}(st_1, st_2)$.

Доказательство. Воспользуемся индуктивным подходом, разбив доказательство на базис и индуктивный шаг:

Базис. Пусть для некоторых строк st_{10} и st_{20} выполняется $d_a(st_{10}, st_{20}) = 0$, т.е. одна из строк является аббревиатурой или сокращением другой. Тогда в соответствии с утверждением 8 и фактом, что длина строки, представляющей аббревиатуру, всегда меньшая либо равная длине строки, представляющей полное наименование, вытекающем из определения аббревиатуры (5), MyLCS=0. Следовательно, для $d_a=0$ утверждение справедливо.

Индукция. Предположим, что утверждение справедливо для некоторых строк st_{1n} и st_{2n} и $d_a(st_{1n}, st_{2n}) = n$, т.е

$$n \ge \mathrm{MyLCS}(st_{1n}, st_{2n}). \qquad (8)$$

Предположим также, что в результате искажения одной из строк одной операцией редактирования расстояние редактирования аббревиатур стало равным n+1. Тогда, в соответствии с принципом индукции, доказав, что из неравенства (8) следует неравенство $n+1 \ge \mathrm{MyLCS}(st_{1(n+1)}, st_{2(n+1)})$, мы докажем все утверждение в целом. Рассмотрим совместно неравенство (8) и неравенство, вытекающее из утверждения 10:

$$\begin{cases} n \ge MyLCS(s_{1n}, s_{2n}) \\ MyLCS(st_{1n}, st_{2n}) + 1 \ge MyLCS(st_{1(n+1)}, st_{2(n+1)}) \end{cases} \Rightarrow$$

$$\begin{cases} n+1 \ge MyLCS(s_{1n}, s_{2n}) + 1 \\ MyLCS(st_{1n}, st_{2n}) + 1 \ge MyLCS(st_{1(n+1)}, st_{2(n+1)}) \end{cases} \Rightarrow$$

$$n+1 \ge MyLCS(st_{1n}, st_{2n}) + 1 \ge$$
$$\ge MyLCS(st_{1(n+1)}, st_{2(n+1)}) \Rightarrow$$
$$n+1 \ge MyLCS(st_{1(n+1)}, st_{2(n+1)}),$$

что и требовалось доказать.

Утверждение 11. Расстояние редактирования аббревиатур между любыми двумя строками st_1 и st_2 не меньше половины разности между абсолютной разностью простых расстояний Левенштейна от st'_1 и st'_2 до третьей строки st'_3 и абсолютной разностью длин строк st'_1 и st'_2, т.е.

$$d_a(st_1, st_2) \ge \frac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| - \left|\left|st'_1\right| - \left|st'_2\right|\right|}{2}.$$

Доказательство. Поскольку простое расстояние Левенштейна является метрикой, для него справедливо неравенство треугольника, следовательно, расстояние между двумя точками не меньше абсолютной разности расстояний от этих точек до третьей точки: $d_s(st'_1, st'_2) \ge \left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right|$. Тогда, заменив $d_s(st'_1, st'_2)$ на правую часть равенства (7), получим

$$\left|st'_1\right| + \left|st'_2\right| - 2\left|lcs(st'_1, st'_2)\right| \ge \left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| \Rightarrow$$

$$\Rightarrow -\left|lcs(st'_1, st'_2)\right| \ge \frac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| - \left|st'_1\right| - \left|st'_2\right|}{2}.$$

Прибавим к обеим частям неравенства

$$\min(\left|st'_1\right|, \left|st'_2\right|) : \min(\left|st'_1\right|, \left|st'_2\right|) - \left|lcs(st'_1, st'_2)\right| \ge$$
$$\frac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| + 2\min(\left|st'_1\right|, \left|st'_2\right|) - \left|st'_1\right| - \left|st'_2\right|}{2}.$$

Отсюда, согласно определению MyLCS и свойствам функции min:

$$\begin{cases} MyLCS(st_1, st_2) \ge \dfrac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| + \left|st'_1\right| - \left|st'_2\right|}{2}, \\ \text{если } \left|st'_2\right| \ge \left|st'_1\right|; \\ MyLCS(st_1, st_2) \ge \dfrac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| + \left|st'_2\right| - \left|st'_1\right|}{2}, \\ \text{если } \left|st'_1\right| > \left|st'_2\right|, \end{cases}$$

следовательно,

$$MyLCS(st_1, st_2) \ge \frac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| - \left|\left|st'_1\right| - \left|st'_2\right|\right|}{2}.$$

Тогда, принимая во внимание полученное неравенство и неравенство из утверждения 11, получим

$$d_a(st_1, st_2) \ge \frac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| - \left|\left|st'_1\right| - \left|st'_2\right|\right|}{2},$$

что и требовалось доказать.

Утверждение 12. Если st_1, st_2 – строки, расстояние редактирования аббревиатур между которыми не больше некоторого порога λ, то точки $P(st_1)$ и $P(st_2)$ удалены в E^k друг от друга на расстояние не более $(2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)\sqrt{k}$.

Доказательство. По определению расстояния между двумя точками в евклидовом пространстве

$$\rho(P(st_1), P(st_2))^2 = \sum_{i=1}^{k} (d_s(st'_1, o'_i) - d_s(st'_1, o'_i))^2. \quad (9)$$

Согласно условию и утверждению 11:

$$\lambda \ge d_a(st_1, st_2) \ge \frac{\left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right| - \left|\left|st'_1\right| - \left|st'_2\right|\right|}{2},$$

отсюда

$$2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right| \ge \left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right|. \quad (10)$$

Тогда, учитывая что обе части неравенства всегда неотрицательны, возведем их в квадрат:

$$(2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)^2 \ge \left|d_s(st'_1, st'_3) - d_s(st'_2, st'_3)\right|^2,$$

тогда очевидно неравенство

$$(2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)^2 \ge (d_s(st'_1, st'_3) - d_s(st'_2, st'_3))^2. \quad (11)$$

Принимая во внимание (9) и (11) получим

$$\rho(P(st_1), P(st_2))^2 \le \sum_{i=1}^{k} (2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)^2 \Rightarrow$$

$$\rho(P(st_1), P(st_2))^2 \le (2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)^2 k$$

и, следовательно,

$$\rho(P(st_1), P(st_2)) \le (2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)\sqrt{k},$$

что и требовалось доказать.

Утверждение 13. Если st_1, st_2 – строки, расстояние редактирования аббревиатур между которыми не больше некоторого порога λ, то точка $P(st_2)$ размещается в E^k в пределах гиперкуба с центром в $P(st_1)$ и стороной $2(2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)$.

Доказательство. Согласно полученному ранее неравенству (10):

$$\begin{cases} \left|d_s(st'_1, o'_1) - d_s(st'_2, o'_1)\right| \le 2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right| \\ \ldots \\ \left|d_s(st'_1, o'_k) - d_s(st'_2, o'_k)\right| \le 2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right| \end{cases} \Rightarrow$$

$$\begin{cases} P(st_2)_1 \ge P(st_1)_1 - 2\lambda - \left|\left|st'_1\right| - \left|st'_2\right|\right| \\ P(st_2)_1 \le P(st_1)_1 + 2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right| \\ \ldots \\ P(st_2)_k \ge P(st_1)_k - 2\lambda - \left|\left|st'_1\right| - \left|st'_2\right|\right| \\ P(st_2)_k \le P(st_1)_k + 2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right| \end{cases} \quad (12)$$

Геометрический смысл системы неравенств (12) представляет собой гиперкуб с центром в точке $P(st_1)$ и стороной $2(2\lambda + \left|\left|st'_1\right| - \left|st'_2\right|\right|)$.

Утверждение 14. Если st_1, st_2 – строки, расстояние редактирования аббревиатур между которыми не больше некоторого порога λ, то абсолютное значение разности расстояний от точек $P(st_1)$ и $P(st_2)$ до начала координат в E^k не превышает

$$(2\lambda + \left\|st_1'\right| - \left|st_2'\right\|)\sqrt{k}\text{, т.е.}$$

$$|\rho(P(st_i),0) - \rho(P(st_j),0)| \le (2\lambda + \left\|st_1'\right| - \left|st_2'\right\|)\sqrt{k}\,.$$

Доказательство данного утверждения следует непосредственно из неравенства треугольника и утверждения 12.

Теперь опишем подробнее суть выполняемых действий на каждом из этапов предлагаемого решения:

1. Как уже было сказано выше, на первом этапе происходит отображение исходного набора ST в k-мерное евклидово пространство E^k. Здесь следует лишь отметить, что структура данных, используемая для хранения точек пространства, должна обеспечивать быстрый последовательный доступ ко всем точкам, для которых заданы значения расстояния до начала координат и длины исходной строки из набора ST.

2. На втором этапе для каждой точки $P(st_i)$, варьируя значения переменной len от минимального значения длины строки до максимального, в соответствии с утверждением 14, просматриваем точки, для которых расстояние до начала координат находится в диапазоне $[\rho(P(st_i),0) - (2\lambda + \left\|st_i'\right| - len\right|)\sqrt{k};$ $\rho(P(st_i),0) + (2\lambda + \left\|st_i'\right| - len\right|)\sqrt{k}]$. Каждая рассматриваемая точка, в свою очередь, проверяется в соответствии с утверждением 13 на попадание в пределы гиперкуба с центром в точке $P(st_i)$. Далее для строк, соответствующих точкам, оказавшихся в пределах гиперкуба, рассчитывается величина MyLCS. И, наконец, в соответствии с утверждением 10, только в том случае, если MyLCS не превышает порогового значения, производится "дорогое" вычисление расстояния редактирования аббревиатур.

3. Экспериментальные исследования полученных методов

Экспериментальные исследования описанных методов были проведены на основе универсального отношения, содержащего информацию об абитуриентах, поступавших на факультет систем управления летательными аппаратами университета «ХАИ» в 2008 году. Количество записей в списке составило 194, а атрибутов 34. Таким образом, было выполнено 34 эксперимента для 34 списков из 194 строк. Все эксперименты проводились на персональном компьютере с процессором CELERON 566 МГц и 196 Мб ОЗУ. Операционная система – Windows NT 4 Server, компилятор – Borland Delphi 6.

Для таких атрибутов, как фамилия, имя, оценка по физике и др., в которых операторы не использовали аббревиатур, использовался первый из предложенных методов. Были выявлены такие похожие строки как «Глушковський», «Глушковский» (d=1); «Авионика», «Авіоніка» (d=2);. «Першотравневий», «Пешротравнеий» (d=2). Оценки быстродействия метода позволили утверждать, что предлагаемый метод позволяет получить результаты в среднем в 15,2 раза быстрее, чем наивный метод, заключающийся в оценке всех возможных пар строк.

Похожие строки, представляющие такие атрибуты, как область, наименование оконченной школы, тип населенного пункта выявлялись с помощью второго метода из-за регулярного использования операторами аббревиатур в этих атрибутах. Так применение предложенного метода позволило выявить, например, такие пары строк как «АРК Крим», «Кримська» (d_a=2); «Полтавская», «Полтавська» (d_a=2);. «загально освітня школа №43», «ЗОШ №43» (d_a=2). Следует отметить, что несмотря на существенно меньшее быстродействие второго метода, большая часть из обнаруженных методом пар не находится при использовании других критериев.

Заключение

Таким образом, получены новые методы поиска похожих строк в наборе. Первый из предложенных методов базируется на известном расстоянии Левенштейна, однако, в отличие от известных методов позволяет достичь высокого быстродействия, находя при этом все пары похожих строк. Для случая, когда строки могут содержать сокращения и полные наименования, новый критерий, позволяющий находить строки, представляющие одну и ту же сущность реального мира и искаженные относительно друг друга ошибками оператора и использованием сокращений и аббревиатур. Представленные необходимые условия похожести строк позволяют ограничить область поиска и, следовательно, получить эффективные методы. Эффективность предлагаемых методов подтверждается проведенными экспериментами.

Литература

1. English L. Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits / L. English. – New York: John Wiley & Sons, 1999. – 544 p.

2. Hernandez M.A. The merge/purge problem for large database / M.A. Hernandez, J.S. Stolfo // ACM SIGMOD International Conference on Management of Data, 21-23 May 1995. – P. 127-138.

3. Hernandez M.A. Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem / M.A. Hernandez, J.S. Stolfo // Journal of Data Mining and Knowledge Discovery. – 1998. – Vol. 2. – P. 9-37.

4. Maletic J. Data Cleansing: Beyond Integrity Analysis / J. Maletic, A. Marcus // The Conference on Information Quality (IQ2000). Boston, 20-22 Oct. 2000. – P. 200-209.

5. Кнут Д. Искусство программирования для ЭВМ: В 3 т. / Д. Кнут. – М.: Мир, 1978. – Т. 3. Сортировка и поиск. – 844 с.

6. Approximate string joins in a database (almost) for free / L. Gravano, G. Panagiotis, H. Jagadish, N. Koudas, S. Muthukrishnan, D. Srivastava // Procee dings of the VLDB Conference. 11-14 Sept., 2001. – P. 491-500.

7. Baeza-Yates R. A practical index for text retrieval allowing errors / R. Baeza-Yates, G. Navarro // Proceedings of the XXIII Latin American Conference on Informatics (CLEI'97) 14-15 Nov. 1997. – P. 273-282.

8. Информационно-аналитические модели управления техническими высшими учебными заведениями / А.Н. Гуржий, В.С. Кривцов, А.С. Кулик и др. – Х.: Нац. аэрокосм. ун-т «Харьк. авиац. ин-т», 2004. – 387 с.

9. Jin L. Efficient Similarity String Joins in Large Data Sets / L. Jin, C. Li, S. Mehrotra. – Technical Report, University of California, Department of Information and Computer Science, Feb., 2002. – 26 p.

Поступила в редакцию 12.11.2008

Рецензент: д-р техн. наук, проф., зав. каф. информатики А.Л. Ерохин, Харьковский национальный университет внутренних дел, Харьков.

МЕТОДИ БИСТРОГО ПОШУКУ СХОЖИХ РЯДКІВ

О.Л. Сидоренко, С.А. Раков, А.С. Кулік, А.Г. Чухрай, А.Ю. Завгородній

Проблема знаходження рядків, що не є дублікатами, але представляють одну сутність реального світу часто зустрічається у системах підвищення якості даних інформаційних систем. Такі схожі рядки можуть з'являтися у результаті помилок при введенні даних або використання абревіатур. Проблема, що досліджується у роботі також тісно пов'язана з проблемою злиття баз даних і є важко вирішуваною, коли вимоги до високої точності та швидкості отримання результатів ставляться одночасно. В дані роботі пропонується два високоточних метода пошуку схожих рядків для випадків, коли можливість використання абревіатур враховується або ні. Наведені експериментальні результати, що підтверджують корисність використання створених методів у системах підвищення якості даних інформаційних систем.

Ключові слова: якість даних, критерій схожості рядків, метод пошуку схожих рядків.

QUICK SIMILAR STRING SEARCHING METHODS

A.L. Sidorenko, S.A. Rakov, A.S. Kulik, A.G. Chukray, A.Yu. Zavgorodniy

The problem of detecting strings that are not exact duplicates but are concerning the same real-world entity is frequently encountered in management information system data cleansing applications. Such similar strings may appear in databases due to data entry errors and using various abbreviations. The problem we study is closely connected with merge/purge problem and is difficult to solve both in scale and in accuracy. In this paper, we propose two accurate methods for detecting similar string, when the possibility of using abbreviations is taken into account or not. Results of tests on real-world data are shown. This results show high accuracy and performance for both methods and confirm that our methods are useful in data cleansing systems.

Key words: data quality, string similarity criterion, similar string search method.

Сидоренко Александр Леонидович – доктор соц. наук, профессор, чл.-корр. Академии педагогических наук Украины, директор Украинского центра оценивания качества образования, Харьков, Украина, e-mail: org@testportal.com.ua.

Раков Сергей Анатольевич – доктор пед. наук, профессор, зам. директора Украинского центра оценивания качества образования, Харьков, Украина, e-mail: Rakov_S@ukr.net.

Кулик Анатолий Степанович – доктор техн. наук, профессор, зав. кафедры систем управления летательными аппаратами Национального аэрокосмического университета им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: kulik @d3.khai.edu.

Чухрай Андрей Григорьевич – канд. техн. наук, доцент кафедры систем управления летательными аппаратами Национального аэрокосмического университета им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: chukhray@d3.khai.edu.

Завгородний Андрей Юрьевич – канд. техн. наук, ассистент кафедры систем управления летательными аппаратами Национального аэрокосмического университета им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: boband@d3.khai.edu.