Уфа : УГАТУ, 2007 Вестник УГАТУ • Управление, ВТ и И Т. 9, № 2 (20). С. 98–105

УДК 004.4

Ю. С. КАБАЛЬНОВ, С. В. МАКСИМОВ М. Б. КАЛЕНТЬЕВА

КОНТЕКСТНО-СЛОВАРНОЕ СЖАТИЕ ТЕКСТОВОЙ ИНФОРМАЦИИ НА ОСНОВЕ ЛЕКСИЧЕСКИХ ПРАВИЛ

Предложен путь улучшения сжатия текстовой информации на основе лексических правил и применение контекстно-словарного сжатия. Тем самым открывается возможность применения предложенных моделей для создания поисковых систем нового поколения. Показано, что применения лексических правил даёт возможность семантического анализа содержания текстов, смыслового поиска информации, формирования электронных архивов. *Сжатие текстовой информации*; *контекстно-словарное сжатие*; *сжатие на основе лексических правил*

ВВЕДЕНИЕ

В ряде случаев при организации хранения и передачи информации приходится иметь дело с текстовой информацией. Примером этому могут быть различного рода поисковые системы (Rambler, Yandex, Google и т.д.), системы хранения специализированной текстовой информации для дистанционного обучения, речевого общения пользователя с компьютером и т.д.

В этих случаях, для повышения эффективности хранения и для передачи по компьютерным сетям больших объемов текстовой информации используется ее сжатие.

Известные на сегодняшний день методы сжатия тестовой информации можно разделить на три группы: словарные методы, контекстные методы, статистические методы [1, 2, 4].

Недостатками первой группы методов являются:

- зависимость скорости распаковки от размера словаря;
- низкая эффективность сжатия разнородных данных;
- показатели кодирования проигрывают декодированию, так как тратится время на поиск фраз возможной наибольшей длины;
- семантическая неполнота состоит в том, что словарь содержит только текущие фрагменты сжимаемого объекта, что, с одной стороны, приводит к уменьшению размера словаря, а, с другой стороны, к потере информации о возможных объектах.

Недостатками второй группы методов являются:

- скорость сжатия уступает словарным методам;
- медленное декодирование;
- несовместимость кодера и декодера в случае изменения алгоритма;
- медленная обработка мало избыточных данных;
- большие запросы на память;
- проигрыш в эффективности сжатия для длинных повторяющихся блоков, по сравнению со словарными методами.

Недостатками третьей группы методов являются:

- использование в качестве объектов сжатия слабо структурируемых цепочек символов (двоичных кодов);
- не используется информация о правилах словообразования, характерных для естественных языков, что снижает точность определения условных вероятностей появления тех или иных морфем.

Общим недостатком известных методов сжатия является то, что объекты сжатия не рассматриваются как системные объекты, представленные в виде совокупности элементов (морфем), связанных между собой с правилами словообразования естественного языка.

На наш взгляд, одним из возможных резервов повышения эффективности сжатия текстовой информации является предварительная структуризация сжимаемой текстовой информации на основе знаний о правилах словообразования естественных языков.

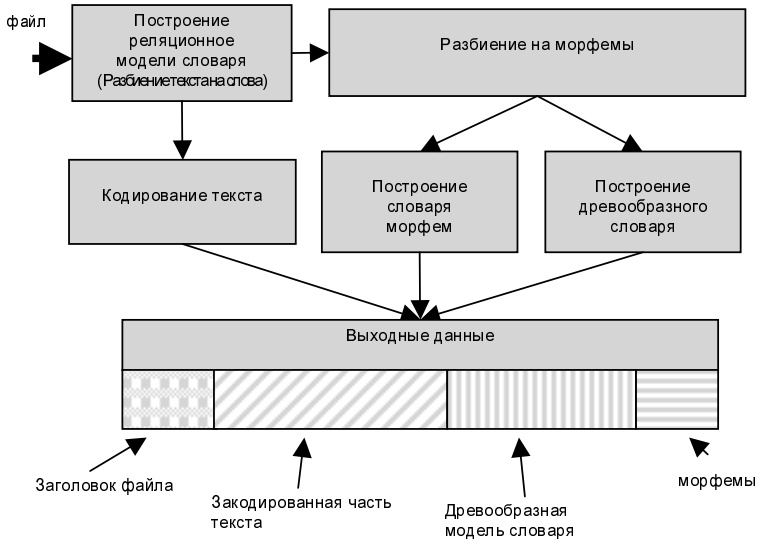

Рис. 1. Укрупненная структурная схема процесса контекстно-словарного сжатия текстовой информации

1. КОНТЕКСТНО-СЛОВАРНАЯ МОДЕЛЬ СЖАТИЯ ТЕКСТОВОЙ ИНФОРМАЦИИ

Как известно, текстовая информация сама по себе уже изначально содержит избыточность, это связано с особенностями языка, на котором мы разговариваем, читаем и пишем. Основой для ее представления является алфавит, используя который составляем различные слоги, слова, предложения, т. е. комбинируя и переставляя буквы алфавита, получаем необходимую информацию. Но информация имеет большой объем, и, с точки зрения экономного использования памяти компьютера, в таком виде хранение ее нецелесообразно. Для определения размера и способов хранения информации существуют формулы, позволяющие рассчитать выделение необходимого ее количества для хранения данных. К сожалению, приведенные формулы могут быть использованы для всех случаев, но они не ориентированы на частные случаи. Используя определенные особенности для каждой формы представления информации, можно получить лучший способ, чем классический.

Учитывая, что сжатие предусматривает замену часто используемых символов короткими кодами, объем информации можно уменьшить до 30–50%.

Как известно, текст на естественных языках является слабоструктурированной информацией. Исследования сжатия текстов показали, что высокую степень сжатия для текстовых файлов невозможно получить по следующим причинам:

• редко встречаются повторяющиеся последовательности вида «ааааааааа»;

• необходимо рассматривать не только отдельный символ, но и группы символов, образующих слова;

• для текстов на естественных языках длина повторяющихся последовательностей обычно не превышает 20 символов.

На рис. 1 показано, что над текстовой информацией необходимо провести следующие преобразования: построить реляционную модель словаря; разбить на морфемы. Для построения словаря необходимо разбить текст на слова (для разбиения на слова используются разделительные знаки), полученные слова расчленить на морфемы (используются аналогичные правила как на естественных языках). Реляционную модель словаря используем для кодирования теста, полученный словарь морфемы используется для построения древообразной модели словаря и кодирования ее с помощью словаря морфем. Узлы древообразного словаря являются ссылками на словарь морфем. Полученные данные сохраняются в той последовательности как показано на рис. 1.

Процесс сжатия текста выглядит следующим образом.

1) Построение реляционной модели словаря исключением повторов.

2) Сортировка по частоте встречаемости и длине.

3) Построение кода.

4) Кодирование текста с учетом особенностей построенного кода словаря.

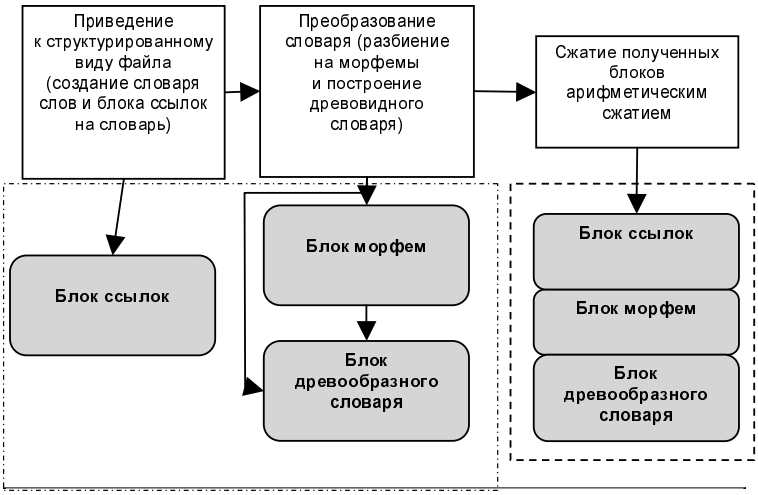

Рис. 2. Формирование компонентов модели контекстно-словарного сжатия текстового файла

Эффективный код сжатого текста зависит от того, насколько объективно оценивается частота появления слов. Чем чаще появляется слово, тем короче будет предоставлен для него код. Каждому хранящему слову присваивается код в зависимости от частоты появления. Это означает, что у самого часто встречаемого слова будет самый короткий код. Полученный код сохраняется в выходном файле. Сохранение кода слова вместо самого слова также позволяет сократить объем закодированного текста.

Эффективный код сжатия также зависит от длины слова, сохраняемого в словаре. Максимальная длина слова может дать выигрыш в случае, если код слова, используемый для кодирования, станет длиннее длины кодируемого слова.

2. ДРЕВОВИДНАЯ ЛОГИЧЕСКАЯ МОДЕЛЬ ПОПОЛНЯЕМОЙ БАЗЫ МОРФЕМ

Представление словаря в виде иерархической структуры является наиболее важным в данной работе. Используя языковые особенности и существующие ограничения, можно заметить, что при составлении слов прослеживается иерархическая структура [5].

Хранение словаря в виде реляционной базы является неэффективным. Она пригодна для замены контекстами кодируемого текста, но не пригодна для хранения.

Преобразование реляционного словаря в иерархическую структуру предлагается выполнить следующим образом:

1) Разложить слово на морфемы. Полученные морфемы по необходимости добавлять в словарь морфем, либо просчитывать количество совпавших морфем для получения оптимального кода для контекста.

2) Полученные индексы морфем в порядке появления в слове присоединяют в древовидный словарь. Каждая морфема образует узел соответствующего уровня.

3) В случае совпадения морфем соответствующего уровня, осуществляется переход на следующий уровень. В случае отсутствия добавляется новая ветка.

4) После окончания обработки каждой ветке присваивается соответствующий контекст.

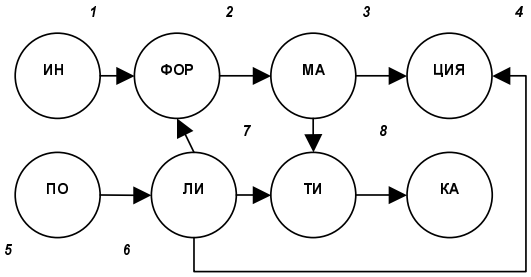

Рис. 3. Объектно-когнитивная модель представления словаря

Предлагаемая нами модель иерархического словаря приведена на рис. 4. Она включает в себя:

- элементы (узлы) словаря, которые являются ссылками на словарь морфем;
- ребра, которые позволяют реконструировать слово из морфем.

Окончанием ветки словаря являются ссылки на знаки препинания, которые имеют код для кодирования текста.

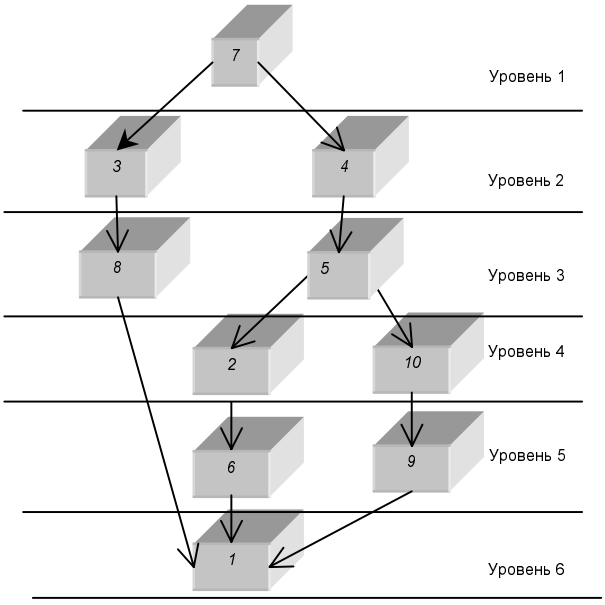

Рис. 4. Древовидная модель хранения морфем

Реляционная модель словаря морфем (табл.) представлена в виде 4-местного кортежа $R = \{r_1, r_2, r_3, r_4\}$, где r_1 — порядковый номер (ключ отношения), r_2 — морфема, r_3 — двоичный код переменной длины, r_4 — частота встречаемости в словаре.

Таблица

Структура пополняемого словаря морфем

r_1	r_2	r_3	r_4
1	,_	1	3
2	с	10	3
3	е	11	1
4	жа	100	1
5	жи	101	1
6	ма	110	1
7	мое	111	1
8	тие	1000	1
9	щие	1001	1
10	ю	1010	1

Реляционная модель словаря морфем и древовидная модель хранения морфем в памяти хранится в упорядоченном виде, для того чтобы была возможность быстрого поиска слов в тексте при кодировании. В качестве алгоритма сортировки выбрана «быстрая сортировка». Использование дополнительной таблицы (хеш-таблицы) к словарю позволит организовать контекстно-индексированный поиск. Структурная модель хеш-таблицы выглядит следующим образом: таблица состоит из 256 ячеек, где определены начало позиции группы в словаре и количество элементов в этой группе. Поиск по словарю производится алгоритмом бинарного поиска (дихотомия).

Элементами (узлами) словаря являются ссылки на словарь морфем. Это связано с тем, что каждое слово в словаре составляется из последовательностей морфем. Морфемы желательно хранить отдельно от модели словаря. Количество различных вариантов морфем, которые встречаются в тексте, значительно меньше слов. Длина морфемы короче длины слова, так как слова составляются из морфем. Поиск и построение словаря заметно ускоряется, так как определяется на уровне разложения слова на морфемы.

Определяя окончание ветки словаря ссылками на знаки препинания, закладываем основу для восстановительного процесса. Это позволит воссоздать реляционную базу словаря для восстановления сжатого текста.

Древообразная форма представления позволяет исключать из словаря еще на первом уровне дерева повторяющиеся слоги, которые используются при составлении слова.

3. АЛГОРИТМЫ КОНТЕКСТНО-СЛОВАРНОГО СЖАТИЯ ДАННЫХ НА ОСНОВЕ ПРЕДЛОЖЕННЫХ МОДЕЛЕЙ

Анализ проводился в трех направлениях: повышение эффективности сжатия, убыстрение работы алгоритма и осуществление сжатия на основании новой системы контекстов.

Одним направлением для исследований является подгонка схем сжатия к естественным языкам. Сегодняшние системы работают полностью на лексическом уровне. Использование больших словарей с синтаксической и семантической информацией может позволить получить преимущества от имеющейся в тексте высокоуровневой связности. Поэтому специальные алгоритмы сжатия, взятые в расчете на лингвистическую информацию более высокого уровня, будут, несомненно, зависеть от системной сферы. Вероятно, что поиски методов улучшения характеристик сжатия в данном направлении будут объединяться с исследованиями в области контекстуального анализа по таким проблемам, как извлечение ключевых слов и автоматическое абстрагирование.

Второй подход диаметрально противоположен описанному выше направлению и состоит в поддержании адаптивности системы и поиске улучшений в имеющихся алгоритмах. Необходимы лучшие пути организации контекстов и словарей. Например, еще не обнару-

жен пригодный метод построения моделей состояний; метод DMC (динамическое сжатие модели Маркова) — лишь форма контекстно-ограниченной модели. Другим направлением исследований является метод выделения кодов ухода в частично соответствующих контекстуальных моделях. В итоге, преобразование систем, определяемых состояниями, таких как скрытые модели Маркова, может дать улучшение для сжатия текстов.

В результате проведенного анализа существующих алгоритмов было установлено, что ни один из известных методов не обеспечивает одновременно высокую степень сжатия и высокую скорость распаковки, что затрудняет их использование в целом ряде практических областей. Актуальной является задача разработки методов сжатия информации без потерь, обеспечивающих одновременно высокую степень сжатия информации и высокую скорость распаковки с учетом предыдущих замечаний.

Предлагаемый алгоритм сжатия чисто текстовой информации основан на эмпирическом законе Ципфа: частота появления слов, отсортированных по частоте употребления, обратно пропорциональна их номерам; другими словами,

$$P_k \approx \frac{1}{k}\frac{1}{S}, \qquad (1)$$

где

$$S \equiv \sum_{k}^{N} \frac{1}{k}. \qquad (2)$$

Закон Ципфа выполняется для многих языков (английского, французского, немецкого, испанского, русского, китайского), невзирая на их существенные различия в грамматике и синтаксисе.

В основе алгоритма сжатия лежит композиция (агрегирование) более простых процедур: замена ссылками на тезаурус сжимаемого текста и построение тем самим тезауруса; организация сжатия тезауруса путем построения его древовидной модели, вершинами которой является реляционные модели морфем.

Применение алгоритмов поиска совпадающих последовательностей бинарных кодов для объектов контекстно-словарного сжатия (ссылки на тезаурус, древовидная модель тезауруса, словарь морфем) используется для формирования оптимальных последовательностей сжатых бинарных кодов указанных объектов. Показано, что сложность разработанного алгоритма растет нелинейно относительно длины входной последовательности символов.

Возможность использования разработанных алгоритмов в сочетании с алгоритмами поиска на основе хеш-таблиц позволяет достичь максимальной скорости поиска совпадающих последовательностей бинарных кодов для объектов контекстно-словарного сжатия.

Алгоритм декомпозиции исходной текстовой информации выглядит следующим образом.

Пусть $A = \{a_1, \dots, a_l\}$ — некоторый алфавит, используемый для записи слов. Имеется последовательность символов $x \in A_N$:

$$x = a_{i_0}, ..., a_{i_{N-1}};$$

последовательность (поток $T = \{x\}$) текущих текстовых символов, образующих слова; $G = \{a_{k-m}, \dots, a_k, \dots, a_{k+m}\}$ — множество знаков препинания и служебных разделителей, входящих в алфавит A.

Множество $S = \Big\{ s[i] \subset \{x \in AN\}, \mathrm{count} \in \in N, \mathrm{Code} \in N \Big\}$ будем называть словарем.

Преобразование $F(x) : T \to S$ будет описывать процедуру структуризации исходной текстовой информации. В процессе структуризации элементам $s[i]$ ставится в соответствие последовательность элементов потока T, окончанием которого являются элементы множества G.

Сжатие потока T будем описывать с помощью преобразования

$$K(x) : T \to T'(x \to S.\mathrm{Code}),$$

где count — частота появления $s[i]$ в словаре; Code — код, присвоенный после сортировки по убыванию частоты; T' — поток, состоящий из ссылок на словарь, который является контекстом (ссылки на словарь), описывающим поток T.

Ниже представлены блок-схемы алгоритмов структуризации информации, реализующие указанные выше преобразования $F(x) : T \to S$, $K(x) : T \to T'(x \to S.\mathrm{Code})$.

Полученный словарь в ходе преобразования неудобно хранить. Во-первых, он получается слишком большим и громоздким. Во-вторых, в нем встречается много повторов, которые отличаются всего несколькими символами. Можно заметить, что существует возможность уменьшения объема словаря. Для этого необходимо:

1) Разбить слова на морфемы.

2) Представить словарь в древовидной форме на основе проведенных ранее разбиений.

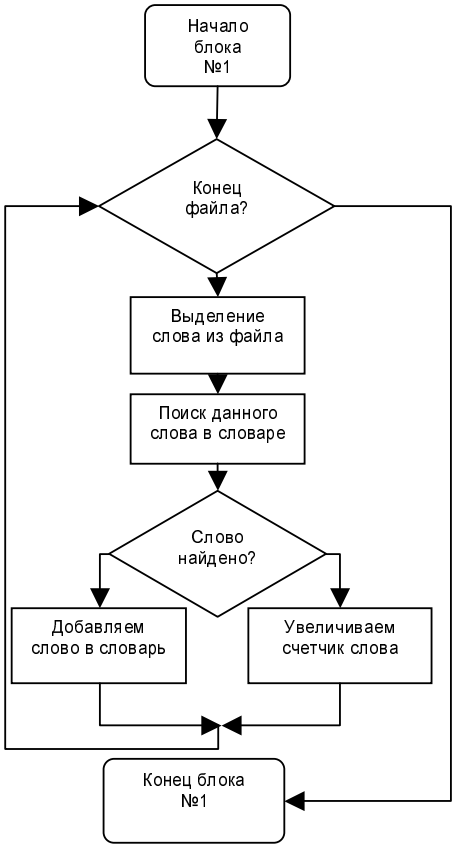

Рис. 6. Выявление структуры и построение модели $(F(x) : T \to S)$

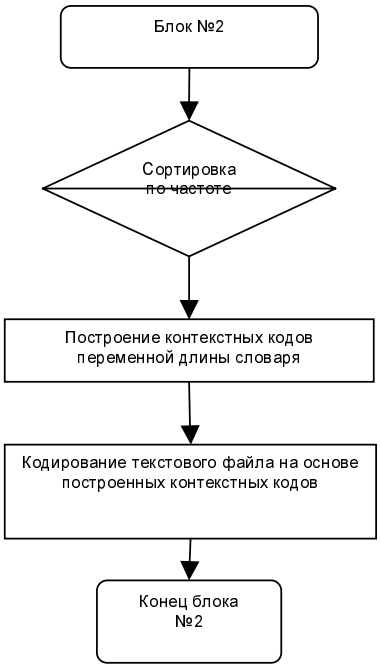

Рис. 7. Построение и проведение кодирования на основе выявленных структур $(K(x) : T \to T'(x \to S.\mathrm{Code}))$

Разбиение слова на морфемы можно производить одним из двух предложенных способов:

- разбиение слова на слоги;
- разбиение на составные части слова (морфемы).

Разбиения на слоги предлагается выполнить следующим образом.

Возьмем для примера алфавит русского языка $A = \{\mathrm{А}, ..., \mathrm{Я}, \mathrm{а}, ..., \mathrm{я}\}$. Буквами обозначаем звуки, которые произносим во время разговора. Звуки делятся на глухие и звонкие, по аналогии со звуками, буквы также делятся на два типа:

- гласные {'А', 'Е', 'Ё', 'И', 'О', 'У', 'Ы', 'Э', 'Я', 'Ю', 'а', 'е', 'ё', 'и', 'о', 'у', 'ы', 'э', 'я', 'ю', 'ь', 'ъ'};
- согласные {'Ц', 'К', 'Н', 'Й', 'Г', 'Ш', 'Щ', 'З', 'Х', 'ф', 'В', 'П', 'Р', 'Л', 'Д', 'Ж', 'Ч', 'С', 'М', 'Т', 'Б', 'ц', 'к', 'н', 'й', 'г', 'ш', 'щ', 'з', 'х', 'ф', 'в', 'п', 'р', 'л', 'д', 'ж', 'ч', 'с', 'м', 'т', 'б'}.

Зная, что за согласной буквой должна следовать гласная, легко можно выделить из слова слоги.

Например: «машина» → «ма–ши–на».

В данном случае воспользовались правилом «сг». Для большинства слов, встречающихся в русском языке, применимо данное правило, но бывают исключения:

- появление в слове сразу нескольких гласных («сггс», «гсггсг»);
- появление в слове сразу нескольких согласных («гсссг», «ссссгссг»).

Частота появления таких комбинаций, как выяснилось в ходе исследования, в русском языке встречается не часто, поэтому данная проблема решается следующим образом.

Необходимо выявить все возможные комбинации, которые встречаются. Полученные комбинации добавляются в базу знаний. База знаний позволяет выявлять из существующих комбинаций все возможные комбинации, и выбирается та, которая наиболее лучше подходит для исследуемого слова.

Алгоритм выделения слогов выглядит следующим образом:

1) Вводится обрабатываемое слово.

2) Преобразуется в последовательность из «с» «г».

3) Производится поиск в базе знаний, если указанного слога нет в базе, слово заносится в список нераспознанных слогов. Данный список позволит выявить типы слогов, не заложенные в базу знаний. Само слово сохраняется в списке слогов неизменно. Это условие

необходимо для возможности обратить сжатую последовательность.

4) Определяются индексы начала и конца.

5) Полученный слог заносится в словарь слогов; если найденный слог существует, то увеличивается счетчик слог.

6) Повторяем шаги 3–5, пока не достигнем конца слова.

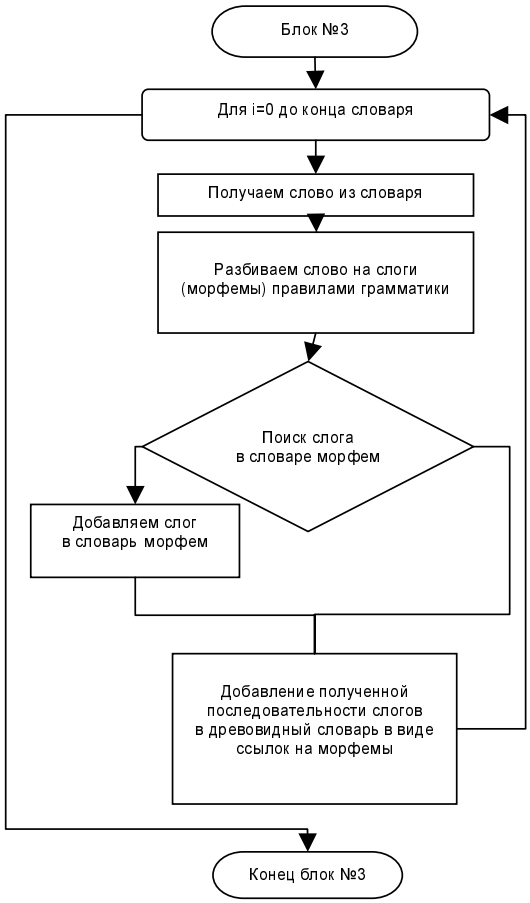

Рис. 8. Блок-схема алгоритма получения древовидной модели словаря

Шаблон, по которому определяются правила разбиения на слоги, назначается следующим образом. Обозначим через «с» — согласные, «г» — гласные. Шаблон составляется так, чтобы в нем был небольшой избыток информации, в котором указывается дополнительная информация всевозможных следующих пар слогов, которые могут следовать после распознанного слога. Например, для получения слога «сг» указываем следующий шаблон «сгсг». Для правильного определения основной части слога дополнительно задается позиция значимой части шаблона, который необходим для получения правильной части слога, например, в данном случае 2.

Следовательно, формат шаблона (правила) представляет собой запись. В него входят следующие поля:

Rule[20]: правила «сг»;

I: количество символов, которые являются значимыми для выделения морфема из поля Rule.

Если слово заканчивается знаком препинания, то в конце ставим «п». Процесс отделения знаков препинаний происходит в самом начальном этапе распознавания морфем.

Другой метод для преобразования словаря — деление слов по составу, т.е. на приставки, корни, суффиксы и окончания. Данный метод обусловлен тем, что в языке встречаются и однокоренные слова, поэтому корни разных слов мы можем заменить одним и тем же индексом. Это помогает уменьшить размеры слов, следовательно, и словаря. Чем больше букв в любой части слова, тем лучше, так как целую часть слова мы заменяем одним индексом [3] .

Таким образом, получаем матрицу, состоящую из приставок, корней, суффиксов и окончаний, которые содержатся в словаре. Для уменьшения объема получаемого файла, при сжатии словаря, создаются готовые списки приставок, суффиксов и окончаний, которые часто встречаются. Каждый элемент списка имеет индекс, который в дальнейшем будет являться кодом. Используя полученные коды, преобразуем словарь. Слова делятся, используя следующее простое правило.

Проверяется длина слова. Если слово состоит из пяти или менее букв, определяем его как корень исследуемого объекта. Оно заносится в список корней. В случае, если длина исследуемого слова превышает пять символов, пытаемся выделить приставку, затем окончание и, наконец, суффикс. И то, что осталось, определяется как корень. При условии, если в слове две приставки или два суффикса, то они объединяются и запоминаются как один. Полученное словосочетание заносят в список приставок (суффиксов). После преобразования словаря на части слова (приставку, корень, суффикс и окончание) элементы словаря заменяются в соответствующие индексы. Структура матрицы индексов хранится в виде таблицы, столбцами которой являются: приставки, корни, суффиксы, окончания. Записями являются правила, по которым строятся слова. Пустые элементы означают отсутствие данного раздела у слова. Таблица сохраняется по столбцам для исключения повторов и пустых элементов.

ЗАКЛЮЧЕНИЕ

В работе показана целесообразность использования системной формы типа «сущность-связь» при сжатии текстовой информации на естественных языках. Это позволяет за счет знаний правил словообразований, принятых в том или ином естественном языке, принципиально обеспечить возможность достижения более высокой степени сжатия текстовой информации. Предложены:

• системные модели контекстно-словарного сжатия текстовой информации, реализующие декомпозиционные процедуры сжатия текстовых данных, обладающие более высоким «качеством» сжатия;

• древовидная модель процедуры словообразования, упрощающая получение нужных символьных конструкций на этапах сжатия и восстановления слов;

• реляционная модель хранения морфем в виде 4-местного кортежа, включающего порядковый номер (ключ отношения), морфему, двоичный код переменной длины и частоту встречаемости морфемы в подмножестве предложений естественного языка, соответствующих рассматриваемой предметной области;

• алгоритм контекстно-словарного сжатия текстовых данных, базирующийся на предложенных моделях текстовых данных. В основе алгоритма лежит композиция (агрегирование) более простых процедур: замена ссылками на тезаурус сжимаемого текста и построение тем самым тезауруса; организация сжатия тезауруса путем построения его древовидной модели, вершинами которой являются реляционные модели морфем.

На основе разработанных метода и алгоритмов контекстно-словарного сжатия текстовой информации создан программный комплекс MSV Quick Reader, используемый в настоящее время на кафедре программирования и вычислительной математики Башкирского государственного педагогического университета. Экспериментальные исследования эффективности данного комплекса показали, что он обеспечивает увеличение на 5–7% степени сжатия текстовых данных по сравнению с известными методами и, как следствие, снижение на такую же величину трафика учебной информации в компьютерных сетях.

СПИСОК ЛИТЕРАТУРЫ

1. **Кабальнов, Ю. С.** Сжатие информации использованием статистических прогнозирующих моделей / Ю. С. Кабальнов, С. В. Максимов, И. В. Павлов // Проблемы техники и технологий телекоммуникаций : матер. V междунар. науч.-техн. конф. Самара, 2004. С. 154–156.
2. **Максимов, С. В.** Алгоритм передачи данных в телекоммуникационных системах / С. В. Максимов // Вестник МаГУ : периодич. научн. журн. Вып. 5. Естественные науки. Магнитогорск : МаГУ, 2004. С. 330.
3. **Кабальнов, Ю. С.** Сжатие текстовых данных с учетом особенностей словообразования в русском языке / Ю. С. Кабальнов, С. В. Максимов // Ученые записки : сб. науч. статей. Вып. 7. Уфа : БГПУ, 2005. С. 238–241.
4. **Кабальнов, Ю. С.** Модели и алгоритмы сжатия информации применением цепей Маркова / Ю. С. Кабальнов, С. В. Максимов, С. В. Павлов // CSIT-2005 : тр. междунар. конф. Уфа, 2005. Ч. 1. С. 85–91. (На англ. яз.).
5. **Максимов, С.В.** Древовидная модель словаря представления слов / С. В. Максимов // ЭВТ в обучении и моделировании : сб. науч. тр. / Бирск, 2005. Ч. 2. С. 201–205.

ОБ АВТОРАХ

Кабальнов Юрий Степанович, проф., зав. каф. информатики. Дипл. инж. электронной техники (УАИ, 1971). Д-р техн. наук по управлению в технических системах (УГАТУ, 1993). Иссл. в обл. адаптивного и интеллектуального управления.

Максимов Сергей Владимирович, ст. преп. той же каф. Дипл. преп. математики и информатики (БГПИ, 1993). Канд. техн. наук по мат. и прог. обеспеч. выч. машин, комплексов и комп. сетей (УГАТУ, 2006). Иссл. в обл. сжатия информации.

Калентьева Маргарита Борисовна, ассист., аспирантка той же каф. Дипл. преп. математики и информатики (БГПУ, 2005). Готовит дис. по моделям и алгоритмам семантического сжатия информации.