САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

На правах рукописи

Лифшиц Юрий Михайлович

АЛГОРИТМЫ И АНАЛИЗ ТРУДОЕМКОСТИ ОБРАБОТКИ СЖАТЫХ ТЕКСТОВ

05.13.17 — Теоретические основы информатики

А В Т О Р Е Ф Е Р А Т

диссертации на соискание ученой степени
кандидата физико-математических наук

Санкт-Петербург — 2007

Работа выполнена в лаборатории математической логики
Санкт-Петербургского отделения
Математического института им. В.А.Стеклова РАН

Научный руководитель: член-корреспондент РАН, профессор
Матиясевич Юрий Владимирович

Официальные оппоненты: доктор физико-математических наук,
профессор Романовский Иосиф
Владимирович

кандидат физико-математических наук,
Шень Александр Ханьевич

Ведущая организация: Вычислительный центр
им. А.А. Дородницына
Российской академии наук

Защита состоится "___" _____ 2007 г. в ___ часов на заседании диссертационного совета Д 212.232.51 по защите диссертаций на соискание ученой степени доктора наук при Санкт-Петербургском государственном университете по адресу: 198504, Санкт-Петербург, Старый Петергоф, Университетский пр., 28.

С диссертацией можно ознакомиться в научной библиотеке им. М. Горького Санкт-Петербургского государственного университета по адресу: 199034, Санкт-Петербург, Университетская наб., 7/9.

Автореферат разослан "___" _____ 2007 г.

Ученый секретарь
диссертационного совета,
доктор физ.-мат. наук,
профессор Мартыненко Б. К.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. В последние пятнадцать лет много усилий было направленно на разработку такого представления данных, при котором, по возможности, одновременно минимизируются и размер, и время доступа. Одним из наиболее перспективным подходов к этой проблеме является построение алгоритмов поиска в сжатых текстах, которые бы не требовали полной распаковки исходного файла [3, 4, 7, 13, 16, 18]. Очень быстро от рассмотрения конкретных алгоритмов архивирования исследователи перешли к общей модели сжатого текста — *прямолинейной программе*. Неформально, прямолинейная программа является грамматикой, которая порождает ровно один текст. Оказалось, что тексты, сжатые практическими архиваторами (например, LZ77 [22]), быстро и без значительного увеличения размера могут быть переведены в грамматику, описывающую тот же текст [21].

Опишем кратко наиболее важные результаты по обработке текстов, представленных в виде прямолинейных программ. Задача о равенстве двух сжатых текстов впервые была решена в работе [19] в 1994 году. Была доказана оценка $\mathcal{O}(n^4)$ на время работы этого алгоритма, где n равно сумме размеров прямолинейных программ, порождающих два сравниваемых текста. Алгоритм для поиска сжатой подстроки в сжатом тексте впервые появился в 1995 году в статье [14]. Вслед за этим удалось расширить алгоритм для вычисления различных комбинаторных свойств текста [9]. Далее, в 1997 году Миязаки, Шинохара и Такеда [17] построили алгоритм, требующий $\mathcal{O}(n^2m^2)$ времени для поиска подстрок, где m и n — размеры сжатого шаблона и сжатого текста, соответственно.

В то же время, ряд классических текстовых задач до сих пор не был решен в постановке со сжатым представлением входных данных. В частности, не было предложено ни одного алгоритма поиска подпоследовательностей напрямую в сжатом тексте. Определение вложимости сжатого шаблона в сжатый текст могло оказаться как решаемым за полиномиальное

время, так и PSPACE-полной задачей. Также, оставалась неизвестной сложность вычисления расстояния Хэмминга между сжатыми текстами. Этот вопрос является самой простой формой приближенного поиска подстрок, который полезен для анализа как биологических данных, так и медиафайлов. Кроме того, вычисление расстояния Хэмминга между сжатыми текстами является естественным продолжением задачи о равенстве сжатых текстов.

Алгоритмы на сжатых текстах имеют прямое отношение к ряду других задач теоретической информатики. Так, с их помощью впервые удалось построить полиномиальный по памяти алгоритм решения уравнений в словах [20], а также полиномиальный алгоритм верификации диаграмм сообщений [10].

Быстрый поиск по сжатым данным имеет и прикладное значение. Архивирование индексов поисковых систем важно для интернет-приложений, коллекций аудио и видео, биологических баз данных. Второй областью приложения является верификация программ и микросхем. Обычно для верификации необходимо проверить некое свойство множества допустимых состояний программы и графа переходов. Для современных систем число состояний не поддается никаким переборным алгоритмам. Естественный выход — хранить его в неявном виде и проверять его свойства без явного порождения.

Цели работы. Диссертационное исследование было направлено на решение следующих основных задач:

1. Построить новые алгоритмы для сравнения сжатых текстов, поиска сжатых шаблонов в сжатых текстах, вычисления периодов сжатых текстов. Упростить алгоритмы, представленные в работах [9, 12, 14, 17, 19] и/или уменьшить верхние оценки на время их работы.

2. Определить существование полиномиальных алгоритмов для следующих задач: вложимость явно заданного шаблона в сжатый текст, вложимость сжатого шаблона в сжатых текст, вычисление расстоя-

ния Хэмминга между сжатыми текстами, вычисление минимального накрытия сжатого текста.

3. Построить систему компактного описания текстов, основанную на использовании частично определенных слов.

Общая методика работы. В диссертации используются идеи, хорошо известные в рамках теоретической информатики. Представленные алгоритмы основаны на методе динамического программирования и используют ряд комбинаторных свойств текстов. Оценки трудоемкости получены путем сведений общепризнанно-трудных задач [1] к рассматриваемым проблемам. Алгоритм поиска разреженных периодов минимального размера использует вариацию поиска кратчайших путей в ациклических графах.

Основные результаты.

1. Разработаны алгоритмы поиска сжатых подстрок в сжатых текстах, вычисления минимальных периодов и накрытий сжатого текста, поиска явно заданной подпоследовательности в сжатом тексте.

2. Доказана #P-полнота задачи о вычислении расстояния Хэмминга между сжатыми текстами, NP- и coNP-трудность задачи о поиске сжатой подпоследовательности в сжатом тексте.

3. Предложено понятие разреженной периодичности. Найдено соотношение примитивного классического и примитивного разреженного периодов.

4. Разработан алгоритм поиска разреженных периодов минимального размера.

Научная новизна. Все основные результаты являются новыми.

Практическая и теоретическая ценность. Представленные алгоритмы для проверки равенства сжатых текстов и поиска сжатых подстрок в сжатых текстах могут быть полезны для проверки эквивалентности программ в рамках модели, предложенной в работе [15]. Описание текстов с

помощью их разреженных периодов может быть использовано для обобщения ряда классических методов архивирования, таких как *кодирование длин серий* (RLE). Доказательство трудности вычисления расстояния Хэмминга между сжатыми текстами показывает границы применимости алгоритмов для обработки текстов, представленных в виде прямолинейных программ. Фактически, можно сделать вывод, что для эффективного *приближенного* сравнения сжатых текстов нужна другая модель компактного хранения.

Апробация работы. Основные результаты обсуждались на следующих конференциях и семинарах:

- Международный симпозиум Mathematical Foundations of Computer Science, Словакия, 2006.
- Международный симпозиум Computer Science in Russia, Санкт-Петербург, 2006.
- Школа-семинар "Синтез и сложность управляющих схем", Санкт-Петербург, 2006.
- Школа-семинар в Дагштуле "Combinatorial and Algorithmic Foundations of Pattern and Association Discovery", Германия, май 2006.
- Русско-французская конференция молодых ученых, Москва, 2006
- Научные семинары ПОМИ РАН, СПИИРАН, МГУ, ИППИ РАН, университетов Таллина и Турку.

Результаты, лежащие в основе диссертации, дважды представлялись на конкурс Мебиуса. В 2005 году работа диссертанта стала финалистом конкурса, в 2006 году — отмечена жюри.

Публикации. Основные результаты диссертации опубликованы в пяти работах [23, 24, 25, 26, 27], перечисленных в конце автореферата.

Работы [23, 24] опубликованы в издании, входившем в перечень ВАК на момент публикации.

Структура и объем работы. Диссертация объемом 82 страницы состоит из введения и четырех основных глав, разбитых на разделы и подразделы. Список цитируемой литературы состоит из 47 наименований.

СОДЕРЖАНИЕ РАБОТЫ

Во **введении** приводится обоснование актуальности темы диссертации, формулируются основные полученные результаты, поясняется их положение в контексте текущих исследований, а также кратко описывается структура диссертации.

Вторая глава содержит описание прямолинейных программ, которые являются абстрактной моделью сжатых текстов.

Определение. Прямолинейной программой называется контекстно-свободная грамматика $\mathcal{P}$, в которой нетерминальные символы $X_1, \ldots, X_m$ упорядочены (X_m — стартовый символ), и где у каждого нетерминального символа есть только одно правило: $X_i \to a$, где a — терминал, или $X_i \to X_j X_k$ для некоторых $j, k < i$.

Третья глава посвящена построению алгоритмов для следующих трех задач:

1. Даны два сжатых текста, представленные в виде прямолинейных программ. Определить, совпадают ли исходные тексты.

2. Даны два сжатых текста. Определить, входит ли первый из них во второй. При положительном ответе найти место первого вхождения и общее число вхождений.

3. Дан шаблон (явно заданный) и сжатый текст. Определить, образуют ли буквы шаблона подпоследовательность в тексте. Другими словами, можно ли вычеркнуть часть букв в тексте так, чтобы остался шаблон?

В разделе 3.1 этой работы приводится новый алгоритм, который решает задачу о поиске подстрок за $\mathcal{O}(n^2m)$ шагов. Применение этого алгоритма к задаче о равенстве приводит к оценке $\mathcal{O}(n^3)$ на время работы. Таким образом, улучшены результаты всех предыдущих работ по этим двум задачам [12, 17, 14, 19, 9].

В разделе 3.2 алгоритмическая техника, использованная для решения задачи о поиске подстроки, применяется к вычислению длин минимального периода и минимального накрытия сжатого текста.

Далее, в разделе 3.3 приводится алгоритм, который определяет вложимость шаблона в текст. Этот алгоритм также выдает количество минимальных подстрок и количество подстрок определенной длины, в которые вкладывается шаблон. Трудоемкость алгоритма равна $\mathcal{O}(mk^2 \log k)$, где k — длина шаблона, а m — размер прямолинейной программы, порождающей текст. Таким образом, время работы линейно относительно размера сжатого текста.

Четвертая глава посвящена доказательству вычислительной трудности следующих двух задач:

1. Дано два сжатых текста одинаковой длины. Определить количество несовпадающих символов (расстояние Хэмминга) между ними.

2. Дано два сжатых текста. Определить, является ли один текст подпоследовательностью второго.

В разделе 4.1 доказана следующая теорема.

Теорема 4.1.1. Задача о вычислении расстояния Хэмминга между сжатыми текстами является #P-полной.

Напомним, что #P — это класс функций, своеобразное расширение класса предикатов NP. Далее, в разделе 4.2 доказаны две теоремы о трудности поиска сжатой подпоследовательности в сжатом тексте (задача определения вложимости).

Теорема 4.2.1. Задача определения вложимости сжатого шаблона в сжатый текст является NP-трудной.

Теорема 4.2.2. Определение вложимости сводится (по Карпу) к определению невложимости. Определение невложимости сводится к определению вложимости.

Утверждение теоремы можно переформулировать следующим образом. Для каждой пары прямолинейных программ F и G можно быстро построить такую пару программ F' и G', что текст, порожденный F, вкладывается в текст, порожденный G, тогда и только тогда, когда текст, порожденный F', *не* вкладывается в текст, порожденный G'. Таким образом, задача вложимости лежит вне класса NP(при предположении NP≠coNP). Последний результат оказался большой неожиданностью, так как по своей природе постановка задачи очень напоминает представителей класса NP.

Пятая глава. В предыдущих главах нашего исследования изучаются архивы, которые легко могут быть преобразованы в прямолинейную программу, порождающую тот же текст. Базовой операцией восстановления текста из такого описания является конкатенация двух уже определенных фрагментов. Оставался открытым вопрос о том, есть ли такие системы представления текстов, которые используют более широкий класс операций. В пятой главе представлено новое понятие — *разреженную периодичность*, — которое может быть использовано для компактного описания некоторых длинных текстов.

Слово S называется *чисто периодическим*, если $S = W^k = W \dots W$. Другими словами, чистая периодичность соответствует делимости $p|n$ и равенству $s_i = s_{i+p}$ для всех $1 \le i < i + p \le n$.

Два слова на иллюстрации внизу не являются чисто периодическими. Однако, они обладают некоторой структурной закономерностью. Так, в соответствии с введенным в диссертации новым обобщенным понятием периодичности эти строки становятся периодическими.

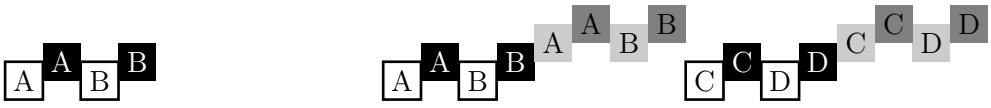
Частично определенное слово — это слово в алфавите $\Sigma \cup \{\diamond\}$, где $\diamond$ — это специальная прозрачная (или неопределенная) буква [2, 5, 6]. Другими словами, частично определенное слово представляет собой последовательность обычных слов (блоков), разделенных пропусками фиксированной (но необязательно одинаковой) длины.

Частично определенное слово S называется *разреженным периодом* (обычного) слова T, если T можно разделить на одну или несколько параллельно сдвинутых копий S, которые будут удовлетворять следующим условиям:

- Все определенные (видимые) буквы S-копий совпадают с соответствующими буквами в тексте;
- Каждая буква текста покрыта *в точности одной* определенной (видимой) буквой из S-копий.

Представим, что имеется несколько копий частично определенного слова, напечатанных на прозрачной бумаге. Тогда это слово будет разреженным периодом некоторого текста, если можно сложить данные копии в стопку так, чтобы образовалась одна связная строка без перекрытий видимых букв.

На следующей иллюстрации показаны разбиения, доказывающие разреженную периодичность исходных примеров.



Понятие разреженной периодичности вызывает следующие важные вопросы: (1) Как перечислить все разреженные периоды? (2) Каждое ли слово имеет единственный *примитивный* разреженный период? (3) Как найти все разреженные периоды минимального размера? (4) Сколько разреженных периодов может иметь слово длины n? (5) Каково отношение между примитивным классическим периодом и примитивным разреженным периодом?

В разделе 5.1 определяется частичный порядок на разреженных периодах. Показано, что, в отличие от классического случая [8], у текста может оказаться несколько примитивных разреженных периодов. Представлен пример (24 буквы), обладающий двумя независимыми примитивными разреженными периодами. Таким образом, получено опровержение гипотезы Теро Харью об *общем подразбиении* [11].

В подразделе 5.2.1 установлена связь между разреженными периодами слова в однобуквенном алфавите и факторизациями его длины.

Теорема 5.2.1. Существует биективное соответствие между разреженными периодами унарного слова длины n и упорядоченными разложениями $n = n_1 \cdot \ldots \cdot n_k$, где $n_2, \ldots, n_k \geq 2$.

В подразделе 5.2.2 доказана теорема о связи примитивных разреженных периодов некоторого текста T с его классическим примитивным периодом.

Теорема 5.2.2. Любой примитивный разреженный период Q слова T является также разреженным периодом для классического примитивного периода T.

Из этого свойства следует, что разреженная периодичность живет "внутри" классического примитивного периода. Доказательство основано на расширенном алгоритме Евклида.

Наконец, в подразделе 5.2.3 представлен алгоритм для нахождения (всех) разреженных периодов минимального размера для данного текста. Время работы алгоритма составляет $n^{1+o(1)}$ шагов.

Список литературы

[1] *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи // Пер. с англ.— Москва, Мир, 1982.

[2] *Шур А.М., Гамзова Ю.В.* Частичные слова и свойство взаимодействия периодов // Известия РАН. Серия математическая, 2004, 68:2, 191–214.

[3] *Amir A., Benson G., Farach M.* Let sleeping files lie: Pattern matching in Z-compressed files // *SODA'94*, 1994.

[4] *Berman P., Karpinski M., Larmore L., Plandowski W., and Rytter W.* On the complexity of pattern matching for highly compressed two-dimensional texts // *Journal of Computer and Systems Science*, 65(2):332–350, 2002.

[5] *Blanchet-Sadri F.* Periodicity on partial words // *Computers and Mathematics with Applications*, 47(1):71–82, 2004.

[6] *Boasson L., Berstel J.* Partial words and a theorem of Fine and Wilf // *Theoretical Computer Science*, 218(1):135–141, 1999.

[7] *Farach M., Thorup M.* String matching in Lempel-Ziv compressed strings // *STOC '95*, pages 703–712, ACM Press, 1995.

[8] *Fine N., Wilf H.* Uniqueness theorems for periodic functions // *Proc. Amer. Math. Soc.*, 16:109–114, 1965.

[9] *Gąsieniec L., Karpinski M., Plandowski W., and Rytter W.* Efficient algorithms for Lempel-Ziv encoding (extended abstract) // *SWAT'96*, LNCS 1097, pages 392–403, Springer-Verlag, 1996.

[10] *Genest B., Muscholl A.* Pattern matching and membership for hierarchical message sequence charts // *LATIN'02*, LNCS 2286, pages 326–340, Springer-Verlag, 2002.

[11] *Harju T.* Defect theorem // Lecture notes of "Combinatorics of words" Tarragona course, 2002/2003.

[12] *Hirao M., Shinohara A., Takeda M., and Arikawa S.* Fully compressed pattern matching algorithm for balanced straight-line programs // *SPIRE'00*, pages 132–138, IEEE Computer Society, 2000.

[13] *Kärkkäinen J., Navarro G., and Ukkonen E.* Approximate string matching over Ziv-Lempel compressed text //*CPM'00*, LNCS 1848, pages 195–209, Springer-Verlag, 2000.

[14] *Karpinski M., Rytter W., and Shinohara A.* Pattern-matching for strings with short descriptions // *CPM'95*, LNCS 937, pages 205–214, Springer-Verlag, 1995.

[15] *Lasota S., Rytter W.* Faster algorithm for bisimulation equivalence of normed context-free processes // *MFCS'06*, LNCS 4162, pages 646–657, Springer-Verlag, 2006.

[16] *Lohrey M.* Word problems on compressed word // *ICALP'04)*, LNCS 3142, pages 906–918, Springer-Verlag, 2004.

[17] *Miyazaki M., Shinohara A., and Takeda M.* An improved pattern matching algorithm for strings in terms of straight line programs // *CPM '97*, LNCS 1264, pages 1–11, Springer-Verlag, 1997.

[18] *Navarro G., Raffinot M.* A general practical approach to pattern matching over Ziv-Lempel compressed text // *CPM'99*, LNCS 1645, pages 14–36, Springer-Verlag, 1999.

[19] *Plandowski W.* Testing equivalence of morphisms on context-free languages // *ESA'94*, LNCS 855, pages 460–470, Springer-Verlag, 1994.

[20] *Plandowski W.* Satisfiability of word equations with constants is in PSPACE // *J. ACM*, 51(3):483–496, 2004.

[21] *Rytter W.* Application of Lempel-Ziv factorization to the approximation of grammar-based compression // *Theoretical Computer Science*, 302(1–3):211–222, 2003.

[22] *Ziv J., Lempel A.* A universal algorithm for sequential data compression // *IEEE Transactions on Information Theory*, 23(3):337–343, 1977.

ПУБЛИКАЦИИ АВТОРА ПО ТЕМЕ ДИССЕРТАЦИИ

[23] *Карьюмяки Ю., Лифшиц Ю.М.* Разреженная периодичность // Препринт ПОМИ 22/06, 2006.

[24] *Лифшиц Ю.М.* Алгоритмические свойства сжатых текстов // Препринт ПОМИ 23/06, 2006.

[25] *Лифшиц Ю.М.* Обработка сжатых текстов // Материалы XVI Международной школы-семинара "Синтез и сложность управляющих систем", стр. 64–68, Изд-во механико-математического факультета МГУ, 2006.

[26] *Cégielski P., Guessarian I., Lifshits Y., and Matiyasevich Y.* Window subsequence problems for compressed texts // *CSR'06*, LNCS 3967, pages 127–136, Springer-Verlag, 2006.

[27] *Lifshits Y., Lohrey M.* Querying and embedding compressed texts // *MFCS'06*, LNCS 4162, pages 681–692, Springer-Verlag, 2006.