СОЗДАНИЕ И ПЕРЕРАБОТКА ЛЕКСИЧЕСКИХ БАЗ ДАННЫХ В ИНТЕГРИРОВАННОЙ ИНФОРМАЦИОННОЙ СРЕДЕ STARLING[1]

CREATION AND PROCESSING OF LEXICAL DATABASES IN THE ENVIRONMENT OF THE STARLING INTEGRATED INFORMATIONAL SYSTEM

Крылов С.А. *(krylov-58@mail.ru),* ***Старостин С.А****,*
Институт востоковедения РАН

Задачи компьютерной лексикографии, решаемые системой STARLING: (1) создание лексических баз данных (ЛБД); (2) разметка ЛБД (автоматическая и полуавтоматическая); (3) Переструктурирование создаваемых ЛБД, приближающее их формальную структуру к их содержательному наполнению.

0. STARLing и компьютерная лексикография

Доклад является логическим продолжением работы, опубликованной в сборнике трудов конференции «Диалог 2006»[2]. Он посвящён основным принципам создания и переработки лексических баз данных (ЛБД) в ИИС STARLing.

Лексические базы данных (ЛБД) могут иметь различное происхождение. С точки зрения происхождения можно выделить следующие виды ЛБД: 1) на основе некоторых предварительно составленных «словарей» (в широком смысле слова, т. е. включая в понятие словарей такие источники, как разного рода таблицы, указатели и т. п.).

1. Происхождение ЛБД: основные типы

Лексические базы данных (ЛБД) могут иметь различное происхождение. С точки зрения происхождения можно выделить следующие виды ЛБД: 1) на основе некоторых предварительно составленных «словарей» (в широком смысле слова, т. е. включая в понятие словарей такие источники, как разного рода таблицы, указатели и т. п.), в том числе: (1а) «бумажных» или «небумажных» словарей, непереводимых автоматически в стандартный текстовый формат; и (1б) словарей на читаемых электронных носителях, переводимых в стандартный текстовый формат; 2) на основе некоторых связных текстов или корпусов текстов, в том числе: (2а) «бумажных» или «небумажных», непереводимых автоматически в стандартный текстовый формат; и (2б) текстов, представленных в электронной форме в стандартном текстовом формате.

Тип «2а» сводится к типу «2б» путём сканирования с распознаванием, поэтому самостоятельного интереса не представляет.

Процедуры преобразования текстовых данных (типа «2б») в стандартную текстовую базу данных (ТБД) были рассмотрены в работе [1]. В настоящей работе рассмотрим подробнее процедуры обработки данных типа «1а» и «1б».

Для типа «1а» возможно две принципиально разных стратегии работы («Ст-1» и «Ст-2»). Стратегия «Ст-1» («Предварительное создание электронной текстовой версии словаря») аналогична переходу «2а→2б»; в этом случае мы осуществляем преобразование «1а→1б», и задача дальнейшей обработки сводится к обработке данных типа «1б». При стратегии «Ст-2» («Ручной ввод информации в ЛБД») создаётся новая ЛБД, далее заполняемая информацией «вручную».

2. Расстановка делимитаторов в тексте электронного словаря

В рамках стратегии Ст-1 основная задача сводится к тому, чтобы в электронной версии словаря расставить «делимитаторы» (то есть сигналы границ), необходимые для конвертирования текста словаря в ЛБД и для

[1] Работа выполнена по гранту РФФИ номер 05-06-80236.

[2] С. А. Крылов, С. А. Старостин Интегрированная информационная среда STARLING и ее использование в сфере корпусной лингвистики // Компьютерная лингвистика и интеллектуальные технологии: Труды международной конференции «Диалог 2006» (Бекасово, 31 мая – 4 июня 2006 г.) – М.: Изд-во РГГУ, 2006.

дальнейшей логической переработки этой ЛБД. Такие делимитаторы бывают двух типов: делимитаторы записей («records»), или R-делимитаторы, и делимитаторы полей («fields»), или F-делимитаторы.

В большинстве изданных словарей основным делимитатором вокабул (словарных статей) служит абзацный отступ. Поэтому та ASCII-последовательность, которая является стандартным способом выражения абзацного отступа в текстах (а именно, последовательность «символ 13 + символ 10»), как раз и выбрана в качестве стандартного R-делимитатора при конвертировании словаря в ЛБД. В структуре исходной ЛБД этот делимитатор (назовём его «главным» делимитатором, или «D_0») выполняет роль показателя «первичного дробления» словаря на вокабулы. «Первичным» дроблением словаря осмысленно считать такое дробление, результатом которого являются вокабулы, не входящие в состав никаких более крупных объединений («супервокабул») данного словаря.

Если в исходном тексте электронного словаря используются абзацные отступы нескольких разных уровней глубины, то для сохранения этой информации в ЛБД каждый тип исходных сигналов таких отступов должен тем или иным образом получить уникальное для него обозначение в форме некоторой уникальной последовательности ASCII-кодов. Лишь один из этих типов (D_0) может быть объявлен «главным» R-делимитатором; все остальные будут объявлены побочными делимитаторами.

При конвертировании словарного текста в формат ЛБД каждый абзац превращается в «запись» (в формальном смысле этого слова). Это, однако, не означает, что побочные делимитаторы оказываются несущественными. Они учитываются, но на следующих этапах преобразования.

Дальнейший учёт побочных делимитаторов осуществляется с помощью одного из двух подходов: во-первых (D_1-R), некоторый тип побочных делимитаторов может быть на некотором этапе объявлен R-делимитатором; во-вторых (D_1-F), некоторый тип побочных делимитаторов может быть на некотором этапе объявлен F-делимитатором.

При осуществлении процедуры D_1-R существенно обеспечить сохранение информации о позициях делимитатора D_0. Для этого, вообще говоря, существует два приёма (логически достаточно использования лишь одного из них, причём всё равно какого; однако практически иногда удобнее прибегать к избыточному сочетанию обоих приёмов): $Sa_1(D_0)$ и $Sa_2(D_0)$.

3. Предварительная нумерация словарных статей, подлежащих декомпозиции

Приём $Sa_1(D_0)$ состоит в том, что в структуру исходной ЛБД с помощью стандартной процедуры “Modify structure” добавляется дополнительное поле F_i (назовём его «адрес D_0»), после чего с помощью стандартной процедуры “Replace” это поле заполняется (в глобальном диапазоне, то есть во всех записях исходной ЛБД) информацией о номере той записи, в которой локализован фрагмент словарной информации между данным вхождением делимитатора D_0 и его последующим вхождением. Поле F_i может принадлежать либо к числовому типу (в таком случае оно заполняется числовой функцией Recno(), содержательно равной «номеру записи»), либо к словесному типу; однако есть некоторое практическое преимущество в выборе словесного типа поля; в этом последнем случае оно заполняется составной «цепочечной» (словесной) функцией Str(Recno()), содержательно тоже равной «номеру записи», но формально соответствующей преобразованию этого номера в «цепочечный вид», при котором числу соответствует сочетание небольшого числа начальных пробелов с последовательной конкатенацией всех цифр, входящих в десятичную запись данного числа.

В результате этой операции информация о номере записи, содержавшей данный фрагмент информации в исходной ЛБД, продолжает храниться в производной от неё ЛБД – в частности, и после того, как в качестве главного делимитатора (R-делимитатора) был объявлен некоторый новый делимитатор, не совпадающий с делимитатором D_0.

4. Дублирование инварианта словарных гнёзд, подлежащих декомпозиции

Приём $Sa_2(D_0)$ состоит в том, что в структуру исходной ЛБД с помощью стандартной процедуры “Modify structure” добавляется дополнительное поле F_j (назовём его «инвариант словарного гнезда»), после чего с помощью стандартной процедуры “Replace” это поле заполняется (в глобальном диапазоне, то есть во всех записях исходной ЛБД) информацией о номере той записи, в которой локализован фрагмент словарной информации между данным вхождением делимитатора D_0 и его последующим вхождением. Поле F_i может принадлежать либо к числовому типу (в таком случае оно заполняется числовой функцией Recno(), содержательно равной «номеру записи»), либо к словесному типу; однако есть некоторое практическое преимущество в выборе словесного типа поля; в этом последнем случае оно заполняется составной «цепочечной» (словесной) функцией Str(Recno()), содержательно тоже равной «номеру записи», но формально

соответствующей преобразованию этого номера в «цепочечный вид», при котором числу соответствует сочетание небольшого числа начальных пробелов с последовательной конкатенацией всех цифр, входящих в десятичную запись данного числа.

В результате этой операции информация о номере записи, содержавшей данный фрагмент информации в исходной ЛБД, продолжает храниться в производной от неё ЛБД – в частности, и после того, как в качестве главного делимитатора (R-делимитатора) был объявлен некоторый новый делимитатор, не совпадающий с делимитатором D_0.

На следующем этапе происходит вторичное дробление вокабул обрабатываемого словаря. Это вторичное дробление осуществляется с помощью процедуры «дробления» или «декомпозиции» (Decompose) ЛБД.

5. Процедура дробления (декомпозиции) БД (Decompose).

Результат применения процедуры «дробления» к некоторой БД зависит от значения двух параметров: того поля, которое подвергается дроблению, и того символа, который объявлен границей дробления. Производная база состоит из тех же полей, что и исходная.

Однако записей в ней оказывается больше благодаря «дроблению» некоторых («раздробляемых») записей на несколько: каждая запись, содержащая одно или более вхождение границ дробления (для определённости, *n* вхождений границ дроблений), «дробится» на несколько записей. А именно: поле, подвергаемое дроблению, «разрезается» на столько отрезков, сколько образуется из него при рассечении его на части по местам прохождения границ дробления (то есть на *n+1* частей), после чего для каждого из полученных отрезков создаётся соответствующая ему новая («дроблёная») запись.

В каждой из этих «дроблёных» записей значение «дробимого» поля равно тому отрезку, который получается, если «вырезать» из содержания данного поля тот фрагмент, который находится между *i*-м вхождением границы дробления в дробимое поле и *i+1*-м вхождением границы границы дробления в дробимое поле. Значения остальных полей равны соответствующим значениям одноимённых полей в «исходной» (раздробляемой) записи.

В составе «производной» БД любая «нераздробляемая» запись остаётся на своём месте, а любая «раздробляемая» заменяется на *n+1* «дроблёных» записей, получаемых из неё при последовательной замене содержания «дробимого» поля на его отрезки, получаемые в результате рассечения этого поля по границам дробления.

6. Приведение формальной структуры ЛБД к «идеальному» виду

Задача логического расчленения словарной информации в соответствии с её содержательной сложностью требует приведения формальной структуры ЛБД в некоторое (по возможности максимальное) соответствие с её содержательной неоднородностью. Содержательная неоднородность ЛБД проистекает из того, что подавляющее большинство существующих словарей являются многоаспектными. Структура словарной статьи в таких словарях обычно не сводится к логической паре, построенной по схеме «вход - выход», а состоит из нескольких (и нередко даже из нескольких десятков) «зон». Соответственно этому, процесс приведения ЛБД к некоторому логическому «идеалу» должен был бы состоять в том, чтобы для каждого «типа словарной информации» (то есть, в других терминах, для каждой «зоны» словарной статьи, или для каждого «лексикографического параметра») было заведено особое, однозначно соответствующее этому типу «поле».

Процесс приведения ЛБД к такому «логическому идеалу» начинается с того, что в её структуру с помощью операции «Модификация структуры» («Modify structure») добавляются в требуемом количестве (либо все сразу, либо по мере необходимости) новые «поля»: для каждого типа словарной информации вводится своё «поле».

7. Задание информации, обрамляемой делимитаторами

Как было отмечено выше, при стратегии «D_1-F», некоторый тип побочных делимитаторов может быть на некотором этапе объявлен F-делимитатором.

В таком случае информация, содержательно соответствующая некоторому типу, оказывается воплощена в цепочке символов, расположенной между инициальным и финальным делимитатором соответствующей зоны: левая граница соответствующего информационного фрагмента будет совпадать с инициальным делимитатором этой зоны, а правая – с её финальным делимитатором.

Следует помнить, что в качестве лексикографически релевантных делимитаторов в реально

опубликованных словарях могут выступать не только эксплицитные (словесные и буквенные) сигналы, но и разнообразные знаки препинания (двоеточие, тире, кавычки, разные виды скобок, запятая, точка с запятой, точка), а также супрасегментные графические средства – сигналы разнообразных видов текстовых выделений (таких, как жирность, разрядка, подчёркивание, курсив). Поэтому непременным предварительным условием успешности работы над ЛБД является предельная внимательность разработчика ЛБД к супрасегментным графическим средствам при поиске разграничительных сигналов подобного рода и предельная тщательность при разграничении разных видов словарной информации, обрамляемых этими сигналами.

Основная операция, применяемая при переносе информации из одного поля в другое – это стандартная операция логической замены (Replace). Первым параметром этой операции является указание на то, какое именно из уже наличествующих в базе полей подвергается логической замене. Вторым параметром является указание на ту переменную или постоянную величину, на которую требуется заменить наше (заменяемое) выражение. Если это величина переменная, то для её задания мы должны прибегнуть к логическому выражению, содержащему наименования некоторых функций (и, естественно, также наименования постоянных или переменных выражений, являющихся значениями этих функций).

Важнейшие функции, используемые при таком переносе – следующие.

Функция «подцепочки» (Substr), значение которой зависит от значений, принимаемых тремя переменными (аргументами): 1) первым её аргументом является символьное выражение, подвергаемое замене (обычно здесь речь идёт именно о переменных выражениях, а не о константах); 2) вторым её аргументом является указание (константное или переменное) на порядковое (равное некоторому натуральному числу) место того символа, с которого начинается возвращаемая «подцепочка», в составе той цепочки, которая подвергается замене; 3) третьим её аргументом является указание (константное или переменное) на длину возвращаемой «подцепочки».

Функция «позиции» (At), значение которой зависит от двух аргументов – 1) первым её аргументом является некоторое символьное выражение (константное или переменное), предположительно входящее («включённое») в состав некоторого более протяжённого выражения, объемлющего данное; 2) вторым её аргументом является то более протяжённое («включающее») символьное выражение (тоже константное или переменное), которое предположительно включает в свой состав символьное выражение, выполняющее роль первого аргумента. Значение функции «позиция» равно целому числу, равному порядковому месту, занимаемому первым из символов, входящих в состав первого (или единственного) вхождения первого аргумента («включённого» выражения) в состав второго аргумента («включающего» выражения), в составе этого включающего выражения (то есть место первого (инициального) символа включённого выражения по отношению к первому (инициальному) символу этого включающего выражения).

Функция «длины» (Len), указывающая количество символов в некоторой цепочке, носит числовой характер (возвращает целое неотрицательное число).

Функция «количества вхождений» (Howmany), указывающая на то, сколько раз некоторое предположительно «включённое» выражение входит в состав некоторого предположительно более протяжённого (предположительно «включающего») выражения, также возвращает некоторое целое неотрицательное число.

8. Перенос информации, обрамляемой делимитаторами, из одного поля в другое

Если выбирается стратегия «D_1-F» (и некоторый тип побочных делимитаторов на некотором этапе объявлен F-делимитатором), то общая схема выделения релевантного для наших задач фрагмента словарной статьи, сводится к тому, чтобы объявить таким релевантным фрагментом подцепочку словарной статьи, начинающуюся с инициального делимитатора обрабатываемой зоны и имеющую своей длиной число, равное разности между позицией инициального делимитатора линейно следующей за ней зоны и позицией инициального делимитатора обрабатываемой зоны.

В ходе реальной лексикографической работы основную и весьма существенную трудность представляет тот факт, что количество зон, заполненных ненулевым содержанием, весьма сильно колеблется от статьи к статье. Это связано не столько с субъективными факторами (такими, как непоследовательность авторов словаря), сколько с наложением друг на друга двух объективных факторов: (а) онтологического фактора, который связан с неравноценностью словарных единиц; (б) гносеологического фактора, который связан с тем, что метаязыку словарных описаний, стремящихся к максимальному удобству для одушевлённого (являющегося человеком) пользователя, свойственны многие черты естественного языка – в частности, тенденция к экономии усилий, проявляющаяся в опущении избыточной (или обычно ощущаемой как избыточная) информации при повторах.

В связи с этим львиная доля труда разработчика ЛБД тратится на восстановление той «глубинной структуры» словарных статей, которая может быть реконструирована путём «снятия» результатов таких трансформаций, как «сочинительное сокращение» и «опущение повторов при анафорическом эллипсисе».

9. Экстраполяция (перенос информации) из одной записи в другую

Для экстраполяции (переноса информации из одной записи в другую) удобнее всего использовать операцию «логической замены» (Replace) в сочетании с функцией «содержание записи» (Record).

Чаще всего некоторое содержание должно быть перенесено из исходной записи в запись, соседнюю с ней (причём чаще в «последующую», чем в «предыдущую» запись) или в непрерывный ряд таких соседних записей (причём чаще в «последующий», чем в «предыдущий» ряд).

Прямая (инерционная) экстраполяция, переносящая информацию из предыдущей записи в последующую, осуществляется несколько проще, чем обратная (предвосхищающая) экстраполяция. Однако такая несимметричность этих двух типов операций может считаться вполне обоснованной. Она хорошо согласуется с тем фактом, что при подаче информации в словарях (в полном соответствии с основной тенденцией подачи информации в связных текстах) ретроспективная отсылка (или «собственно анафора», отсылка назад) представляет собой гораздо более распространённое явление, чем проспективная отсылка (или «катафора», отсылка вперёд). В определённой (хотя и несколько меньшей) степени сходная несимметрия присутствует и в технике сочинительного сокращения, широко используемого при написании словарных статей (а именно, «вынесение общего члена за скобку» чаще реализуется в виде препозиции этого общего члена, чем в его постпозиции).

Перенос информации в последующую запись (или в непрерывный ряд последующих записей) осуществляется подстановкой значения некоторого сложного выражения (вершиной которого является наименование функции Record, первым аргументом – число, равное «Recno()-1», то есть номеру данной записи, из которого вычтена единица, а вторым аргументом – имя того поля, которое содержит интересующую нас информацию, подлежащую «переброске» из одной записи в другую) в качестве значения одного из параметров процедуры «логической замены» (Replace) – а именно, в качестве значения параметра «подставляемое выражение» (= «на что заменить»).

Экстраполяция информации на целый непрерывный ряд последующих друг за другом записей может осуществляться чисто автоматически (что и делается «по умолчанию», если нет специальной необходимости в запрете такой множественной экстраполяцию), так как при переходе к обработке содержания последующей записи содержание предыдущей записи уже является «обновлённым».

Однако в силу той же причины экстраполяция информации «назад» (от последующих записей к предыдущим) по умолчанию оказывается лишь одношаговой. Поэтому при необходимости произвести «обратную» («предвосхищающую») экстраполяцию более чем на один шаг назад («множественную обратную экстраполяцию») наиболее удобным практически оказывается приём «двойной вертикальной реверсии» ЛБД. Он состоит в следующем:

1. На основе исходной ЛБД с помощью операции «сортировки» (Sort) по параметру «-Recno()» создаётся производная от неё («вертикально перевёрнутая») ЛБД, в которой набор и число записей – те же, что в исходной ЛБД, но которая при этом отличается от исходной ровно одним признаком: а именно, тем, что все записи в ней идут в порядке, обратном исходному.

2. К полученной таким образом «вертикально перевёрнутой» ЛБД применяется операция «прямой (инерционной) экстраполяции».

3. К полученному результату вторично применяется операция «сортировки» по параметру «-Recno()». Таким образом, ЛБД ещё раз подвергается вертикальному перевёртыванию.

Результатом такого трёхшагового применения операций является производная ЛБД, отличающаяся от исходной ЛБД ровно одним признаком: в ней осуществлена множественная обратная экстраполяция.