Изучение периодических структур в произвольных символьных последовательностях на грид-системах из персональных компьютеров

Колпаков Р. М.[1] (foroman@mail.ru), Посыпкин М. А.[2] (mposypkin@gmail.com), Храпов Н. П.[2] (nkhrapov@gmail.com).

[1] Франко-Российский Институт Информатики и Прикладной Математики.

[2] Институт Проблем Передачи Информации РАН

Аннотация

Исследование периодических структур в произвольных символьных последовательностях является важным направлением в дискретной математике. Результаты исследований используются при создании алгоритмов сжатия данных и систем поиска по шаблону. В последние годы задача периодических структур получила широкое применение также и в биологии при анализе ДНК-последовательностей. В данной статье изложена постановка одной из задач символьных последовательностей и приведены результаты, полученные к настоящему времени на основе грид-систем из персональных компьютеров.

Описание задачи поиска периодических структур в произвольных символьных последовательностях

Пусть $w = w[1]w[2]\ldots.w[n]$ - произвольное слово, где $w[i]$ может принимать бинарные значения. Длина n слова w обозначается через $|w|$. Фрагмент $w[i]....w[j]$ слова w, где $1 \le i \le j \le n$, называется *фактором* слова w и обозначается через $w[j...j]$. Положительное число p называется *периодом* слова w если $w[i] = w[i+p]$ для любого $i = 1, \ldots, n - p$. Любое слово имеет тривиальный период, равный длине слова, но при этом оно может также иметь более короткие периоды. Из всех периодов слова w можно выбрать минимальный период, который обозначается через p(w). Отношение $|w|/p(w)$ называется *порядком* слова w и обозначается через $e(w)$. Слово называется *примитивным*, если его порядок не является целым числом большим, чем 1.

Под периодичностью в слове понимается любой фактор, порядок которого не меньше, чем 2. Периодичности играют фундаментальную роль как в словарной комбинаторике [1], так и в различных приложениях, таких как алгоритмы на строках [2, 3], молекулярная биология [4] или сжатие текстовых данных [5]. Простейшим и наиболее изученным примером периодичностей являются факторы вида uu, где u – некоторое непустое

слово. Такие периодичности называются *квадратами*. Факторы вида *uuu*, где u – некоторое непустое слово, называются *кубами*. Кубы и квадраты представляют собой частные случаи факторов вида $u^k = uu \ldots . u$ (*k* подряд идущий непустых одинаковых слов) , где $k > 1$ и *u* — непустое слово. Такой фактор называется *k*-ой *степенью* слова *u*. Если слово *u* является примитивным, то называется *примитивной k*-ой степенью. Если слово *w* имеет целый порядок $k > 1$, то оно, очевидно, является примитивной *k*-ой степенью своего префикса длины *p(w)*. С другой стороны, нетрудно показать (см., например, [6]), что при $k > 1$ порядок примитивной *k*-ой степени равен *k*. Таким образом, слово имеет целый порядок $k > 1$ тогда и только тогда, когда оно является примитивной *k*-ой степенью. Наряду с целыми степенями слов можно также рассмотреть "дробные" периодичности вида $u^k u^j$, где $k \geq 2$, *u* – примитивное непустое слово и u^j – некоторый префикс слова *u*, отличный от *u*. Такие периодичности имеют дробный порядок.

Определение. Периодичность *w*[*i*..*j*] с минимальным периодом *p* в слове *w* называется максимальной, если она удовлетворяет следующим условиям:

если $i > 1$, то $w[i - 1] \neq w[i - 1 + p]$,

если $j < n$, то $w[j + 1 - p] \neq w[j + 1]$.

Таким образом, периодичность в слове называется *максимальной*, если она не содержится внутри некоторой более длинной периодичности с тем же минимальным периодом, т.е. периодичность в слове является максимальной, если она не может быть расширена в этом слове ни на один символ ни вправо, ни влево с сохранением своего минимального периода.

На рисунке 1 приведены примеры бинарных слов со всеми возможными периодичностями.

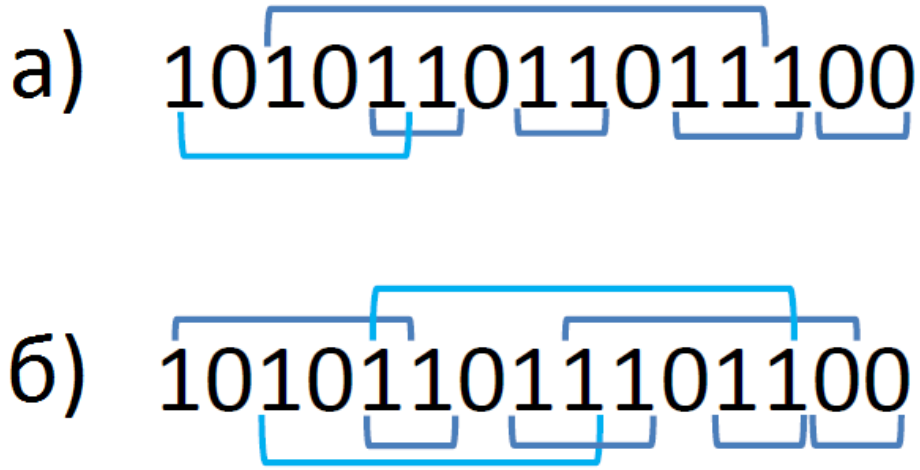

Рисунок 1. Поиск максимальных периодичностей в словах. В слове а 6 периодичностей; в слове б – 8.

Любая периодичность содержится в некоторой единственной максимальной периодичности с тем же минимальным периодом, поэтому максимальные периодичности естественным образом задают в слове все остальные периодичности (квадраты, кубы, *k*-е степени и т.д.), что позволяет их использовать в многочисленных приложениях [7]. В [8] доказано, что максимальное возможное число максимальных периодичностей в словах длины n равно $O(n)$. Более того, было показано, что максимальная возможная сумма порядков максимальных периодичностей в словах длины n равна $O(n)$.

Понятие периодичности нетрудно обобщить на случай факторов, порядок которых не меньше, чем $1 + \varepsilon$, где $0 < \varepsilon < 1$.

Определение. ε-квазипериодичностью называется фактор: порядок которого не меньше, чем 1 + ε.

Рассматриваемая нами задача состоит в оценке максимального возможного числа $mrn(n, \varepsilon)$ максимальных ε-квазипериодичностей в словах длины n для различных значений n и ε. Можно показать, что $mrn(n, \varepsilon) = O(n/\varepsilon^2)$. Однако остаётся открытым вопрос, является ли данная оценка для $mrn(n, \varepsilon)$ точной по порядку.

Методология проведения вычислительных экспериментов

Для получения достаточно точной оценки максимума исследуемой величины требуется большой объем вычислений, выполнение которых заняло бы на одном персональном компьютере неприемлемо длительное время. В таких случаях целесообразно применение методов параллельных и распределенных вычислений [9-11].

Грид-системы из персональных компьютеров (далее ГСПК) способны привлекать ресурсы добровольцев для организации масштабных вычислений. Это наиболее доступный для научного коллектива тип вычислительных ресурсов, обладающий наилучшим соотношением цена/мощность. ГСПК способна решать вычислительные задачи, состоящие из множества независимых подзаданий. В силу изложенных причин ГСПК является наиболее подходящим для проведения расчётов ресурсом.

Для поиска значений $mrn(n, \varepsilon)$ была разработана программа smallexp, анализирующая символьные последовательности. Последовательности создавались в коде программы специально разработанным генератором случайных последовательностей. Данное приложение было реализовано на языке Си с использованием стандартных библиотек, вследствие чего имело версии для различных современных операционных систем. Наиболее распространенной технологией распределенных вычислений в настоящее время является система BOINC (Berkeley Open Infrastructure for Network Computing). В рамках данной инфраструктуры можно запускать приложения, разработанные с применением специальных BOINC-библиотек. Для запуска стандартного неадаптированного приложения в рамках инфраструктуры BOINC можно использовать вспомогательную программу-wrapper. Данная программа по отношению к инфраструктуре является BOINC-приложением, которое перенаправляет нужным образом потоки ввода-вывода и запускает целевое приложение.

Расчеты проводились на инфраструктуре BOINC в рамках проекта OPTIMA@HOME [9]. Особенность работы приложения smallexp на инфраструктуре BOINC состоит в том, что все вычислительные узлы получали задания с одинаковыми входными данными; в процессе работы на различных узлах генерировались различные случайные символьные последовательности для анализа.

Отладка приложения проводилась на тестовой инфраструктуре, состоящей из физических и виртуальных машин. Изначально был создан BOINC-проект в рамках тестовой инфраструктуры boinc-test.isa.ru, включающей в себя несколько десятков физических и виртуальных вычислительных машин под управлением различных версий операционных систем Windows и Linux. После этапа отладки, заключавшегося с запуске приложения на всех узлах тестовой инраструктуры, приложение было перенесено в проект OPTIMA.

Проведение и результаты вычислительного эксперимента

В изначальной версии приложение smallexp, именуемое *решателем*, выполняло перебор последовательностей для конкретно заданного параметра ε. В процессе дальнейшей работы решатель был оптимизирован для подсчёта *mrn(n, ε)* для нескольких параметров *ε* одновременно. Таким образом эффективность решателя удалось повысить примерно в 5 раз.

В общей сложности было обработано 2.5×10^{13} случайных двоичных слов. Результаты вычислительного эксперимента приведены в таблице 1. Графическое отображение результатов приведено на рисунке 2.

n	*k*	*ε*	*mrn(1; 1 + 1/k)*
120	1	1	81
120	2	1/2	142
120	3	1/3	197
120	4	1/4	252
120	5	1/5	303

Таблица 1. Результаты вычислительного эксперимента по подсчёту максимального числа *ε*-квазипериодичностей в произвольных словах.

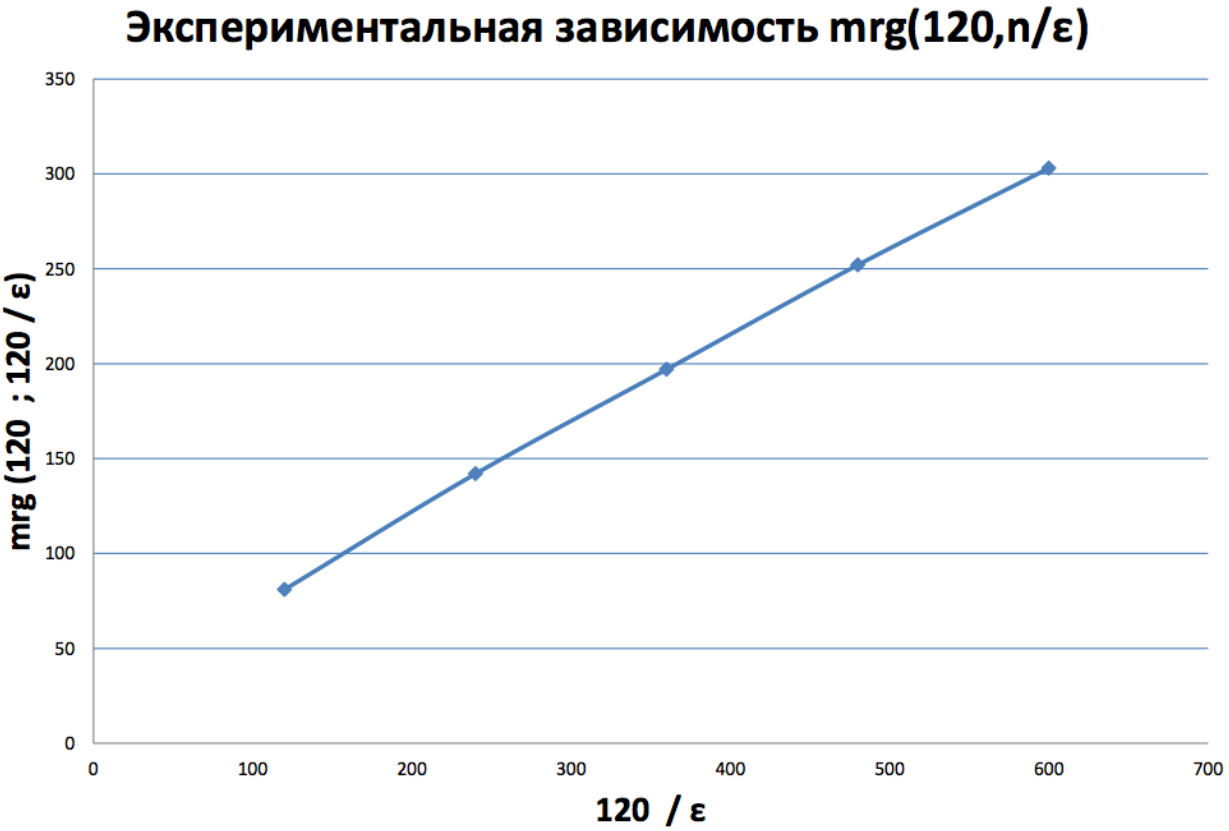

Рисунок 2. Графическое отображение экспериментальных результатов вычислительного эксперимента.

В результате предварительных расчётов были получены точные нижние грани *mrn(n,(k + 1)/k)* для случаев $n = 120$ и $k = 1, 2, 3, 4, 5$. Полученные результаты позволяют сделать предположение, что *mrn(n, ε)= O(n / ε)*. Однако для ε = 1 была составлена теоретическая оценка значения mrn [12], равная 110. По причине большого расхождения теоретической оценки и экспериментального значения, полной уверенности в верности данной гипотезы на данный момент быть не может. Полученный к настоящему моменту результат является важным приближением к решению поставленной задачи.

Перспективы дальнейших исследований

Проведенные к настоящему моменту исследования показали, что для улучшения полученного результата необходимо затратить много больший объем вычислительных ресурсов или повысить эффективность использования уже имеющихся ресурсов.

Эффективность использования уже имеющихся ресурсов в свою очередь можно повысить посредством оптимизации решателя. Возможно два основных способа оптимизации решателя:

- Оптимизация на уровне процессорных регистров. Данный способ предполагает использование побитовых операций языка Си или использования языка ассемблера. При данном способе исследуемое слово или его фрагмент помещаются в регистры компьютера и исследуются посредством операции побитового сдвига.
- Оптимизация генератора случайных слов. Нынешний подход предполагает анализ случайных последовательностей. Новый подход предполагает составление приложения таким образом, чтобы оно анализировало не абсолютно случайные слова, а слова, прошедшие некоторую проверку. Данная оптимизация позволит высвободить процессорное для анализа более подходящих слов.

Совместная реализация двух данных методов позволит существенно повысить эффективность использования имеющихся ресурсов. Кроме того, дальнейшее расширение вычислительной инфраструктуры позволит задействовать для вычислений новый ресурс и также существенно улучшить результат.

Список литературы

1. M. Lothaire, Combinatorics on Words, volume 17 of Encyclopedia of Mathematics and Its Applications, Addison Wesley, 1983.
2. Z. Galil, J. Seiferas, Time-space optimal string matching, J. Comput. System Sci. 26(3) (1983), 280–294.
3. M. Crochemore, W. Rytter, Squares, cubes, and time-space efficient string searching, Algorithmica 13 (1995), 405–425.
4. D. Gusfield, Algorithms on Strings, Trees, and Sequences, Cambridge University Press, 1997.
5. J. Storer, Data compression: methods and theory, Computer Science Press, Rockville, MD, 1988.
6. R. Kolpakov, G. Kucherov, Periodic structures in words, chapter for the 3rd Lothaire volume Applied Combinatorics on Words, Cambridge University Press, 2005.

7. M. Crochemore, C. Iliopoulos, M. Kubica, J. Radoszewski, W. Rytter, T. Walen, Extracting powers and periods in a string from its runs structure, Lecture Notes in Comput. Sci. 6393 (2010), 258–269.
8. R. Kolpakov, G. Kucherov, On Maximal Repetitions in Words, J. Discrete Algorithms 1(1) (2000), 159–186.
9. О.С. Заикин, М.А. Посыпкин, А.А. Семѐнов, Н.П. Храпов. Опыт организации добровольных вычислений на примере проектов OPTIMA@HOME и SAT@HOME.Вестник Нижегородского университета им. Н.И. Лобачевского, 2012, No5(2), с.340-347.
10. Р. Ловаш, А.П. Афанасьев, В.В. Волошинов, М.А. Посыпкин, О.В. Сухорослов, Н.П. Храпов. О грид-системах из персональных компьютеров и их интеграции с другими видами грид-систем. Сборник избранных трудов V Международной научно-практической конференции "Современные информационные технологии и ИТ-образование". Москва: ИНТУИТ.РУ 2010 г.
11. О.В. Сухорослов. Реализация и композиция проблемно-ориентированных сервисов в среде MathCloud // Вестник ЮУрГУ. Серия «Математическое моделирование и программирование», Вып. 8, № 17(234). – Челябинск: Издательский центр ЮУрГУ, 2011. (с. 101-112)
12. http://www.csd.uwo.ca/~ilie/runs.html