ISBN 978-5-901795-13-2 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2008

Н. В. Юмагужин

Сопоставление записей и интеграция данных о физических лицах[*)]

Научный руководитель: чл.-корр. РАН С. М. Абрамов

Аннотация. В подавляющем большинстве проектов по системной интеграции встречается задача получения или синхронизации данных об одном человеке из различных источников, поэтому задача сопоставления данных о физических лицах является наиболее распространенным примером в работах по интеграции данных и имеет большое практическое значение. В данной статье разработан алгоритм сопоставления данных о физических лицах и проведен анализ его эффективности. Для описания взаимосвязей между схемами данных применяется методика, описанная в более ранней работе автора, позволяющая упростить этап проектирования и избежать типичных ошибок.

1. Введение

В данной работе рассматривается задача интеграции данных о физических лицах между несколькими информационными системами. Сама по себе задача сопоставления записей о физических лицах не является новой. Например, с 1950-ого года вышло более ста публикаций на тему поиска дубликатов записей о пациентах в различных медицинских базах данных для эпидемиологических исследований [1]. В частности, эти исследования показывают, что при условии высокого качества данных, полностью автоматическое сопоставление записей работает точнее, чем полуавтоматическое сопоставление с участием пользователей [2]. Не так давно вышел ряд публикаций по сопоставлению записей о гражданах для социологических исследований по трудоустройству [3–5]. Большой интерес к проблеме сопоставления данных о физических лицах есть в банках (для отслеживания кредитной истории), в розничной торговле (для выстраивания долгосрочных отношений с клиентами), в налоговых органах и других государственных структурах (для контроля исполнения требований

[*)] Представлено по тематике: *Математические основы программирования, Информатизация управления предприятием, Методы разработки информационных систем.*

законодательства). В данной работе будет рассматриваться задача, стоящая перед многими государственными структурами: учет данных о платежах совершаемых физическими лицами на определенные банковские счета и проверка достоверности сведений представленных о себе плательщиками. Для решения этой задачи мы применим подход, описанный в моей более ранней статье [6]. Суть этого подхода заключается в том, что на этапе анализа разработчики совместно со специалистами в предметной области определяют взаимосвязи между отдельными атрибутами в интегрируемых системах и наборы атрибутов, которые являются потенциальными ключами. На основе этих сведений производится классификация взаимосвязей между схемами данных в интегрируемых системах и делаются выводы: понадобится или нет участие пользователя в переносе данных, возможно ли появление дубликатов при переносе данных, будет ли возможность делать запросы, использующие информацию из нескольких источников данных.

2. Постановка задачи

Существуют две внешние системы, с которыми должна быть обеспечена интеграция. Первая — это банковская система, в которой хранятся платежи. Информацию о платежах необходимо импортировать в нашу систему. Вторая — это реестр физических лиц, в котором производится проверка достоверности сведений, представленных о себе плательщиками. Сведения о физических лицах из импортированных платежей направляются в реестр физических лиц, и затем обрабатывается результат проверки: плательщик найден или не найден, сведения соответствуют или не соответствуют, а также анализируются правильные сведения о жертвователе.

3. Процедура исследования

3.1. Импорт платежей из банковской системы

Импорт платежей из банковской системы проходит по следующему сценарию:

(1) Создать запись о платеже с реквизитами дата платежа, сумма платежа и расчетный счет.
(2) Найти получателя платежа по расчетному счету и связать его с записью о платеже.

(3) Найти или создать плательщика (физическое лицо) и связать его с записью о платеже.

В данной работе нас интересует третий шаг: как гарантировать, что если плательщик есть в справочнике, то мы его найдем и не создадим дубликат? Во-первых, определим потенциальные ключи в списке физических лиц. Под потенциальным ключом мы понимаем набор реквизитов в записях о физических лицах, совпадение которых говорит о совпадении этих лиц. Или, другими словами, в справочнике физических лиц не может быть двух записей с совпадающими потенциальными ключами. Для физических лиц в соответствии с требованиями предметной области потенциальными ключами традиционно считаются наборы реквизитов $K_1 = \{$ФИО, дата рождения$\}$ и $K_2 = \{$ФИО, паспортные данные$\}$. При этом различные написания ФИО («Иванов» и «ИВАНОВ »), различные форматы даты рождения(«21.10.1982» и «10/21/1982»), наличие пробела в серии паспорта не учитываются. Для реализации этого требования предметной области в нашей системе, мы фиксируем формат хранения реквизитов физического лица: каждый из реквизитов «Фамилия», «Имя», «Отчество» хранится в отдельном поле, заглавными буквами без начальных и конечных пробелов; формат даты «дд.мм.гггг», серия и номер паспорта содержат только цифры.

Теперь определим правила импорта отдельных реквизитов из банковской системы. Текстовые реквизиты, задающие регион и адрес места жительства, имеет смысл переносить без изменений (тождественным преобразованием). А определенные выше реквизиты, для которых мы зафиксировали формат, необходимо приводить к указанному формату соответствующим преобразованием: удалять пробелы, менять регистр символов и т.п. Причем если значения реквизитов в банковской и в нашей системе семантически эквивалентны (например «Иванов » и «ИВАНОВ»), то преобразование переводит значение реквизита из банковской системы в соответствующее значение в нашей системе («Иванов » преобразуется в «ИВАНОВ»). В статье [6] было доказано утверждение, согласно которому если для всех реквизитов входящих в потенциальный ключ во второй системе правила переноса данных из первой системы удовлетворяют описанному выше свойству, то при переносе данных запись во второй системе можно определить однозначно. Другими словами, можно гарантировать, что при импорте данных из банковской системы в справочнике физических лиц дубликаты созданы не будут.

4. Интеграция с реестром ФЛ

Интеграция с реестром физических лиц (ФЛ) требуется для проверки достоверности информации, которую представили о себе плательщики при совершении банковских переводов. При выгрузке данных выгружаются сведения из платежа, представленные о себе плательщиком, а так же идентификатор физического лица, с которым связан платеж. Все реквизиты выгружаются без изменений (тождественным преобразованием). При загрузке сведений из реестра ФЛ, создается объект «результат проверки ФЛ». В результате проверки ФЛ хранятся достоверные сведения о физическом лице, начиная с определенной даты (даты начала действия результата проверки). Результат проверки ФЛ связывается с физическим лицом по идентификатору ФЛ из файла. Перенос остальных реквизитов из реестра ФЛ осложняется тем, что семантически одинаковые реквизиты в платеже и в реестре ФЛ могут различаться. Например, плательщик мог указать улицу «пр. Мира», а в реестре ФЛ эта же улица записывается «Мира проспект». Для нашей системы важно, чтобы значения реквизитов в пожертвовании и в результате проверки совпадали в точности, если они совпадают семантически, так как по совпадению реквизитов определяется достоверность представленных сведений и правомерность платежа. Для того чтобы обеспечить точное совпадение реквизитов, в результат проверки ФЛ записываются реквизиты из файла, если значения семантически различны, и реквизиты из пожертвования, если реквизиты семантически совпадают. Это преобразование можно представить в виде функции:

$$f(x) = \begin{cases} x, & P_{semantic}(x, y) = 1 \\ y, & P_{semantic}(x, y) = 0 \end{cases} \tag{1}$$

Где x — значение реквизита в файле, y — значение реквизита в пожертвовании. $P_{semantic}$ — функция семантической эквивалентности. $P_{semantic}$ должна равняться 1, если реквизиты семантически совпадают, и 0 если реквизиты семантически различаются. Например, $P_{semantic}$ («Мира проспект», «пр.Мира»)=1, а $P_{semantic}$ («пр.Мира», «пр.Вернадского»)=0.

Будем рассматривать два приближения функции $P_{semantic}$:

(1) Реализуем хранимый признак $P_{manual}(x, y) \in \{0, 1\}$, вводимый пользователем в специальном интерфейсе. Пользователь указывает значение $P_{manual}(x, y) = 1$ если реквизиты

x и y по его мнению совпадают, и $P_{manual}(x, y) = 0$ если реквизиты x и y по его мнению различны.

(2) Реализуем алгоритм автоматического сопоставления реквизитов. Будем обозначать степень эквивалентности реквизитов, вычисляемую алгоритмом, $P_{auto}(x, y) \in [0, 1]$.

Для строковых атрибутов степень эквивалентности традиционно вычисляется как расстояние Левенштейна [7] или дистанция редактирования. Расстояние Левенштейна — это мера разницы двух последовательностей символов (строк) относительно минимального количества операций вставки, удаления и замены, необходимых для перевода одной строки в другую. Например, чтобы перевести слово «конь» в слово «кот» нужно совершить одно удаление и одну замену, соответственно расстояние Левенштейна составляет 2.

Особенностью данных, с которыми мы работаем, является наличие аббревиатур и сокращений. Для учета этих особенностей будем использовать алгоритм Смита-Ватермана [8]. Этот алгоритм был изначально разработан для поиска эволюционных взаимосвязей в биологических протеинах и цепочках ДНК и широко применяется в приложениях проверяющих орфографии.

Алгоритм Смита-Ватермана находит наименьший «вес» серии изменений, которые переводят одну строку в другую, то есть наименьшую дистанцию редактирования с учетом весов отдельных операций: изменение символа (мутация), добавление символа, удаление символа, начало зазора (аббревиатуры), продолжение зазора (аббревиатуры). Веса операций являются параметрами алгоритма.

Для определенности зададим алфавит A, с которым работает алгоритм, из букв в верхнем и нижнем регистрах, десяти цифр и трех знаков препинания: пробела, запятой и точки. Все остальные символы удаляются перед применением алгоритма.

Алгоритм Смита-Ватермана принимает на вход три параметра: m, s и c. Если задан алфавит A, то m это матрица размера $|A| \times |A|$, в которой указаны веса соответствия для каждой пары символов алфавита $|A|$. В матрице m мы будем задавать вес для точного совпадения символов, для возможного совпадения и для точного несовпадения символов. Изначально в алгоритме Смита-Ватермана эта матрица моделировала вероятность мутаций происходящих в цепочках ДНК в природе. В нашей работе мы будем использовать эту матрицу для того, чтобы учесть типичные орфографические ошибки и опечатки, которые могут возникать при ручном вводе записей в базу данных.

Основное преимущество алгоритма Смита-Ватермана перед аналогичными алгоритмами состоит в возможности вставлять зазоры в строки. Зазор — это последовательность символов в строке, которые не участвуют в сопоставлении. В следующем примере зазоры указаны звездочками:

Строка 1: 152020, Переславль-Залесский Ярославской области,

Строка 2: 152020, Переславль********** Ярославской обл****,

Строка 1: Институт программных систем РАН.

Строка 2: И******* П********** С***** РАН.

Скалярный параметр s — это вес начала зазора, параметр c — это вес продолжения зазора. Значения этих параметров существенно влияют на поведение алгоритма. Например, если значения параметров таковы, что относительно легко продолжать зазор ($c < s$), то алгоритм Смита-Ватермана будет стараться найти один длинный зазор вместо нескольких коротких. Поскольку алгоритм Смита-Ватермана позволяет учитывать зазоры из несоответствующих символов, интуитивно понятно, что он должен хорошо работать с аббревиатурами. Так же он должен хорошо работать когда строки имеют незначительные синтаксические отличия или когда в них пропущены короткие подстроки.

В ходе работы алгоритм Смита-Ватермана вычисляет матрицу весов E. Одна из строк помещается по горизонтальной оси матрицы, а вторая строка помещается по вертикальной оси. Элемент $E(i,j)$ матрицы — это лучший вес сопоставления подстрок $s_1[1..i]$ первой строки и $s_2[1..j]$ второй строки. Когда подстроки (или строки целиком) полностью совпадают, оптимальный путь проходит в матрице E по главной диагонали. При приблизительном соответствии строк оптимальный путь проходит на небольшом расстоянии от главной диагонали. Формально значение $E(i,j)$ вычисляется следующим образом:

$$(2)\quad E(i,j) = max \begin{cases} E(i-1,j-1)+ & m(s_1(i), s_2(j)) \\ E(i-1,j)+c & align(i-1,j-1) = gap \\ E(i-1,j)+s & align(i-1,j-1) = match \\ E(i,j-1)+c & align(i-1,j-1) = gap \\ E(i,j-1)+s & align(i-1,j-1) = match \end{cases}$$

Здесь $align(i,j) = gap$ означает зазор, $align(i,j) = match$ означает совпадение.

Параметры для алгоритма были подобраны экспериментально. Для совпадающих символов независимо от регистра элементы в матрице m заполняются значением «5». Для близких по звучанию или написанию символов элементы в матрице m заполняются значением «3». Все остальные элементы матрицы m заполняются значением «-3». Веса начала и продолжения зазора равняются 5 и 1 соответственно.

Второй особенностью данных, с которыми мы работаем, является возможность перестановки слов в строке. Например, если сравнивать следующие три строки:

1: Институт Программных Систем, Переславль-Залесский

2: Переславль-Залесский, Институт Программных Систем

3: Институт программирования и систем управления

то при использовании алгоритма Смита-Ватермана для всех трех будет выдано близкое расстояние. Хотя первые две строки семантически похожи гораздо больше. Для того чтобы учесть возможность перестановки подстрок в строке будем повторно анализировать зазоры оставленные алгоритмом Смита-Ватермана. Для приведенного примера при первом проходе алгоритма для первых двух строк сопоставляются следующие подстроки:

1: Институт Программных Систем**********

2: **********Институт Программных Систем

Соответственно при втором проходе будут сопоставлены подстроки:

1: **Переславль-Залесский

2: Переславль-Залесский**

То есть мы получаем полное совпадение за исключением нескольких символов. Чтобы учесть, что подстроки в строке переставлены, результат второго прохода алгоритма можно домножить на некоторый коэффициент меньший единицы. Эксперименты показывают, что повторение этой операции более 2-3 раз не дает дополнительного результата. Соответственно, можно считать, что теоретическая сложность алгоритма остается квадратичной. Время работы при этом увеличивается на 50%.Будем использовать описанный алгоритм для вычисления P_{auto}.

На данный момент у нас есть база данных с указанными вручную признаками соответствия $P_{manual}(x, y)$ для 1000 плательщиков.

Этого не достаточно для полноценной оценки работы алгоритма вычисления P_{auto}, но предварительно можно сказать, что округленное значение P_{auto} совпадает с P_{manual} более чем в 90% случаев.

5. Результаты

На основе проведенного анализа мы получили следующие результаты:

(1) Доказали, что импорт данных о платежах из банковской системы возможен в автоматическом режиме и при этом не могут возникать дубликаты плательщиков в справочнике физических лиц.
(2) Установили, что если реализовать проверку достоверности сведений о плательщиках в автоматическом режиме, то возможно появление ошибок.
(3) Предложили реализацию проверки достоверности сведений о плательщиках в автоматическом и полуавтоматическом режимах.

Кроме того, были начаты экспериментальные исследования по оценке эффективности предложенного алгоритма автоматического сопоставления записей.

6. Выводы

Исследования в области сопоставления схем данных о физических лицах и сопоставления записей (поиска и устранения дубликатов) имеют большое практическое значение. Как видно на примере данной работы, приведенная в статье [6] классификация взаимосвязей в схемах данных может применяться в широком спектре проектов по интеграции информационных систем, на этапе анализа предметной области и формирования проектных решений. Преимущество подхода, описанного в статье [6], перед классическим описанием алгоритмов переноса данных заключается не только в простоте понимания неподготовленным человеком, но и в снижении риска скрытых ошибок. При описании бизнес-процесса, сценария или алгоритма переноса данных очень легко пропустить некоторые альтернативные пути или непредвиденные события, которые могут повлиять на процесс. При описании взаимосвязей между схемами данных такие ошибки практически исключены. Кроме того, как видно на примере

данной работы, уже в самом начале этапа анализа можно делать выводы: понадобится или нет участие пользователя в переносе данных, возможно ли появление дубликатов при переносе или синхронизации данных.

Список литературы

[1] Howard B. Newcombe Handbook of record linkage: methods for health and statistical studies, administration, and business: Oxford University Press, 1988.

[2] Newcombe H. B. and Smith M. E. Methods for Computer Linkage of Hospital Admission-Separation Records into Cumulative Health Histories: Methods of Information in Medicine, 1975. — 14 (3), 118-125 с.

[3] Abowd J. M. and Vilhuber L. The Sensitivity of Economic Statistics to Coding Errors in Personal Identifiers (with discussion): Journal of Business and Economic Statistics, 2005. — 23 (2), 133-165 с.

[4] Abowd J. M. and Woodcock S. D. Disclosure Limitation in Longitudinal Linked Data. — North Holland: Amsterdam: Confidentiality, Disclosure, and Data Access, 2002.

[5] Abowd J. M. and Woodcock S. D. Multiply-Imputing Confidential Characteristics and File Links in Longitudinal Linked Data. — Springer: New York: Privacy in Statistical Databases 2004, 2004. — 290-287 с.

[6] Юмагужин Н.В. Классификация взаимосвязей в схемах данных. — Тверь: Программные продукты и системы, номер 3, 2007.

[7] В.И. Левенштейн Двоичные коды, обеспечивающие синхронизацию и исправление ошибок. — Москва: Тезисы кратких научных сообщений Международного конгресса математиков, 1996. — Секция 13, 24 с.

[8] Smith T. F. and Waterman M. S. Identification of common molecular subsequences: Journal of Molecular Biology, 1981. — 147: 195-197 с.

N. V. Yumaguzhin. *Record linkage and person data integration* // Proceedings of Programm Systems institute scientific-practical conference "Program systems: Theory and applications", devoted to the 15th anniversary of Pereslavl University named A. K. Ailamazyan. — Pereslavl-Zalesskij, 2008. — p. 265 — 273. — ISBN 978-5-901795-13-2 *(in Russian).*

Abstract. In many system integration projects appears a problem of getting or synchronizing data about one person from different information sources, therefore person record linkage is the most popular example in data integration papers and it has a high practical value. This article describes a person record linkage algorithm and contains an analysis of its effectiveness. For description of schema correlation author uses method described in his earlier article that allows to simplify the design phase and to avoid typical mistakes.

Перевод проверен: Н. В. Юмагужин