УДК 681.142.2

О.Г. Шевелев

ПРЕДСТАВЛЕНИЕ НАБОРА ТЕКСТОВ В РЕЛЯЦИОННОЙ БАЗЕ ДАННЫХ ДЛЯ ЦЕЛЕЙ ЛИНГВИСТИЧЕСКОГО АНАЛИЗА

Рассматриваются возможности использования СУБД в качестве инструмента исследования текстов. Описывается структура базы текстов. Показано, что эффективность получения различных количественных характеристик текста с помощью СУБД в сравнении с алгоритмами, основанными на непосредственном анализе текстовых файлов, существенно выше.

С появлением в электронном виде большого числа текстовых данных открылись широкие возможности для компьютерного анализа текстов. Одним из наиболее популярных направлений в этой области является математическая лингвистика (computational linguistics) [1]. Ее использование предполагает исследование свойств текста путем применения математических методов к его количественным характеристикам.

Подсчет количественных характеристик текста лучше всего производить автоматически. Вместе с тем, хранение данных в текстовых файлах не предполагает никаких удобств для подсчета этих характеристик. Исследователям приходится писать специальные программы обработки текстовых файлов, которые могут существенно различаться для разных характеристик. Всего же количественных характеристик текста может быть как угодно много. Время работы подобных программ в случае больших массивов данных (что в исследованиях по математической лингвистике не редкость [2]), может быть достаточно велико.

В данной статье предлагается способ размещения текстов в реляционной базе данных, определенная структура которой позволяет производить быстрый и удобный подсчет разнообразных количественных характеристик текста.

ПРЕИМУЩЕСТВА ПРЕДСТАВЛЕНИЯ ТЕКСТОВ В БАЗЕ ДАННЫХ

В отличие от хранения текстов в виде текстовых файлов, внесение текстов в базу данных предполагает их централизованное хранение и осмысленную разбивку на элементарные составляющие. Так, разрозненная информация об авторах (обычно хранящаяся в самих файлах или структуре каталогов) приобретает вид таблицы. Тоже происходит с перечнем названий текстов и, главное, с самими текстами, о разбиении которых на отдельные составляющие будет сказано ниже. Между всеми этими элементами появляются связи и возможность фильтрации по ним.

Размещая тексты в БД, помимо удобства хранения мы получаем удобство обращения к ним. Появляется возможность извлечения агрегированных характеристик текста при помощи одного или нескольких коротких (по сравнению с программой обработки текстовых файлов) запросов. Текст перестает быть простым набором байт и букв, а становится последовательностью слов и предложений.

Основное преимущество перевода текстов в табличный вид заключается в том, что, заполнив базу один раз и потратив на это значительное количество времени, в дальнейшем мы можем достаточно быстро манипулировать данными.

Повышению скорости обработки во многом способствует наличие табличных индексов. Предполагается, что исследователь индексирует все поля и связки полей, по которым будут происходить запросы. Использование индексов увеличивает объем базы данных (из-за необходимости хранения служебной информации), но заметно ускоряет ее обработку.

За счет сетевой ориентации используемой СУБД можно осуществить удаленный доступ к базе, что дает возможность проводить удаленные исследования без закачки текстовой базы на компьютер исследователя и расположить базу на каком-либо сервере, по производительности существенно превосходящем рабочую станцию.

СУЩЕСТВУЮЩИЕ НАРАБОТКИ В ДАННОЙ ОБЛАСТИ

Идея создания баз данных текстов не нова. Базы данных предоставляют удобную организацию и доступ к данным, поэтому при обсуждении проблемы хранения и обработки текстовой информации эти вопросы, так или иначе, затрагиваются. Хотя зачастую под базами данных понимают не хранилище под управлением СУБД, а всего лишь некоторый массив текстовой информации, неструктурированной и недоступной для каких-либо запросов, кроме простейшего поиска по внешним признакам текста (автору, названию, времени написания и т.д.). Примерами таких баз могут послужить известные электронные библиотеки Мошкова [3], «Русский текст» [4] и др.

Более сложной организацией баз текстов, но также без поддержки СУБД, является хранение текстов с внутренней разметкой [5]. Разметка, как правило, осуществляется на языке XML. Существуют специальные стандарты описания текстовых данных [6]. Такое хранение обеспечивает более гибкую обработку текстов, но все же не предназначенную для извлечения количественных характеристик.

Размещение текстов в реляционных базах данных может быть осуществлено различными путями. Зачастую тексты помещаются в поля баз данных целиком (тип Memo, Text). По этому принципу размещена текстовая информация на большинстве сайтов интернета, которые используют технологию СУБД, и в некоторых библиотеках с расширенной функциональностью [7]. При такой организации базы SQL-запросы не дают практически никаких преимуществ в плане извлечения количественных характеристик. Полнотекстовая индексация обеспечивает лишь быстроту поиска и удобство сортировки по релевантности, но в процессе обработки тексты все равно приходится каждый раз просматривать целиком, так же, как если бы мы работали с текстовыми файлами.

Наибольшие возможности для сбора количественных характеристик, на наш взгляд, обеспечивает представление текстов в реляционной базе данных в виде

последовательности слов. Была найдена единственная подобная разработка, которая упоминается в ряде статей группы исследователей Петрозаводского государственного университета [8–11]. Судя по описанию, основной причиной разбиения текстов на слова в этой разработке была необходимость установления связей между конкретными словами текста и их морфологическим описанием. Объем текстовой базы был незначительным – несколько десятков небольших произведений. Ниже предлагается структура базы данных, ориентированной на получение количественных характеристик большого числа текстовых данных. Одним из важнейших критериев оценки базы является скорость получения различных количественных характеристик.

ИНСТРУМЕНТАРИЙ

В качестве СУБД для текстовой базы нами была выбрана СУБД MySQL 4.0. Эта СУБД обладает рядом полезных преимуществ для исследователя.

1. Бесплатность. СУБД MySQL [12] распространяется по бесплатной лицензии GPL, позволяя таким образом проводить исследования на легальных условиях без каких-либо материальных вложений.

2. Доступность и небольшой размер. Так как пакет СУБД бесплатный, его можно свободно скачать с сайта производителя. Размер последней рабочей версии MySQL 23 Мб, что существенно меньше каких-либо других СУБД.

3. Мультиплатформенность. СУБД MySQL реализована на всех крупных ОС, что позволяет проводить исследования не только на рабочих станциях Windows, но и на более мощных серверных платформах под управлением Unix.

4. Функциональность. По функциональности пакет MySQL может сравниться с крупными СУБД – есть развитой язык запросов, механизм транзакций, репликация.

5. Удобство доступа, манипуляции и представления данных. Для СУБД MySQL написано множество клиентских программ, с помощью которых можно очень быстро и наглядно произвести обработку данных без написания программы. Кроме того, обращаться к СУБД можно из командной строки через утилиты MySQL с последующим перенаправлением результата в файл (что позволяет осуществить пакетную обработку), а также из программ, написанных на интерпретаторе PHP (что удобно для наглядного представление данных на интернетовской странице), или из программы на C++ через MySQL API в том случае, если нужна высокая скорость промежуточных вычислений.

СТРУКТУРА БАЗЫ ТЕКСТОВ

Предлагаемая нами база текстов состоит из четырех таблиц типа MyISAM, поля в которых представлены четырьмя стандартными типами данных СУБД MySQL (табл. 1) [12].

Таблица 1

Название	Диапазон значений	Размер
bit	0/1	1 бит
unsigned mediumint	0 – 16777215	3 байта
unsigned smallint	0 – 65535	2 байта
varchar	Строка до 255	1–255 байт

От выбора нужного типа данных зависит объем базы и скорость обработки запросов. Выбор типа MyISAM для всех таблиц не случаен. Другой возможный тип таблиц InnoDB более медлителен. Такие таблицы, прежде всего, обеспечивают надежность хранения, пополнения (механизм транзакций) и целостность данных (внешние ключи), что в нашем случае менее важно, чем скорость манипуляции с данными.

Рассмотрим таблицы базы подробнее.

1. Таблица author (автор).

Хранит список авторов и содержит сокращенное имя автора (varchar), полное имя автора (varchar), идентификатор записи (unsigned mediumint) и его индекс (рис 1).

Id	sAuthorName	sAuthorLName
1	chehovap	Чехов А.П.
2	dostoevskyfm	Достоевский Ф.М.
3	gogolnv	Гоголь Н.В.
4	goncharovia	Гончаров И.А.
5	kuprinai	Куприн А.И.
6	leskovns	Лесков Н.С.
7	pushkinas	Пушкин А.С.
8	saltykov-shedrinme	Салтыков-Щедрин
9	tolstoyln	Толстой Л.Н.
10	turgenevis	Тургенев И.С.

Рис. 1. Пример данных в таблице author

Разделение на полное и сокращение имя автора было произведено для удобства отображения данных в различных случаях.

2. Таблица text (текст).

Предназначена для хранения перечня названий имеющихся текстов. Включает в себя идентификатор записи (unsigned mediumint), его индекс, сокращенное название текста (varchar), полное название текста (varchar) и идентификатор автора текста (unsigned mediumint) (рис. 2).

Id	sTextTitle	iAuthorId	sTextLTitle
38	anna	9	Анна Каренина
39	detstvo	9	Детство
40	hadjimurat	9	Хаджи-Мурат
41	smert	9	Смерть Ивана Ильича
42	voskr	9	Воскресение
43	asya	10	Ася
44	dvory_gnezdo	10	Дворянское гнездо
45	nakanune	10	Накануне
46	neschasnaya	10	Несчастная
47	otcydeti	10	Отцы и дети

Рис. 2. Пример данных в таблице text

3. Таблица word (слово).

Хранит неповторяющиеся текстовые элементы (слова) – словоформы, знаки препинания и переносов и т.д. Содержит поле идентификатора записи (mediumint), его индекс, значение слова (varchar) и индекс значения слова (для ускорения поиска по слову) (рис. 3).

Так как база рассчитана на большое число текстов, таблица word может содержать сотни тысяч, а возможно даже миллионы слов. Цифра довольно большая, требующая внимания к размерам ее полей и заполнению.

Тип идентификатора записи – трехбайтовый unsigned mediumint – был выбран как наименьший из возможных (двухбайтовый smallint предполагает недостаточное – 65

тыс. – число значений). Поле значения слова имеет переменный (в зависимости от длины слова) размер.

Id	sWord
1427	бумаги
1428	поднялся
1429	лениво
1430	головокружением
1431	зевая
1432	шлепая
1433	туфлями
1434	соседнюю
1435	комнату
1436	открытого
1437	стоял
1438	молодых
1439	сослуживцев
1440	раскладывал

Рис. 3. Пример данных в таблице word

Чтобы избежать хранения лишних данных, была написана простая программа предобработки preproc, которая отмечала появление нестандартных для текстовых файлов символов, а также смешение различных видов символов (русские, английские буквы, цифры) в пределах одного слова. Как оказалось, компьютерные тексты, в том числе используемые нами тексты с электронной библиотеки Мошкова [3] и «Русский текст» [4], содержат немало опечаток. Часть из них можно обнаружить в процессе автоматического анализа. Так, с помощью preproc на примере наших данных было выявлено более 10 тыс. подобных ошибок на 56 Мб текстовых данных. Часть из них удалось исправить автоматически.

4. Таблица wordslist (список слов).

Содержит связанные списки слов, то есть фактически сами тексты. Каждая запись – это определенное по порядку слово в конкретном тексте. Таблица состоит из следующих полей: идентификатора текста (unsigned mediumint), ссылающегося на таблицу text; порядкового номера предложения (unsigned smallint), порядкового номера слова в предложении (unsigned smallint); идентификатора слова (unsigned mediumint), ссылающегося на таблицу word; признака (bit), определяющего заглавие первой буквы; индекса порядкового номера предложения, порядкового номера слова в предложении и тексте; индекса по идентификатору слова (рис. 4).

iTextId	iSentNum	iWordNum	iWordId	bFirstCapital
1	212	60	1333	0
1	212	61	1317	0
1	212	62	4	0
1	213	0	1334	1
1	213	1	14	0
1	213	2	1335	0
1	213	3	14	0

Рис. 4. Пример данных в таблице wordslist

У таблицы нет специально отведенного индекса записи, в качестве него выступает индекс «идентификатор текста – порядковый номер предложения – порядковый номер слова», который однозначно определяет запись.

Размеры полей идентификаторов текста и слова определяются соответственно размерами идентификаторов табл. text и word. Для порядкового номера предложения выбран двухбайтовый тип unsigned smallint, как наиболее подходящий – он может охватить текст с двумя тысячами страниц, что вполне достаточно, а однобайтовый unsigned tinyint слишком мал (255 значений). Разбивка текста на предложения необходима для быстрого установления границ нужного предложения, в противном случае придется каждый раз производить дополнительный анализ текста, что существенно замедлит сбор статистики. Порядковый номер слова в предложении имеет тип smallint с подходящим диапазоном значений (65 тыс.) и в то же время наименьшим из возможных (tinyint с диапазоном 0–255 слишком мал). Так как в табл. word все слова записаны прописными буквами, поле признака заглавия первой буквы слова в wordslist необходимо для того, чтобы сохранить информацию об именах собственных. Предполагается, что слово может начинаться либо с заглавной, либо с прописной буквы – все остальные нестандартные виды написания для унификации сводятся к этим двум.

Слово как наименьший неделимый элемент текста выбрано по следующим причинам. Во-первых, большинство количественных характеристик текста связано с уровнем слова, либо с последовательностью слов (предложением, абзацем). Во-вторых, выбор слова дает хороший задел для исследований: слово можно охарактеризовать по части речи, роли в предложении, смыслу. В-третьих, более мелкое деление на буквосочетание привело бы к катастрофическому росту числа полей БД и гораздо меньшей наглядности для исследователя.

Заполнению табл. wordslist предшествовала небольшая предобработка текстов. В ходе предобработки убирались «лишние» переносы строк, нужные только для форматирования текста в текстовых файлах (перенос сохранялся только в том случае, если следующая строка начиналась с пробела, табуляции, либо другого переноса).

ПРИМЕРЫ ОБРАЩЕНИЕ К БАЗЕ

Для демонстрации удобства сбора количественных характеристик по текстам, находящимся в базе данных, приведем примеры запросов:

1. Получение среднего числа слов в предложении для текстов, у которых этот параметр больше 23 и меньше 5.

SQL-запрос:

```
select sTextLTitle, sAuthorlName,
avg(iWordNum) as wrdavg
from wordslist
join text on (text.id = iTextId)
join author on (author.id = iAuthorId)
group by iTextId
having ((wrdavg > 25) or (wrdavg < 5))
order by wrdavg desc;
```

Возможный результат представлен на рис. 5.

2. Получить слова, подозрительные на имя собственное, с числом встречаемости в тексте более 100 раз (наиболее часто встречаемое из них, скорее всего, окажется главным героем произведения) на примере «Преступления и наказания» Ф.М.Достоевский.

SQL-запрос:

```
select w1.sWord, count (*) as nc
from wordslist as wl1, wordslist as wl2
```

```
join word as w1 on (w1.id = wl1.iWordId)
join word as w2 on (w2.id = wl2.iWordId)
join text on (text.id = wl1.iTextId)
where (wl1.iWordNum <> 0) and (wl1.bFirstCapital) and
(wl1.iTextId = wl2.iTextId) and (sTextLTitle = "Преступление
и наказание") and (wl1.iSentNum = wl2.iSentNum)
and (wl2.iWordNum = wl1.iWordNum - 1) and (not
(w2.sWord in ("\"", "-")))
group by w1.sWord
having nc >100
order by nc desc;
```

Результат представлен на рис. 6.

sTextLTitle	sAuthorlName	wrdavg
Другие берега	Набоков В.	28,5959
Жизнь Арсеньева	Бунин И.	28,4764
Компромисс	Довлатов С.	4,612
Зона	Довлатов С.	4,5668
Соло на ундервуде	Довлатов С.	4,509

Рис. 5. Возможный результат выполнения запроса из примера 1

sWord	nc
раскольников	347
петрович	208
разумихин	196
ивановна	175
соня	173
раскольникова	125
александровна	109
дуня	101

Рис. 6. Результат выполнения запроса из примера 2

3. Подсчитать число слов, встречающихся 1, 2 и т.д. до 10 раз в произведении «Вий» Н.В. Гоголя.

SQL-запрос:

```
create temporary table
tmp(sWord varchar(255), cw int);
insert into tmp
select sWord, count(iWordId) as cw
from wordslist
join word on (word.id = iWordId)
join text on (text.id = iTextId)
where sTextLTitle = "Вий"
group by sWord
having cw <= 10;
select cw, count(*)
from tmp
group by cw
order by cw asc;
```

Результат представлен на рис. 7.

ОЦЕНКА ЭФФЕКТИВНОСТИ

Сравним время обработки базы текстов и текстовых файлов на примере получения количественных характеристик авторского стиля текста. Запросы к базе производились на языке SQL-запросов MySQL [12], текстовые файлы обрабатывались программой на GNU C. Все приводимые оценки получены на компьютере Intel Pentium Xeon 2.3 MHz x 2/4 Gb/30 x 5 Gb SCSCI RAID под управлением операционной системы Windows 2003 Server. Объем исследуемой базы данных 375 Мб, всего авторов – 36, текстов – 155, общее число записей в таблице wordslis*t* – 11 млн., word – 400 тыс. Объем текстовых файлов 56 Мб.

cw	count(*)
1	3141
2	577
3	198
4	101
5	78
6	51
7	39
8	18
9	27
10	14

Рис. 7. Результат выполнения запроса из примера 3

Разделим характеристики авторства текста на следующие классы:

1. Простые характеристики. Простыми назовем такие характеристики, которые можно получить за один просмотр текстовых данных. Среди них характеристики, основанные на подсчете определенных слов – служебных (авторский инвариант Фоменко [13]), слов определенной длины, с определенным набором букв или определенными грамматическими признаками. А также усредняющие характеристики – средняя длина предложения (или длины предложений), средняя длина слова и прочие.

Однопроходной алгоритм, по порядку разбирающий тексты, вычленяя из них слова с помощью лексического анализатора, работает около 6 часов. Такое же время необходимо для заполнения базы данных. После заполнения базы с помощью SQL-запроса на получение любой простой характеристики требуется не более 10 мин. В случае же работы с текстовыми файлами необходимо каждый раз ждать полного просмотра, т.е. 6 ч. И хотя при полном просмотре можно получить сразу несколько характеристик, а SQL-запросом сделать такое удается не всегда, выигрыш во времени при работе с базой данных исчисляется часами.

2. Сложные характеристики. Сложные характеристики можно условно разделить на два вида:

а) двухпроходные характеристики. Для их вычисления необходим предварительный просмотр всех текстов. За два прохода, например, можно получить частоты наиболее часто встречающихся слов. В ходе первого просмотра определяются искомые слова, в ходе второго – производится их подсчет. Время анализа текстовых файлов – 12 ч., базы – 5–10 мин.

б) характеристики, требующие дополнительного анализа данных. Как правило, это характеристики, связанные с формированием словаря по текстам. Например, установление богатства словаря автора, сравнение словарного запаса двух различных текстов. Сюда же можно отнести подсчет количества слов, встречающихся в тексте один раз (hapax legomena), два раза (dislegomena) и т.д. Работая с текстовыми файлами, приходится заводить дополнительный массив слов и на каждом шаге просматривать его. Даже если на ходу попытаться

упорядочить данные, придумать специальную хеш-функцию, подобная обработка займет сутки и более. Извлечение подобных признаков из индексированной базы данных, уже включающей список всех слов (таблица word), с помощью специализированных SQL-операторов требует 5–20 мин.

ВЫВОДЫ И ПЕРСПЕКТИВЫ

Как видно из экспериментов, время получения количественных характеристик из текстов, специальным образом размещенных в БД, существенно меньше, чем из текстовых файлов. Код для получения количественных характеристик из базы невелик и более нагляден, чем программа обработки текстовых файлов. И хотя мы допускаем существование таких количественных характеристик, сбор которых без применения СУБД будет проще и эффективней, в целом предложенный подход оправдывает свое использование для целей лингвистического анализа текстов.

В дальнейшем, мы планируем расширить базу данных таблицами, включающими в себя различные словари русского языка, например, грамматический словарь А.А. Зализняка [14]. С помощью дополнительных словарей можно получать дополнительные лингвистические характеристики текста.

Мы надеемся, что вовлечение баз данных в процесс анализа текстов поможет преодолеть сложности обработки больших объемов текстовой информации и облегчит работу исследователей в области математической лингвистики.

ЛИТЕРАТУРА

1. *Анисимов А.* Компьютерная лингвистика для всех: Мифы. Алгоритмы. Язык. Киев: Наук. думка, 1991. 208 с. (http://lib.bigmir.net/read.php?e=967)
2. *Хмелёв Д.В.* Распознавание автора текста с использованием цепей А.А. Маркова. Вестник МГУ. Сер. 9. Филология. № 2. 2000. С. 115–126 (http://zhurnal.lib.ru/h/hmelew_d_w/vestnik1999.shtml).
3. *Электронная* библиотека Максима Мошкова (http://lib.ru).
4. *Электронная* библиотека Русский текст (http://www.russiantext.com).
5. *Сичинава Д.В.* К задаче создания корпусов русского языка в Интернете, 2002 (http://rscorpora.narod.ru/article.html).
6. *The XML Version* of the TEI guidelines. The text encoding initiative consortium, 2001 (http://www.tei-c.org.uk/P4X/).
7. *The ARTFL Project*: A textual database (http://humanities.uchicago.edu/orgs/ARTFL/artfl.flyer.html).
8. *Сидоров Ю.В.* Математическая и информационная поддержка методов обработки литературных текстов на основе формально-грамматических параметров: Автореф. дис ... канд. тех. наук. Петрозаводск, 2002. 19 с.
9. *Сидоров Ю.В., Леонтьев А.А., Рогов А.А., Захаров В.Н.* Компьютерная автоматизированная система для лингвистического разбора литературных текстов // IV Санкт-Петербургская ассамблея молодых ученых и специалистов: Тез. докл. СПб. 1999. С. 66.
10. *Захаров В.Н., Леонтьев А.А., Рогов А.А., Сидоров Ю.В.* Программная система поддержки атрибуции текстов статей Ф.М. Достоевского. Труды Петрозаводского государственного университета. Сер. Прикл. матем. и информ. Вып. 9. Петрозаводск: Изд-во ПетрГУ, 2000. С. 113–122.
11. *Rogov A.A., Sidorov Yu. Vl.* Statistical and information-calculating support of the authorship attribution of the literary works. Computer data analysis and modeling: robustness and computer intensive methods // Proc. of the Sixth International conference (September 10–14, 2001, Minsk). Vol. 2: K-S / Edited by prof. dr. S. Aivazian, prof. dr. Yu. Kharin and prof. dr. H. Rieder. Minsk: BSU, 2001. P. 187–192.
12. *MySQL* Reference manual for version 4.0.15a (http://www.inet-sec.org/docs/REF-cards/CHM/mysql-4.0.15a.chm).
13. *Фоменко В.П., Фоменко Т.Г.* Авторский инвариант русских литературных текстов. Предисловие А.Т. Фоменко // Фоменко А.Т. Новая хронология Греции: Античность в средневековье. Т. 2. М.: Изд-во МГУ, 1996. С. 768–820 (http://lib.ru/FOMENKOAT/greece.txt).
14. *Зализняк А.А.* Грамматический словарь русского языка. М.: Русский язык, 1980.

Статья представлена кафедрой прикладных основ информатики факультета информатики Томского государственного университета, поступила в научную редакцию «Информатика» 30 апреля 2004 г.