УДК 004.6
doi:10.15217/issn1684-8853.2015.2.76

КОНТЕКСТНО ЗАВИСИМЫЙ СПОСОБ ПОИСКА НЕЧЕТКИХ ДУБЛИКАТОВ В РЕЛЯЦИОННЫХ БАЗАХ ДАННЫХ

С. В. Тарасов[а], *ведущий инженер исследований и разработки*
В. В. Бураков[б], *доктор техн. наук, профессор*
[а]*Компания Bel Air Informatique, Courtaboeuf Cedex, Франция*
[б]*Санкт-Петербургский государственный университет аэрокосмического приборостроения, Санкт-Петербург, РФ*

Постановка проблемы: *одной из важных проблем в области управления данными является их неполное (нечеткое) дублирование, ведущее к снижению качества, в частности к ошибочной интерпретации информационной системой одного и того же объекта как нескольких разных. Реляционная модель данных, а также промышленные СУБД на основе реляционной модели, позволяют исключить ситуации полного дублирования данных, но не имеют механизмов для распознавания и предотвращения появления нечетких дубликатов. Целью работы является разработка такого способа обнаружения нечетких дубликатов, который мог бы быть реализован в реляционной модели данных и промышленной реляционной СУБД.* ***Результаты:*** *рассмотрена общая для информационных систем проблема нечеткого дублирования, предложены пути внесения смысловой дублирующей информации в реляционную базу данных. Определено, что для решения проблемы неполного дублирования следует использовать механизмы нечеткого сравнения строк с учетом их семантики. Приведен пример практической реализации способа для СУБД PostgreSQL с использованием реляционных механизмов обработки данных.* ***Практическая значимость:*** *разработанный способ позволяет автоматически обнаруживать дубликаты, исключив вмешательство человека-оператора, и тем самым повысить качество данных информационной системы. Пример практической реализации для промышленной СУБД позволяет непосредственно использовать предложенный способ в инженерной практике разработки информационных систем. Данный способ также был использован авторами при разработке коммерческой автоматизированной информационной системы.*

Ключевые слова *— нечеткие дубликаты, смысловые дубликаты, метод n-грамм, реляционная база данных, очистка данных, качество данных.*

Введение

Стандарт [1] определяет качество данных как уровень совокупности присущих объекту характеристик, отвечающий установленным требованиям. «Присущий» в противовес «присвоенному» означает существующий в чем-то как постоянная характеристика объекта. Термин «качество» может применяться с такими прилагательными, как низкий, плохой, годный, хороший и превосходный.

Предлагаемые стандартом оценки уровня качества, таким образом, являются нечеткими («низкий», «годный») и зависят от субъективной оценки данных потребителями. Возникает необходимость вводить в процесс обеспечения качества данных объективные показатели, позволяющие производить независимые оценки и сравнения.

Одним из показателей низкого качества данных является их дублирование, ведущее в итоге к ошибочной интерпретации одного и того же объекта как нескольких разных. Можно выделить два основных типа дублирования атрибутов: имеющих жестко заданную структуру (формат) содержания и не имеющих таковой, т. е. неполно структурированных.

В первом случае примером могут служить различные коды классификаторов или используемые в качестве ключевых атрибутов поиска идентификаторы, такие как номера телефонов, ИНН и т. п. Проблема решается стандартным ограничением уникальности значения атрибута в соответствующей колонке таблицы: поиск дубликатов производится системой управления базой данных (СУБД) по точному совпадению значения.

Во втором случае речь идет о разнообразных именах собственных и названиях, используемых для идентификации, таких как антропонимы, топонимы, названия предприятий, медикаментов, почтовые адреса и т. д. Не представляется возможным использовать стандартные ограничения целостности, предоставляемые реляционной моделью и соответствующими СУБД. Также практически невозможно использовать словари-справочники данных по следующим причинам [2]:

— на текущий момент существует огромное число подобных имен;

— непрерывно порождаются новые имена.

Размер географического справочника адресов для международной рассылки будет исчисляться многими миллионами записей, требующих регулярной поддержки в актуальном состоянии. Структурирование антропонима в общем случае невозможно, так как количество и порядок составляющих полного имени человека может зависеть не только от культуры страны происхождения и степени псевдонимизации, но и от общественного статуса.

Для решения вышеназванных проблем был разработан подход, целями которого являлись:

— использование особенностей реляционной модели;

— введение в рассмотрение объективных численных характеристик качества информации по критерию ее дублирования;

— описание ключевых этапов процесса по обеспечению качества на основе введенных характеристик.

Метод n-грамм и нечеткое сравнение строк

Задача поиска дубликатов может быть решена при помощи алгоритмов нечеткого сравнения строк. Известные способы вычисления дистанции, такие как EDIT[1] или LCS[2], имеют существенный недостаток: они нечувствительны к контексту [3, 4]. Например, LCS для пар названий лекарств «cardura – benadrol» и «osmitrol — benadrol» дает одинаковое значение, равное 3.

Применение метода n-грамм [2, 3, 5, 6] в сочетании с коэффициентом Дайса[3] дает лучшие результаты при сравнении отдельных слов и простых словосочетаний. Коэффициент вычисляется следующим образом [3]:

$$\text{dice}(X, Y) = 2 \times (|\text{n-grams}(X) \cap \cap \text{n-grams}(Y)|)/(|\text{n-grams}(X)| + |\text{n-grams}(Y)|).$$

Здесь X, Y — строки; n-grams(X) — множество грамм строки X; |n-grams(X)| — мощность множества.

Для тех же самых пар названий лекарств из примера LCS при разбиении на триграммы коэффициент будет равен нулю в первом случае и 0,17 во втором.

Но при сравнении предложений и фраз начинают проявляться смысловые ошибки. Например, если информация о контактном лице сравнивается в виде строк (табл. 1), то общий коэффициент схожести может быть высоким, тогда как по некоторым смысловым единицам фразы схожесть низкая.

Метод n-грамм предлагает решать эту проблему использованием вероятностей появлений грамм в соответствующих позициях фразы. Однако определение таблиц таких вероятностей не только представляет собой объемный труд, но и является зависимым от естественного языка [2, 5, 6].

[1] Дистанция редактирования; другое название — дистанция Левенштейна.

[2] От англ. longest common subsequence — нахождение длины наибольшей общей подпоследовательности (подстроки).

[3] От англ. Dice coefficient или Sørensen — Dice index в соответствии с именами ученых; коэффициент схожести двух образцов.

■ ***Таблица 1.*** Пример ошибки нахождения дубликата без учета семантики

Сравнение	Строка 1	Строка 2	Схожесть
Контактное лицо (фраза целиком)	«Иванов и партнеры», В. И. Петров, Энск, ул. Строителей 14	«Петров и партнеры», И. В. Иванов, Энск, ул. Строителей 14	0,91
Атрибуты контактного лица (составной ключ)	Иванов и партнеры	Петров и партнеры	0,73
	В. И. Петров	И. В. Иванов	0,3
	Энск	Энск	1
	ул. Строителей 14	ул. Строителей 14	1

В рамках реляционной базы данных информация уже структурирована в соответствии не только с языковыми особенностями, но, в первую очередь, со спецификой предметной области. Каждый объект, кроме своего первичного ключа, может иметь несколько составных ключей-идентификаторов, состоящих в том числе из текстовых атрибутов. Тогда интегральная оценка схожести составных идентификаторов объектов в базе данных может проводиться с учетом смыслового веса значений входящих в них отдельных атрибутов.

Пусть идентификатор I_k состоит из атрибутов $A_{k1}, A_{k2}, ..., A_{kn}$. Введем для каждого атрибута его смысловой вес w_{j1}, w_{j2}, ..., w_{jn} в диапазоне (0..1]. Очевидно, $w_i > 0$, иначе атрибут A_i исключается из рассмотрения.

Тогда взвешенная схожесть sim$w$ двух идентификаторов I_{k1} и I_{k2} вычисляется по формуле

$$\text{sim}w(I_{k1}, I_{k2}) = \sum_{i=1}^{N} \text{dice}(A_{i1}, A_{i2})(w_{i1} / N).$$

Порядок необходимых действий для поиска дубликатов будет следующим.

1. Определение атрибутов, идентифицирующих объект.

2. Определение весовых коэффициентов для атрибутов в составе идентификатора.

3. Вычисление взвешенной схожести всех пар идентификаторов. Данный показатель будет лежать в интервале [0..1], где 0 — полное несовпадение, 1 — полное совпадение.

4. Экспериментальным путем для тестового массива данных определяется нижний порог взвешенной схожести, за которым количество ошибок распознавания дубликатов становится неприемлемым. Назовем это значение порогом автоматической обработки и обозначим как $П_а$.

Особенности применения способа в автоматизированной информационной системе

Основными путями внесения и изменения информации являются:

— непосредственный ввод пользователями;

— импорт данных из внешних источников.

При пользовательском вводе требуется обеспечить минимальное время отклика системы, поэтому используемый на этом этапе алгоритм должен работать не столько точно, сколько предельно быстро. При этом параметр Π_a для данной операции может быть изменен в соответствии с требованиями по скорости поиска. Так как система распознавания не может предоставить 100%-ю точность, пользователь должен также иметь возможность игнорировать подсказку системы и ввести данные. При таких условиях в базу данных неизбежно будет попадать часть некачественной информации, которая должна быть обнаружена в дальнейшем. Задачу выявления и устранения дубликатов в информационной системе можно разбить на три этапа:

1) первичное выявление дубликатов на уровне ввода информации пользователями и определение их отклонения, если $\mathrm{sim}w > \Pi_a$;

2) выявление дубликатов путем сравнения и анализа уже введенных данных в соответствии с заданным Π_a и автоматическое удаление дублирующей информации, если $\mathrm{sim}w > \Pi_a$;

3) анализ и обработка человеком результатов п. 2, которые не могут быть обработаны автоматически, т. е. $\mathrm{sim}w < \Pi_a$.

Следует упомянуть о недостатке метода n-грамм, который может оказаться существенным для СУБД: большой размер производного множества n-грамм относительно словаря грамм. При заданном количестве символов N для каждой строки будет сформировано M словарных элементов:

$$M = L - N + 1,$$

где L — длина строки, $L \geq N$ (при $L < N$, $M = 0$ и поиск невозможен).

Максимальное количество элементов словаря грамм зависит от N и мощности алфавита m, если считать все комбинации из N букв допустимыми, и равно соответствующему количеству размещений с повторениями:

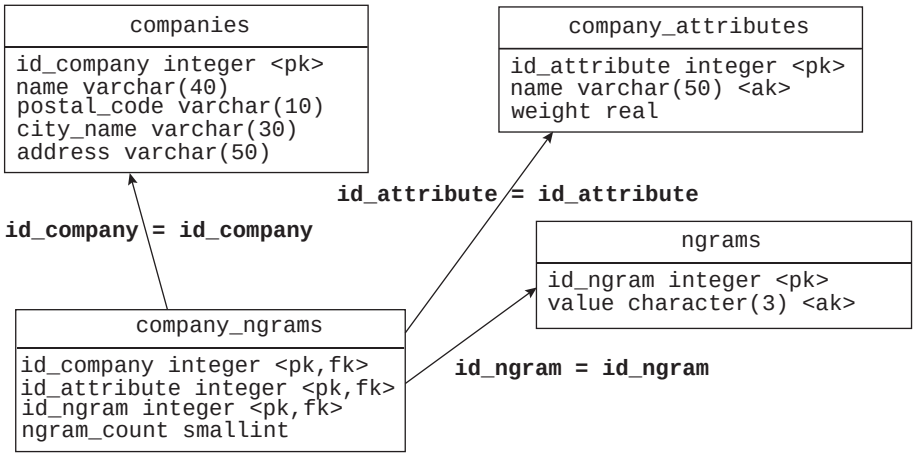
$$N_m^A = m^N.$$

Например, для русского языка при алфавите из 33 букв количество n-грамм при $N = 3$ равно 35 937. Однако, как показывает практика, в реальных текстах реализуется не более 25–30 % n-грамм от общего допустимого их числа [6], т. е. не более 11 000. Если имеется база данных на 50 000 абонентов, средняя длина значений атрибутов составного идентификатора равна 50 символам, то мощность множества триграмм будет составлять примерно $50\ 000 \cdot (50 - 3) = 2\ 350\ 000$ элементов. При этом количество уникальных элементов словаря грамм не будет превосходить теоретический максимум в 35 937 или практический в 11 000. Быстрый поиск по такому множеству при использовании стандартных средств реляционной СУБД в виде индексов будет затруднен, так как избирательность этих индексов мала. Решением проблемы при большом количестве данных для словаря будет увеличение N.

Пример использования метода n-грамм

Рассмотрим пример системы распознавания дубликатов в списке компаний для открытой и свободно распространяемой СУБД PostgreSQL 9. Пусть компания характеризуется следующим набором атрибутов (табл. 2).

В случае триграмм схема данных реализации нечеткого поиска компаний в рамках реляционной модели может выглядеть следующим образом (рисунок).



■ Схема данных реализации метода n-грамм для компаний

■ *Таблица 2*. Атрибуты сущности «Компания»

Атрибут	Тип
Название	Строка до 60 символов
Почтовый индекс	Строка до 20 символов
Населенный пункт (город)	Строка до 40 символов
Адрес	Строка до 80 символов

Средства СУБД PostgreSQL позволяют реализовать разбиение строки на граммы в виде пользовательской функции, возвращающей множество подстрок.

Листинг 1

```
CREATE OR REPLACE FUNCTION str_to_ngrams(n int,
s text)
RETURNS SETOF varchar(10)
AS $$
DECLARE i int;
        str_limit int;
BEGIN
  str_limit := char_length(s) - n + 1;
  FOR i IN 1..str_limit LOOP
    RETURN NEXT substring(s, i, n);
  END LOOP;
  RETURN;
END;
$$
LANGUAGE plpgsql;
```

Тогда для инициализации таблицы грамм компаний может быть использован следующий SQL-запрос.

Листинг 2

```
INSERT INTO companies_ngrams(id_company, id_
attribute, ngram, ngram_count)
SELECT
  id_company, id_attribute, ngram, count(*)
FROM (
SELECT
  c.id_company,
  ca.id_attribute,
  str_to_ngrams(3, c.name) AS ngram
FROM companies c CROSS JOIN company_attributes
ca
WHERE ca.name = 'name'
UNION ALL
SELECT
  c.id_company,
  ca.id_attribute,
  str_to_ngrams(3, c.postal_code) AS ngram
FROM companies c CROSS JOIN company_attributes
ca
WHERE ca.name = 'postal_code'
UNION ALL
SELECT
  c.id_company,
  ca.id_attribute,
  str_to_ngrams(3, c.city_name) AS ngram
FROM companies c CROSS JOIN company_attributes ca
WHERE ca.name = 'city_name'
UNION ALL
SELECT
  c.id_company,
  ca.id_attribute,
  str_to_ngrams(3, c.address) AS ngram
FROM companies c CROSS JOIN company_attributes ca
WHERE ca.name = 'address'
) ngrams
GROUP BY id_company, id_attribute, ngram
```

Для нечеткого поиска может использоваться следующий запрос, выдающий список найденных значений в порядке убывания их взвешенной схожести.

Листинг 3

```
WITH to_find(id_attribute, ngram)
AS (
  SELECT
    id_attribute,
    str_to_ngrams(3, 'НПО Электрон') AS ngram
  FROM company_attributes
  WHERE name = 'name'
  UNION ALL
  SELECT
    id_attribute,
    str_to_ngrams(3, '123456') AS ngram
  FROM company_attributes
  WHERE name = 'postal_code'
  UNION ALL
  SELECT
    id_attribute,
    str_to_ngrams(3, 'Энск') AS ngram
  FROM company_attributes
  WHERE name = 'city_name'
  UNION ALL
  SELECT
    id_attribute,
   str_to_ngrams(3, 'Строительный проезд, 15')
AS ngram
  FROM company_attributes
  WHERE name = 'address'
),
to_find_count(id_attribute, ngram_count)
AS (
  SELECT id_attribute, count(*)
  FROM to_find
  GROUP BY id_attribute
),
src_count(id_company,  id_attribute,  ngram_
count)
AS (
  SELECT id_company, id_attribute, sum(ngram_
count)
  FROM companies_ngrams
  GROUP BY id_company, id_attribute
),
```

```
   dice_intersection(id_company,    id_attribute,
value)
   AS (
        SELECT   cn.id_company,   cn.id_attribute,
count(ngram_count)
     FROM
       companies_ngrams cn
       INNER JOIN to_find
        ON cn.id_attribute = to_find.id_attribute
            AND cn.ngram = to_find.ngram
     GROUP BY cn.id_company, cn.id_attribute
   ),
   dice(id_company, id_attribute, value)
   AS (
     SELECT di.id_company, di.id_attribute,
       cast(2 * di.value AS float) /
        (src_count.ngram_count + to_find_count.
ngram_count)
     FROM
       dice_intersection di
       INNER JOIN src_count
         ON di.id_company = src_count.id_company
               AND di.id_attribute = src_count.
id_attribute
       INNER JOIN to_find_count
           ON di.id_attribute = to_find_count.
id_attribute
   ),
   dicew(id_company, value)
   AS (
     SELECT dice.id_company,
       sum(dice.value * ca.weight) /
       (SELECT count(1) FROM company_attributes)
     FROM
       dice
       INNER JOIN company_attributes ca
         ON dice.id_attribute = ca.id_attribute
     GROUP BY id_company
   )
   SELECT * FROM dicew
   ORDER BY value DESC
```

Заключение

В статье рассмотрены определения качества данных, проблемы снижения качества при наличии дублирующей информации, приведены существующие способы нечеткого сопоставления строк и их характеристики. Предложен способ обнаружения дублирующей информации на основе методов нечеткого сравнения текста. С использованием существующих методов нечеткого сравнения текста способ был доработан для применения в рамках реляционной модели и СУБД с учетом семантики его элементов. Приведен практический пример реализации способа на базе свободно распространяемой СУБД PostgreSQL. Предложенный способ может быть использован для реализации соответствующей встроенной функциональности в промышленных СУБД.

Литература

1. **ГОСТ** Р ИСО 8000-102-2011. Качество данных. Часть 102. Основные данные. Обмен данными характеристик. Словарь. — М.: Изд-во стандартов, 2012. — 16 с.
2. **Нехай И. В.** Применение n-грамм и других статистик уровня символов и слов для семантической классификации незнакомых собственных имен // Компьютерная лингвистика и интеллектуальные технологии: По материалам конференции «Диалог». — М.: Изд-во РГГУ, 2012. Вып. 11(18). Т. 1. С. 477–489.
3. **Мазов Н. А.** N-граммные методы обработки текстовой информации/ОИГГМ СО РАН. — Новосибирск, 1995. — 180 с.
4. **Гудков В. Ю., Гудкова Е. Ф.** N-граммы в лингвистике // Вестник Челябинского гос. ун-та. 2011. № 24(239). С. 69–71.
5. **Kondrak G.** N-Gram Similarity and Distance/ University of Alberta, Department of Computing Science, Edmonton, AB, T6G 2E8, Canada, 2005. — P. 115–126.
6. **Зеленков Ю. Г., Сегалович И. В.** Сравнительный анализ методов определения нечетких дубликатов для Web-документов // Электронные библиотеки: перспективные методы и технологии, электронные коллекции: тр. 9-й Всерос. науч. конф. RCDL, 2007. С. 166–174.

UDC 004.6
doi:10.15217/issn1684-8853.2015.2.76

Context-Dependent N-Gram Method for Detecting Fuzzy Duplicates in Relational Databases

Tarasov S. V.a, M. Sc., Principal R&D Engineer, st@arbinada.com
Burakov V. V.b, Dr. Sc., Tech., Professor, burakov@aanet.ru
[a]Bel Air Informatique, 1 avenue de l'Atlantique, 91976, Courtaboeuf Cedex, France
[b]Saint-Petersburg State University of Aerospace Instrumentation, 67, B. Morskaia St., 190000, Saint-Petersburg, Russian Federation

Purpose: One of the indicators of poor data quality is data duplication which can lead to a misinterpretation of the same object as several different ones. The relational data model and industrial relational databases can exclude full data duplication, but miss mechanisms which would recognize and prevent fuzzy duplicates. The objective of this work is finding such a way of detecting fuzzy duplicates that could be implemented in the relational data model and in an industrial relational DBMS. **Results:** The fuzzy duplication problem was discussed, common for all information systems. Methods were proposed to enter semantic duplicates into a relational database. It was found out that to solve the problem of imcompleteduplication, we should use fuzzy string comparison methods, taking into account the semantics of the strings. A real-life implementation example was given for PostgreSQL DBMS. **Practical relevance:** The developed method can automatically detect fuzzy duplicates without any human interference, improving the information system data quality. An example of practical implementation for an industrial DBMS can help to immediately use this method in the engineering of information systems. The authors have already used this method for automated information system design.

Keywords — Fuzzy Duplicates, Near-Duplicates, Semantic Duplicates, n-Gram Method, Relational Database, Data Cleansing, Information Quality.

References

1. State Standard ISO 8000-102-2011. Data quality. Part 102. Basic data. Data exchange specifications. Dictionary. Moscow, Standartov Publ., 2012. 16 p. (In Russian).
2. Nekhai I. V. The Use of n-gram Statistics and other Levels of Characters and Words for Semantic Classification Unknown Proper Names. *Komp'iuternaia lingvistika i intellektual'nye tekhnologii*: *Po materialam konferentsii "Dialog"* [Computational Linguistics and Intellectual Technologies. Proc. Intern. Conf. "Dialogue"], 2012, iss. 11(18), vol. 1, pp. 477–489 (In Russian).
3. Mazov N. A. *N-grammnye metody obrabotki tekstovoi informatsii* [N-gram Methods for Texts Processing]. Novosibirsk, Ob"edinennyi institut geologii, geofiziki i mineralogii Sibirskogo otdeleniia RAN Publ., 1995. 180 p. (In Russian).
4. Gudkov V. Iu., Gudkova E. F. N-gram in Linguistics. *Vestnik Cheliabinskogo gosudarstvennogo universiteta*, 2011, no. 24(239), pp. 69–71 (In Russian).
5. Grzegorz Kondrak. N-Gram Similarity and Distance. Department of Computing Science, University of Alberta, Edmonton, AB, T6G 2E8, Canada, 2005. Pp. 115–126.
6. Zelenkov Yu. G., Segalovich I. V. Comparative Analysis of Near-Duplicate Detection Methods of Web Documents. *Trudy 9 Vserossiiskoi nauchnoi konferentsii "Elektronnye biblioteki: perspektivnye metody i tekhnologii, elektronnye kollektsii"* [Proc. 9th Intern. Conf. "Digital Libraries: Advanced Methods and Technologies"]. RCDL Publ., 2007, pp. 166–174 (In Russian).

ПАМЯТКА ДЛЯ АВТОРОВ

Поступающие в редакцию статьи проходят обязательное рецензирование.

При наличии положительной рецензии статья рассматривается редакционной коллегией. Принятая в печать статья направляется автору для согласования редакторских правок. После согласования автор представляет в редакцию окончательный вариант текста статьи.

Процедуры согласования текста статьи могут осуществляться как непосредственно в редакции, так и по e-mail (ius.spb@gmail.com).

При отклонении статьи редакция представляет автору мотивированное заключение и рецензию, при необходимости доработать статью — рецензию. Рукописи не возвращаются.

Редакция журнала напоминает, что ответственность за достоверность и точность рекламных материалов несут рекламодатели.