На правах рукописи

Тимошевская Наталия Евгеньевна

РАЗРАБОТКА И ИССЛЕДОВАНИЕ ПАРАЛЛЕЛЬНЫХ КОМБИНАТОРНЫХ АЛГОРИТМОВ

05.13.01 – Системный анализ, управление и обработка информации
(в отраслях информатики, вычислительной техники и автоматизации)

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Томск – 2007

Работа выполнена на кафедре защиты информации и криптографии ГОУ ВПО «Томский государственный университет»

Научный руководитель: доктор технических наук, профессор Агибалов Геннадий Петрович

Официальные оппоненты: доктор технических наук, профессор Костюк Юрий Леонидович

кандидат технических наук, Семенов Александр Анатольевич

Ведущая организация: Институт вычислительной математики и математической геофизики СО РАН, Новосибирск

Защита состоится 24 мая 2007 г. в 12.00 на заседании Диссертационного совета Д 212.267.12 при ГОУ ВПО «Томский государственный университет» по адресу: 634050, г. Томск, пр. Ленина, 36.

С диссертацией можно ознакомиться в Научной библиотеке Томского государственного университета по адресу: г. Томск, пр. Ленина, 34а.

Автореферат разослан 23 апреля 2007 г.

Ученый секретарь диссертационного совета, д.т.н., профессор Смагин В. И.

научно-практического семинара / Под ред. проф. Р.Г. Стронгина. Нижний Новгород: Изд-во Нижегородского университета, 2002. – С. 16 – 20.

2. Тимошевская Н.Е. О распараллеливании обхода дерева поиска // Вестник Томского госуниверситета. Приложение. – 2002. – № 1(II). – С. 241 – 245.

3. Тимошевская Н.Е. О параллельных алгоритмах обхода дерева // Вестник Томского госуниверситета. Приложение. – 2003. – № 6. – С. 202 – 209.

4. Тимошевская Н.Е. Разработка параллельных алгоритмов обхода дерева // Высокопроизводительные параллельные вычисления на кластерных системах // Материалы третьего Международного научно-практического семинара / Под ред. проф. Р.Г. Стронгина. – Нижний Новгород: Изд-во Нижегородского университета, 2003. – С. 198 – 206.

5. Тимошевская Н.Е. Параллельные вычисления в решении систем логических уравнений методом линеаризации // Материалы XV международной школы-семинара "Синтез и сложность управляющих систем" (Новосибирск, 18-23 октября 2004 г.). – Новосибирск: Институт математики, 2004. – С. 97 – 102.

6. Тимошевская Н.Е. Параллельные методы обхода дерева // Математическое моделирование. – 2004. – Том 16. – №1. – С. 105 – 114.

7. Тимошевская Н.Е. Параллельная генерация сочетаний и перестановок. // Вторая Сибирская школа-семинар по параллельным вычислениям / Под ред. А. В. Старченко. – Томск: Изд-во Том. ун-та, 2004. – С. 55 – 59.

8. Тимошевская Н.Е. Экспериментальное исследование стойкости сжимающего генератора // Вестник Томского госуниверситета. Приложение. – 2004. – № 9(I). – С. 84 – 88.

9. Гисс С.С., Тимошевская Н. Е. Распараллеливание алгоритма решения задачи о выполнимости КНФ // Вторая Сибирская школа-семинар по параллельным вычислениям / Под ред. А. В. Старченко. – Томск: Изд-во Том. ун-та, 2004. – С. 65 – 66.

10. Тимошевская Н.Е. О нумерации перестановок и сочетаний для организации параллельных вычислений в задачах проектирования управляющих систем // Известия Томского политехнического университета. – 2004. – Том 307. – №6. – С. 18 – 20.

11. Тимошевская Н.Е. О линеаризационных множествах // Проблемы теоретической кибернетики. Тезисы докладов XIV Международной конференции (Пенза, 23–28 мая 2005 г.) / Под редакцией О.Б. Лупанова. – М.: Изд-во механико-математического факультета МГУ, 2005. – С. 154.

12. Тимошевская Н.Е. Задача о кратчайшем линеаризационном множестве // Вестник Томского госуниверситета. Приложение. – 2005. – № 14 – С. 79 – 83.

13. Тимошевская Н.Е. О линеаризационно эквивалентных покрытиях // Вестник Томского госуниверситета. Приложение. – 2005. – № 14 – С. 84 – 91.

14. Тимошевская Н.Е. Параллельное перечисление разбиений множества методом нумерации // Вестник Томского госуниверситета. Приложение. – 2006. – № 17 – С. 260 – 264.

15. Тимошевская Н.Е. О методах разработки параллельных комбинаторных алгоритмов // Третья Сибирская школа-семинар по параллельным вычислениям / Под ред. А. В. Старченко. – Томск: Изд-во Том. ун-та, 2006. – С. 60 – 72.

дается пошаговое описание действий управляющего и рабочего процессов. Эффективность распараллеливания получилась в среднем равной 0,9.

Наконец оценивается количество покрытий, имеющих линеаризационные множества заданной мощности. Обозначим $L(n, k)$ число всех покрытий множества $X = \{1, 2, \ldots, n\}$ с линеаризационными множествами мощности k.

Теорема 3.8. $L(n, 1) = 1 + 2n(3^{n-1} - n)$.

Пусть D_{ks} есть число покрытий k-элементного множества, состоящих из s блоков. Тогда (**следствие из теоремы 3.9**)

$$\sum_{s=1}^{2^k-1} D_{ks} \cdot (2^{n-k+1} - 1)^s \le L(n, k) \le \binom{n}{k} 2^{n-k} \sum_{s=1}^{2^k-1} D_{ks} \cdot (2^{n-k+1} - 1)^s .$$

Покрытия, не содержащие одноэлементных блоков, будем называть 2-*покрытиями*. Обозначим $U(n)$ число 2-покрытий множества X и $U(n, k)$ число тех из них, для которых существуют линеаризационные множества мощности k. Для $U(n)$ получена следующая формула

$$U(n) = \sum_{r=0}^{n} (-1)^r \binom{n}{r} 2^{\delta(n-r)} \text{, где } \delta(n) = 2^n - 1 - n.$$

Обозначим P_{kst} множество всех s-блочных покрытий некоторого k-элементного множества $L \subseteq X$, содержащих ровно t одноэлементных блоков. Пусть $D_{kst} = |P_{kst}|$. Рассмотрим произвольное $P \in P_{kst}$. Обозначим M_{nkst} число всех таких 2-покрытий F множества X, что каждый блок в P входит в какой-нибудь блок покрытия F и каждый блок в F содержит не более одного элемента из $\overline{L} = X \setminus L$, т.е. L является линеаризационным для F. Подсчитаем число $U_L(n, k)$ 2-покрытий множества X, для которых множество L является линеаризационным.

Теорема 3.10. Пусть $L \subseteq X$, $|X| = n$, $|L| = k > 0$. Тогда

$$U_L(n, k) = \sum_{s=1}^{2^k-1} \sum_{t=0}^{\min\{k,s\}} D_{kst} M_{nkst} \quad \text{и} \quad D_{kst} = \binom{k}{t} \sum_{j=0}^{k-t} (-1)^j \binom{k-t}{j} \binom{\delta(k-j)}{s-t},$$

$$M_{nkst} = \sum_{r=0}^{n-k} (-1)^r \binom{n-k}{r} (2^{n-k-r} - 1)^t (2^{n-k+1-r} - 1)^{s-t} .$$

Из теорем 3.8 – 3.10 следует, что доля покрытий с нетривиальными линеаризационными множествами ничтожна мала, но их количество велико и растет с ростом мощностей покрываемого и линеаризационного множеств, оправдывая постановку задачи поиска линеаризационных множеств для них.

Основные публикации автора по теме диссертации

1. Беляев В.А., Тимошевская Н.Е. Распараллеливание обхода дерева поиска для решения задачи о рюкзаке на кластерной системе // Высокопроизводительные параллельные вычисления на кластерных системах. Материалы Международного

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность проблемы. Потребность в решении комбинаторных задач и, следовательно, в разработке комбинаторных алгоритмов возникает во многих областях, в том числе таких, как автоматизированное проектирование, создание систем искусственного интеллекта, разработка сетей связи и криптография. Под комбинаторными задачами в диссертации подразумеваются *перечислительные* и *поисковые* задачи на конечных комбинаторных множествах, элементами которых служат комбинаторные объекты, представляющие собой комбинации элементов других конечных (возможно, тоже комбинаторных) множеств – сочетания, перестановки, разбиения, покрытия, линеаризационные множества переменных, решения систем логических уравнений и т.п. Перечислительная задача заключается в перечислении всех элементов некоторого подмножества конечного комбинаторного множества, а поисковая – в нахождении в последнем элемента с заданными свойствами. За редким исключением перебор (порождение, перечисление) элементов основного комбинаторного множества и их опробование (тестирование на принадлежность требуемому подмножеству или обладание нужным свойством) является единственным способом решения комбинаторной задачи. В этом случае элементы порождаются обычно в некотором порядке. Во многих случаях это делается в порядке обхода дерева поиска, представляющего собой модель комбинаторного множества.

Почти все комбинаторные алгоритмы имеют экспоненциальную сложность. Вместе с тем на практике необходимо уметь решать комбинаторные задачи достаточно больших размерностей. Одна из возможностей уменьшения времени решения комбинаторных задач состоит в том, чтобы разбить комбинаторное множество на классы и исследовать эти классы параллельно, т.е. одновременно на нескольких процессорах вычислительной системы. В настоящее время ведущую позицию по степени распространенности занимают кластерные системы, относящиеся к классу многопроцессорных вычислительных систем (МВС) с распределенной памятью. Проблема, возникающая при разработке комбинаторных алгоритмов для последующей их реализации на кластерных системах, заключается в том, **как** надо разбивать комбинаторное множество на классы и раздавать последние процессорам так, чтобы обеспечить наивысшую степень ускорения вычислений. Последнее зависит от многих факторов, в том числе от неравномерной загрузки процессоров системы, от накладных расходов на коммуникации, от избыточности просматриваемой области поиска и др. Несмотря на многие сотни работ, выполненных по этой проблеме только за последние три десятка лет, эффективного решения ее на данный момент не найдено ни для МВС с общей памятью, ни для МВС с раздельной памятью, в том числе кластеров. Это справедливо в отношении не только комбинаторной задачи в абстрактной постановке (с произвольным деревом поиска, например), но и для многих конкретных комбинаторных задач, включая названные выше перечислительные и поисковые задачи. Таким образом, проблема создания эффективных параллельных комбинаторных алгоритмов, которой посвящена

реферируемая работа, лежит в русле современного развития вычислительной дискретной математики и является актуальной.

Цель работы и задачи исследования. Целью диссертационной работы является разработка и исследование параллельных алгоритмов решения перечислительных и поисковых комбинаторных задач на вычислительных системах кластерного типа.

Для достижения указанной цели предполагается решение следующих основных задач.

1. Построение методов параллельного обхода дерева поиска в глубину, возникающего в перечислительных и поисковых комбинаторных задачах.
2. Построение метода параллельного перечисления комбинаторных объектов на основе предварительного разбиения комбинаторного множества.
3. Разработка параллельных алгоритмов перечисления сочетаний, перестановок, разбиений, решений задач о рюкзаке и о назначении методами обхода дерева и разбиения комбинаторного множества.
4. Исследование свойств линеаризационных множеств для решения систем логических уравнений.
5. Разработка параллельного алгоритма решения систем логических уравнений с заданным линеаризационным множеством на основе обхода дерева поиска.
6. Экспериментальное исследование разработанных алгоритмов решения комбинаторных задач с помощью компьютерного моделирования.

Методы исследования. Теоретические исследования используют аппарат комбинаторики, теории графов, теории вычислительной сложности, параллельных вычислений, а экспериментальные – программные средства компьютерного моделирования алгоритмов на вычислительных системах кластерного типа.

Научная новизна. Новыми являются все основные результаты, полученные в диссертации, в том числе:

– методы параллельного обхода дерева поиска в глубину – метод назначаемых поддеревьев и метод выделяемых поддеревьев, обеспечивающие высокую эффективность использования ресурсов кластерной системы;

– метод параллельного перечисления комбинаторных объектов на основе предварительного разбиения комбинаторного множества – метод нумерации, обеспечивающий равномерную загрузку процессоров кластера;

– параллельные алгоритмы перечисления сочетаний, разбиений множества, решений задач о рюкзаке и о назначении методами параллельного обхода дерева поиска;

– параллельные алгоритмы решения задач перечисления сочетаний, перестановок, разбиений множества методом нумерации;

– параллельный алгоритм нахождения кратчайшего линеаризационного множества покрытия методом назначаемых поддеревьев;

Утверждение 3.18. Если в графе A1(B) есть клика мощности k, то любое линеаризационное множество покрытия B имеет мощность не меньше $k - 1$.

Утверждение 3.19. Если A1(B) является полным двудольным графом $K_{t,m}$, то всякое кратчайшее линеаризационное множество для B имеет мощность $\min(t, m)$.

Утверждение 3.20. Если граф $A1(B)$ состоит из r компонент связности $G_1, \ldots, G_r$ и для каждого $i = 1, \ldots, r$ в G_i есть клика мощности k_i, то для покрытия B не существует линеаризационного множества мощности меньше $\sum_{i=1}^{r} k_i - r$.

Для решения задачи поиска кратчайшего линеаризационного множества покрытия предложены последовательные приближенный (жадный) алгоритм и **алгоритм А6** обхода дерева, сокращаемого за счет использования функции нижней оценки, который применим для нахождения как приближенного, так и точного решения.

Реализована программа, которая для заданной системы нелинейных уравнений ищет линеаризационное множество соответствующего покрытия по алгоритму А6. В программе предусмотрены следующие режимы работы: 1) выполнять переход к эквивалентному покрытию, построенному алгоритмом А4; 2) искать покрытие в течение заданного интервала времени.

Из результатов работы программы для ряда генерируемых случайным образом систем уравнений следует: 1) вывод о полезности предварительного преобразования соответствующих покрытий алгоритмом А4: оно позволяет в среднем сократить число блоков в покрытии в 15 раз, а время поиска в некоторых случаях – в сотни раз; 2) высокое качество жадного алгоритма: мощность кратчайшего линеаризационного множества отличается от мощности множества, найденного жадным алгоритмом, не более чем на 2; 3) для случайных покрытий множества X мощность линеаризационного множества, доставляемого алгоритмом А6 в течение 600 сек., в среднем в 1,5 раза меньше мощности X.

Распараллеливание алгоритма А6 построения кратчайшего линеаризационного множества выполняется по МНП. На первом этапе управляющий процесс с помощью жадного алгоритма находит некоторое линеаризационное множество, затем выполняет обход дерева в ширину, начиная от его корня, до тех пор, пока не построит заданное число m корневых вершин. При этом для каждой вершины вычисляется функция нижней оценки и, если она показывает перспективность ветвления из данной вершины, то вершина помещается в очередь. Таким образом, сокращение обхода выполняется уже на первом этапе. На втором этапе управляющий процесс, как и предписывает метод, назначает корневые вершины рабочим процессам по мере обхода ими ранее назначенных поддеревьев. Кроме корневой вершины, управляющий процесс передает рабочему минимальную известную ему мощность линеаризационного множества. В свою очередь, если рабочий процесс находит в своем поддереве линеаризационное множество меньшей мощности, то пересылает его мощность управляющему. В работе

Утверждение 3.9. Любое минимальное покрытие $B \in M(X)$ является безызбыточным для любого X.

Утверждение 3.10. Для любого множества X, $|X| \geq 5$, существует кратчайшее, но не минимальное покрытие $B \in M(X)$.

Утверждение 3.11. Для любого множества X, $|X| \geq 8$, существует минимальное, но не кратчайшее покрытие $B \in M(X)$.

Пусть $G(V, E)$ есть неориентированный граф без петель и кратных ребер. Покрытие B множества $X = V$ такое, что множество L является линеаризационным для этого покрытия тогда и только тогда, когда оно является вершинным покрытием графа $G(V, E)$, будем называть *соответствующим* графу G. В работе дается полиномиальный **алгоритм А2** построения соответствующего графу G покрытия, обозначаемого А2(G). Некоторые его свойства отражены в утверждениях (3.15 – 3.17).

Утверждение 3.15. Пусть $B = \{B_0, ..., B_{k-1}\} = \text{A2}(G(V, E))$. Тогда $k \leq |E|$, причем $k = |E|$ тогда и только тогда, когда $|B_i| = 2$ для каждого $i = 0, ..., k-1$; в этом случае $B = E$.

Утверждение 3.16. Пусть $B = \text{A2}(G(V, E))$. Тогда $|B| < |E|$, если и только если в G есть полный подграф с тремя (или более) вершинами.

Утверждение 3.17. Пусть $B = \{B_0, ..., B_{k-1}\} = \text{A2}(G(V, E))$. Тогда $\sigma(B) \leq |E| + k$, причем $\sigma(B) = 2|E|$ тогда и только тогда, когда $B = E$.

Теорема 3.5. Задача построения кратчайшего покрытия, эквивалентного данному покрытию, является *NP*-трудной.

Экспоненциальная сложность алгоритмов минимизации покрытия вынуждает отказаться от поиска точного решения – кратчайшего или минимального покрытия и ограничиться существующим или с меньшими длиной и весом. Последнего в некоторых случаях можно достичь с помощью следующего полиномиального **алгоритма А4**. Последовательно применяются алгоритмы А1, А2 и А3, т.е. вычисляются $G(V, E) = \text{A1}(B)$, $B' = \text{A2}(G)$, $B'' = \text{A3}(B')$ и B'' принимается за результат минимизации B, если $|B''| < |B|$ и $\sigma(B'') < \sigma(B)$.

Теорема 3.7. Задача построения кратчайшего линеаризационного множества покрытия является *NP*-трудной.

Следующие утверждения (3.18 – 3.20) позволяют, во-первых, оценить снизу мощность линеаризационных множеств для покрытий некоторых типов, и, во-вторых, разрабатывать алгоритмы построения покрытий с линеаризационными множествами ограниченной снизу мощности.

– параллельный алгоритм решения системы логических уравнений с заданным линеаризационным множеством соответствующего покрытия методом выделяемых поддеревьев;

– экспериментальные оценки эффективности данных алгоритмов.

Новыми являются также следующие вспомогательные результаты, представляющие, кроме того, самостоятельный интерес для вычислительной дискретной математики, в частности для теории логических уравнений и ее приложений:

– введено понятие эквивалентности покрытий множества, где эквивалентными считаются покрытия с одинаковыми линеаризационными множествами; предложен эффективный способ проверки эквивалентности двух покрытий с помощью их графического представления и показана связь между линеаризационным множеством покрытия и вершинным покрытием соответствующего ему графа;

– в классе эквивалентности покрытий определены понятия безызбыточного, кратчайшего и минимального покрытий; установлены необходимое и достаточное условие безызбыточности покрытия и необходимое условие для кратчайшего покрытия; описана структура класса эквивалентности, представляющая собой соотношения между подмножествами безызбыточных, кратчайших и минимальных покрытий в нем;

– доказаны *NP*-трудность задачи нахождения кратчайшего линеаризационного множества и задачи построения кратчайшего покрытия, эквивалентного заданному;

– предложена композиция алгоритмов, позволяющих построить безызбыточное покрытие меньшей сложности, эквивалентное заданному;

– описан ряд конструкций и алгоритмы построения покрытий множеств с линеаризационными множествами ограниченной снизу мощности;

– получена оценка доли покрытий, для которых существует линеаризационное множество заданной мощности.

Теоретическая и практическая значимость работы. Построенные методы параллельного обхода дерева и нумерации могут применяться для разработки высокоэффективных параллельных алгоритмов решения перечислительных и поисковых комбинаторных задач, что продемонстрировано в работе на ряде конкретных задач. Разработанные параллельные алгоритмы поиска кратчайшего линеаризационного множества покрытия могут использоваться для решения систем нелинейных логических уравнений, а параллельные алгоритмы решения последних и реализующие их параллельные программы могут применяться для исследования криптографической стойкости генераторов ключевого потока. Выявленные свойства линеаризационных множеств покрытий и линеаризационно эквивалентных покрытий полезны для анализа и синтеза дискретных функций с хорошими криптографическими свойствами.

Достоверность результатов. Достоверность полученных в работе теоретических результатов и формулируемых на их основе выводов

обеспечивается строгостью производимых математических выкладок. Справедливость выводов относительно эффективности разработанных алгоритмов подтверждается результатами компьютерного эксперимента.

Основные положения, выносимые на защиту.

1. Методы распараллеливания комбинаторных алгоритмов, а именно: методы назначаемых и выделяемых поддеревьев для параллельного обхода дерева поиска в глубину и метод нумерации для параллельного перечисления комбинаторных объектов.

2. Параллельные алгоритмы решения комбинаторных задач, разработанные на основе методов распараллеливания, включая задачи перечисления (сочетаний, перестановок, разбиений, решений задачи о рюкзаке) и задачи поиска (оптимального назначения, кратчайшего линеаризационного множества покрытия и решения нелинейной системы логических уравнений методом линеаризационного множества), с экспериментальными оценками их эффективности.

3. Элементы теории линеаризационных множеств покрытий в решении систем нелинейных логических уравнений, в том числе доказательства *NP*-трудности задач нахождения кратчайшего линеаризационного множества и задачи построения кратчайшего покрытия, эквивалентного заданному, понятие линеаризационной эквивалентности покрытий множества переменных и соотношения между подмножествами безызбыточных, кратчайших и минимальных покрытий в ее классе, необходимое и достаточное условие безызбыточности покрытия и необходимое условие для кратчайшего покрытия, оценка доли покрытий с линеаризационными множествами заданной мощности, алгоритмы построения безызбыточных покрытий меньшей сложности и покрытий с линеаризационными множествами ограниченной снизу мощности, и др.

Внедрение результатов исследований. Результаты диссертации внедрены:

1) в разработку тем курсовых и дипломных работ в Томском государственном университете в 2004-2006 г.г. (разработка параллельных алгоритмов генерации простых чисел, факторизации числа, решения задач коммивояжера и о выполнимости КНФ);

2) в курсы лекций «Криптографические методы защиты информации», «Методы криптоанализа», «Высокопроизводительные вычислительные системы» и «Комбинаторные алгоритмы» для студентов-математиков ТГУ по специальности 090102 «Компьютерная безопасность» и используются в лабораторных работах по этим курсам;

3) в создание программного обеспечения, предназначенного для исследования стойкости ряда криптосистем, в рамках проекта № 38025 “Параллельные алгоритмы в криптоанализе шифров”, выполненного при поддержке программы «Развитие потенциала высшей школы» в 2005г.;

4) в разработку методов распараллеливания комбинаторных алгоритмов в рамках проектов «Разработка и исследование алгоритмов построения

Построенный так граф G назовем *графическим представлением* покрытия B и будем обозначать A1(B). Алгоритм A1 имеет полиномиальную сложность.

Утверждение 3.1. Множество L является линеаризационным для $B \in M(X)$ тогда и только тогда, когда оно является вершинным покрытием графа A1(B).

Покрытия B и C из $M(X)$ будем называть *эквивалентными*, если любое подмножество $L \subseteq X$ является линеаризационным для B тогда и только тогда, когда оно линеаризационное для C. Класс эквивалентности на $M(X)$, содержащий множество $B \in M(X)$, обозначим $[B]$. Следующая теорема дает конструктивный тест эквивалентности двух покрытий.

Теорема 3.1. Пусть B, $B' \in M(X)$. Покрытия B и B' эквивалентны тогда и только тогда, когда графы A1(B) и A1(B') совпадают.

Число блоков в покрытии будем называть его *длиной*. Число $\sigma(B) = |B_0| + ... + |B_{k-1}|$ назовем *весом* покрытия $B = \{B_0, ..., B_{k-1}\}$. При решении задач поиска некоторого линеаризационного множества заданного покрытия во многих случаях имеет смысл выполнить переход к эквивалентному покрытию, например, меньшей длины или меньшего веса. Покрытие $B \in M(X)$ называется *кратчайшим* или *минимальным*, если соответственно $|B| \le |A|$ или $\sigma(B) \le \sigma(A)$ для любого $A \in [B]$. Покрытие $B = \{B_0, ..., B_{k-1}\} \in M(X)$ называется *безызбыточным*, если не существуют $B_i \in B$ и $u \in B_i$, что $B' = \{B_0, ..., B_{i-1}, B_i - \{u\}, B_{i+1}, ..., B_{k-1}\} \in M(X)$ и $B' \in [B]$.

Теорема 3.2. Если покрытие $B = \{B_0, ..., B_{k-1}\}$ является кратчайшим, то не существует таких множеств $B_{i_1}, B_{i_2}, ..., B_{i_r} \in B$, что $r > 1$, $i_1 < i_2 < \ldots < i_r$ и множество $Y = \bigcup_{j=1}^{r} B_{i_j}$ образует полный подграф в A1(B).

Теорема 3.3. Покрытие $B = \{B_0, ..., B_{k-1}\} \in M(X)$ является безызбыточным тогда и только тогда, когда для любых $B_i \in B$ и $u \in B_i$ имеет место $\exists v \in B_i (u \ne v \;\&\; \forall B_j \in B\ (i \ne j \Rightarrow \neg(\{u, v\} \subseteq B_j)))$.

Доказательство достаточности в теореме 3.3 содержит **алгоритм А3** преобразования заданного покрытия в эквивалентное безызбыточное.

Для $G(V, E)$ = A1(B) показывается (**утверждения 3.3 – 3.5**), что 1) $E \in [B]$ и E безызбыточное, 2) если в $G(V, E)$ нет клики, содержащей более двух вершин, то $|[B]| = 1$.

Утверждение 3.6. Для любого множества X, $|X| \ge 3$, существует безызбыточное, но не кратчайшее и не минимальное покрытие $B \in M(X)$.

Утверждение 3.8. Для любого множества X, $|X| \ge 5$, существует кратчайшее, но не безызбыточное покрытие $B \in M(X)$.

линеаризационным для S, если при любой фиксации значений переменных в L система S становится линейной. Суть метода состоит в построении некоторого линеаризационного множества и в нахождении такого набора значений переменных в этом множестве, при подстановке которого в систему последняя становится совместной. Поиск такого набора значений переменных линеаризационного множества, который делает систему совместной, выполняется алгоритмом, реализующим сокращенный обход бинарного дерева поиска с возвращением. Глубина такого дерева не превосходит мощности линеаризационного множества, и чем она меньше, тем быстрее ищется решение. В работе дается параллельный алгоритм обхода дерева для решения системы по методу выделяемых поддеревьев.

Для проведения экспериментального анализа алгоритма использовались как случайным образом сгенерированные системы логических уравнений, так и системы, описывающие поведение генератора ключевого потока Геффе. В первой серии экспериментов решалась задача поиска всех решений системы, требующая обхода всего дерева. Для систем с достаточно большим линеаризационным множеством (свыше 30 переменных) для 12 процессов средняя эффективность равна 0,84. Во второй серии экспериментов параллельный поиск прекращался, как только один из процессов находил решение. Результаты показали, что в этом случае применение параллельного обхода дерева может сокращать время поиска в число раз, которое может быть как больше, так и меньше числа процессов. Причиной этого является несовпадение областей поиска при последовательном и параллельном обходах.

В связи с использованием метода линеаризационного множества для решения систем логических уравнений возникают задачи: существования линеаризационного множества заданной мощности для заданной системы S, нахождения кратчайшего линеаризационного множества для S, построения систем уравнений с ограничениями на линеаризационные множества и др. В общем случае эти задачи еще не нашли эффективного решения. Далее сообщается о результатах диссертационной работы в этом направлении.

Пусть B есть покрытие множества X. Подмножество $L \subseteq X$ будем называть *линеаризационным множеством покрытия* B, если $|Y - L| \leq 1$ для всех $Y \in B$. Если вернуться к системе логических уравнений S, то здесь X есть множество переменных системы, а B – множество его подмножеств, соответствующих конъюнкциям системы S, каждое из которых состоит из переменных, входящих в соответствующую ему конъюнкцию (с инверсией или без). В этом случае линеаризационное множество покрытия B будет линеаризационным множеством переменных системы S, хотя обратное может и не выполняться. В задачах построения линеаризационных множеств достаточно рассматривать только покрытия, не содержащие одноэлементных блоков. Обозначим $M(X)$ множество всех таких покрытий множества X.

Алгоритм А1 покрытию $B \in M(X)$ в соответствие ставит граф $G(V, E)$ такой, что $V = X$ и $\{u, v\} \in E$, если и только если существует $Y \in B$, что $u \in Y$, $v \in Y$ и $u \neq v$.

кратчайших допустимых разбиений конечных множеств» поддержанного грантом РФФИ (01-01-00774), 2003 г. и «Исследование и разработка математических и программных средств синтеза криптографически защищенных информационных систем» (грант ТОО-3.1-2851 Минобразования РФ по фундаментальным исследованиям в области технических наук), 2001-2002 гг.

Апробация работы. Результаты диссертации докладывались и обсуждались на IV Всероссийской конференции с международным участием «Новые информационные технологии в исследовании дискретных структур» (Томск, 2002), на международных научно-практических семинарах «Высокопроизводительные параллельные вычисления на кластерных системах» (Нижний Новгород, 2001, 2003), на II – V Сибирских научных школах-семинарах с международным участием «Компьютерная безопасность и криптография» (Томск, 2003; Иркутск, 2004; Томск, 2005; Шушенское, 2006), на 2-й и 3-й Сибирских школах-семинарах по параллельным вычислениям (Томск, 2003, 2005), на XV Международной школе-семинаре "Синтез и сложность управляющих систем" (Новосибирск, 2004), на X Юбилейной Международной научно-практической конференции студентов, аспирантов и молодых ученых «Современные техника и технологии СТТ'2004» (Томск, 2004), на XIV международной конференции «Проблемы теоретической кибернетики» (Пенза, 2005), а также на научных семинарах кафедры защиты информации и криптографии Томского госуниверситета.

Публикации. Основные результаты диссертации опубликованы в работах [1 – 15].

Структура и объем работы. Диссертация состоит из введения, трех глав, заключения и библиографии, включающей 40 наименований; её объем – 150 стр.

СОДЕРЖАНИЕ РАБОТЫ

В **главе 1** дается обзор результатов исследований, в основном зарубежных авторов, по параллельному обходу дерева и на его основе строится классификация существующих методов такого обхода. Указывается место результатов автора в этой классификации.

Предлагаются два метода параллельного обхода дерева – *метод назначаемых поддеревьев* (МНП) и *метод выделяемых поддеревьев* (МВП). В первом упор делается на сокращение взаимодействия между процессорами системы, во втором – на их равномерную загрузку. В обоих методах параллельные процессы подразделяются на управляющий и рабочие.

Идея МНП заключается в разбиении дерева на большое число «маленьких» поддеревьев и назначении их для обхода в порядке освобождения процессов. На первом этапе управляющий процесс выполняет построение и обход части дерева в ширину, начиная от его корня. По мере обхода вершины дерева добавляются и удаляются из специально используемой очереди. Обход ведется до тех пор, пока число вершин в очереди не достигнет заранее заданного числа m, которое *много больше* числа рабочих процессов. Вершины, попавшие в очередь, образуют множество *корневых вершин* назначаемых поддеревьев. На втором этапе

вершины из очереди управляющим процессом посылаются рабочим процессам по мере выполнения теми обходов в глубину ранее назначенных им поддеревьев, которые могут быть разного размера.

В данном методе управление работой процессов не требует значительных затрат. Рабочие процессы обмениваются данными с управляющим процессом лишь при назначении очередного поддерева. Вместе с тем метод не гарантирует равномерной загруженности всех процессоров, так как существует опасность, что некоторые процессы получат поддеревья, во много раз превышающие другие по размеру. Для предупреждения этого можно увеличить число m назначаемых поддеревьев. С другой стороны, это приведет к увеличению числа обменов между управляющим и рабочими процессами. Таким образом, МНП рекомендуется для использования в тех случаях, когда можно в результате разбиения получить поддеревья, близкие по размеру. Предварительная подготовка большого числа поддеревьев дает возможность в какой-то степени уравновесить загрузку используемых процессов и ускорить обход всего дерева за счет быстрого назначения освобождающимся процессам новых поддеревьев. Кроме того, метод легко применим для разработки параллельных алгоритмов решения конкретных задач.

В МВП разбиение на подзадачи происходит не сразу, но по мере выполнения обхода. На первом этапе управляющий процесс выполняет обход дерева в ширину, начиная от его корня, до тех пор, пока число вершин в очереди не достигнет числа s рабочих процессов. На втором этапе корневые вершины распределяются по процессам, и каждый процесс начинает выполнять обход в глубину соответствующего поддерева. По окончании обхода процесс освобождается. Для его загрузки выбирается поддерево, обход которого в этот момент ведется одним из рабочих процессов, и в нем выделяется поддерево, которое передается для обхода освободившемуся процессу. При выборе поддерева проверяется некоторое *условие разбиения*, показывающее целесообразность дробления этого поддерева. Например, максимальный номер полностью пройденного яруса должен быть много меньше числа всех ярусов в дереве.

Выбор процесса-источника для выделения поддерева выполняется управляющим процессом по стратегии *оптимального выбора*, нацеленной на то, чтобы направлять запрос к наиболее загруженному процессу. Каждый процесс периодически посылает управляющему информацию о пройденной части поддерева. Эти посылки выполняются на фоне вычислений. Управляющий на основании собранной им информации выбирает из рабочих процессов, для которых еще не нарушено условие разбиения, тот процесс, который имеет наибольшую непройденную часть поддерева, и пересылает ему номер освободившегося процесса. Получив указанный номер, процесс-источник выделяет в своем дереве поддерево для свободного процесса. Для выделения поддерева по возможности большего размера ищется непройденная вершина, минимально удаленная от корня, которая и становится корневой вершиной выделяемого поддерева. Поиск последней и

Теорема 2.2. Пусть $1 \le t < n - 1$, $(a_1, \ldots, a_{t-1})$ – префикс перестановки с номером I и M_{t-1} – его кратность. Пусть также $\{y_1, \ldots, y_{n-t+1}\} = N - \{a_1, \ldots, a_{t-1}\}$ и $y_1 < \ldots < y_{n-t+1}$. Тогда элемент a_t в перестановке с номером I есть y_k для $k = \lfloor (M_{t-1}+(n-1)!-1)/(n-t)! \rfloor$.

Далее описывается применение метода нумерации к перечислению разбиений множества $A = \{a_1, a_2, \ldots, a_n\}$. Пусть в нем элементы упорядочены некоторым фиксированным образом. Будем также фиксировать порядок блоков в любом разбиении множества A и представлять разбиение R вектором $p = (p_1, p_2, \ldots, p_n)$, в котором p_i есть номер блока, содержащего элемент a_i, причем $p_i \le i$. Таким образом, всегда $p_1 = 1$. При таком представлении любое разбиение множества однозначно определяется своим *представляющим вектором*, и перечисление разбиений можно осуществить как перечисление их векторов.

Будем говорить, что разбиение $R^{(t)}$ множества $\{a_1, \ldots, a_m, a_{m+1}, \ldots, a_{m+t}\}$ *порождено* разбиением R множества $\{a_1, \ldots, a_m\}$, если представляющий вектор разбиения R является префиксом представляющего вектора разбиения $R^{(t)}$. Число всех разбиений $R^{(t)}$, порожденных k-блочным разбиением R, не зависит от вида R и определяется однозначно величинами k и t. Обозначим его $D(k, t)$.

Теорема 2.3. Пусть $D(0, 0) = 1$, тогда верны следующие соотношения:

$D(k, 0) = 1$, $k > 0$;

$D(k, t) = k \cdot D(k, t - 1) + D(k + 1, t - 1)$, $k > 0$, $t > 0$.

Требуемые в методе нумерации величины для разбиений n-элементного множества определяются следующим образом: $p_1 = 1$, для $i = 2, \ldots, n$: $Z(p_1, p_2, \ldots, p_{i-1}) = \{1, 2, \ldots, k, k + 1\}$, где $k = \max\{p_1, p_2, \ldots, p_{i-1}\}$, и $N(p_1, p_2, \ldots, p_{i-1}) = D(k, n - i + 1)$.

Пусть K_m есть число всех разбиений с префиксом $(p_1, p_2, \ldots, p_m)$, номера которых не превосходят I, – *кратность* префикса: $K_1 = I$, $K_{m+1} = K_m - (p_{m+1} - 1)D(k, n - m - 1)$.

Теорема 2.4. Пусть $1 < m \le n$, $(p_1, p_2, \ldots, p_{m-1})$ – k-блочный префикс разбиения $R = (p_1, p_2, \ldots, p_n)$ с номером I и K_{m-1} – его кратность. Тогда если $K_{m-1} / D(k, n-m) \le k$, то $p_m = \lceil K_{m-1} / D(k, n-m) \rceil$, иначе $p_m = k + 1$.

Метод нумерации обеспечивает высокую эффективность, благодаря пропорциональному распределению вычислений и отсутствию обменов данными между процессорами, о чем свидетельствуют и результаты вычислительных экспериментов. Для параллельного перечисления сочетаний эффективность в среднем равна 0,87, перестановок – 0,94, эффективность параллельного алгоритма перечисления разбиений множества лежит в диапазоне 0,8 – 0,85.

Глава 3 содержит результаты исследования, связанного с параллельным решением методом линеаризационного множества системы S логических уравнений вида:

$$s_t = f_t(x_1, \ldots, x_n),\ t = 1, \ldots, m,$$

где $x_1, \ldots, x_n$ – булевы переменные, s_t – булевы константы, $f_t(x_1, \ldots, x_n)$ – булевы функции, представленные суммой по модулю 2 элементарных конъюнкций, $t = 1, \ldots, m$. Множество переменных $L \subseteq X = \{x_1, \ldots, x_n\}$ называется

$$N + \sum_{j=1}^{k-1} N(pz_j) \le I < N + \sum_{j=1}^{k} N(pz_j), \qquad (*)$$

то в объекте с номером I элемент с номером i есть $a_i = z_k$.

Таким образом, в методе нумерации вычисление объекта по его номеру I включает следующие шаги.

1. $p := ()$; $Z(p) := A_1$, $N := 1$.
2. Найти такое k, что $1 \le k \le |Z(p)|$ и верно (*).
3. $N := N + \sum_{j=1}^{k-1} N(pz_j)$, $p := pz_k$.
4. Вычислить $Z(p) = \{z_1, z_2, \ldots, z_r\}$.
5. Если $Z(p) = \varnothing$, то искомый объект представлен вектором p, конец; иначе перейти в п. 2.

Для применения метода к конкретным комбинаторным объектам требуется уметь вычислять число $N(p)$ и множество $Z(p)$.

В работе показывается применение метода нумерации для параллельного перечисления сочетаний, перестановок и разбиений множества. В каждом случае нумерация объектов выполняется в лексикографическом порядке представляющих их векторов, описывается алгоритм перечисления объектов в указанном порядке, даются выражения для $N(p)$ и $Z(p)$ в методе построения объекта по его номеру, формулируется алгоритм восстановления объекта по его номеру и приводится обоснование этого алгоритма в виде надлежащей теоремы.

Представляющий вектор для сочетания из n по k есть вектор $(b_1, b_2, \ldots, b_k)$ длины $k \le n$ с компонентами в $\{1, 2, \ldots, n\}$, в котором $b_i < b_{i+1}$ для $i = 1, \ldots, k-1$. Для сочетаний из n по k имеет место $Z(b_1, b_2, \ldots, b_{t-1}) = \{b_{t-1} + 1, b_{t-1} + 2, \ldots, n - k + t\}$ и $N(b_1, b_2, \ldots, b_{t-1}) = \binom{n - b_{t-1}}{k - t + 1}$, где $b_0 = 0$.

Обозначим s_j^t число всех сочетаний с префиксами $(b_1, b_2, \ldots, b_{t-1}, r)$, где $r \in \{b_{t-1}+1, b_{t-1}+2, \ldots, j-1\}$. Имеем: $s_j^t = \sum_{r=b_{t-1}+1}^{j-1} \binom{n-r}{k-t}$. Пусть также $s^t = \sum_{i=1}^{t-1} s_{b_i}^i$.

Теорема 2.1. Пусть $1 \le t \le k$ и $(b_1, b_2, \ldots, b_{t-1})$ есть префикс сочетания с номером I; тогда элемент b_t этого сочетания равен наименьшему из таких j, что $b_{t-1} < j < n - k + t$ и $I \le s^t + s_{j+1}^t$, если такие j существуют, и $b_t = n - k + t$ в противном случае.

В алгоритме параллельного перечисления перестановок методом нумерации требуется построить все перестановки $(a_1, \ldots, a_n)$ элементов множества $A = \{1, 2, \ldots, n\}$. Для таких перестановок $Z(a_1, \ldots, a_{t-1}) = A - \{a_1, \ldots, a_{t-1}\}$ и $N(a_1, \ldots, a_{t-1}) = (n - t + 1)!$.

Обозначим M_t число перестановок с префиксом $(a_1, \ldots, a_{t-1})$, номера которых не превосходят I, – *кратность* префикса. Имеем: $M_0 = I$ и $M_t = M_{t-1} - (k-1)(n-t)!$.

операции на исключение соответствующего ей поддерева требуют дополнительного времени, увеличивая время работы всей программы в целом. Обход дерева заканчивается при завершении обхода всех поддеревьев.

Таким образом, МВП позволяет обеспечить равномерную загруженность процессов системы, однако требует дополнительных накладных расходов на выделение поддеревьев.

Предложенные методы исследовались экспериментально на *полных k-ичных деревьях*. Целью эксперимента являлась оценка величин ускорения и эффективности. На практике *ускорение* определяется отношением времени решения задачи на одном процессоре ко времени решения той же задачи на системе таких же процессоров, а *эффективность* – отношением ускорения к числу процессоров. Эффективность показывает среднюю долю времени выполнения программы, в течение которого процессоры реально используются для решения задачи, и не превышает 1. Для программной реализации разработанных в диссертации параллельных алгоритмов использовались язык C++ и функции библиотеки MPI, представляющей собой стандартизованный набор средств для обмена сообщениями между процессорами. Компьютерное моделирование алгоритмов проводилось в основном на вычислительном кластере Томского госуниверситета, состоящем из 9 двухпроцессорных узлов. Результаты проведенного эксперимента показывают, что 1) эффективность для обоих методов зависит от размера дерева, а именно: по мере увеличения числа узлов в дереве эффективность возрастает и стабилизируется в среднем около 0,85; 2) при достаточно большом числе процессов (более 10) эффективность в среднем равна 0,88 для МНП и 0,89 для МВП.

Дальнейшее исследование методов проводилось на деревьях, возникающих при решении некоторых известных комбинаторных задач разных типов – перечислительных (перечисление сочетаний и разбиений множества), поисковых (поиск подмножеств с заданной суммой – решений задачи о рюкзаке) и оптимизационных (поиск оптимального назначения).

Дерево, используемое для *перечисления сочетаний*, интересно своей структурой, в том смысле, что в нем встречаются как поддеревья с очень большим числом вершин, так и много поддеревьев, небольших по числу вершин, более того, имеющих линейную структуру. В работе представлены пошаговые алгоритмы, описывающие действия управляющего и рабочих процессов при использовании МНП и МВП для параллельного обхода дерева. В частности, для МВП алгоритм включает операции выделения поддерева, число которых в данной задаче невелико. В экспериментах испытывались различные деревья сочетаний из n по k, в том числе: содержащие максимальное для заданного n число концевых вершин (например, $n = 40$, $k = 20$), «глубокие» ($n = 100$, $k = 93$) и «широкие» ($n = 100$, $k = 8$). Результаты показали, что для МНП при правильном выборе числа m назначаемых поддеревьев почти во всех случаях достигаются достаточно равномерная загрузка рабочих процессов и высокая эффективность (для 16 процессов $\approx 0{,}9$). Для деревьев с числом узлов $10^{10} - 10^{11}$ наиболее

подходящим оказалось значение m порядка 10^5. Исключение составляют «глубокие» деревья. Причина в том, что, например, для $n = 100$, $k = 93$ и при значении m порядка 10^5 одно из назначаемых поддеревьев содержит примерно половину всех сочетаний, а при m порядка 10^6 – одну пятую. Средняя достигнутая эффективность при использовании МВП также достаточно высока (≈ 0,86), в том числе и для «глубоких» деревьев.

В основе параллельных алгоритмов *перечисления разбиений множества* лежит рекурсивный алгоритм обхода в глубину дерева, каждая вершина в котором представляет разбиение некоторого подмножества. Экспериментальным путем установлено, что для достижения эффективности, близкой к 1, число корневых вершин в МНП должно быть порядка 10^5 для разбиений множеств мощности 15, 16, 17 и порядка 10^6 – для мощности 18. Для МВП проведено исследование влияния условия разбиения, определяющего минимальную высоту выделяемого поддерева, на ускорение вычислений. Наилучшие результаты (эффективность ≈ 0,9) получены при запрете на выделение поддеревьев высотой меньше 6.

В классе задач поиска комбинаторных объектов с заданными свойствами рассмотрена *задача о рюкзаке* в следующей постановке: заданы натуральное число ω и набор неотрицательных чисел $B = (b_1, b_2, \ldots, b_n)$; требуется найти, если они существуют, все такие подмножества $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$, для которых $\sum_{r=1}^{k} b_{i_r} = \omega$. При решении этой задачи методом поиска с возвращением выполняется обход в глубину дерева, которое, в отличие от деревьев для сочетаний и разбиений, не имеет определенной (заранее предсказуемой) структуры. Для его распараллеливания использовались оба метода. В МНП средняя эффективность получилась равной 0,86 для 12 процессов. Для параллельного алгоритма, разработанного по МВП, исследовались две версии, отличающиеся друг от друга условием разбиения. В первой указывался максимальный номер яруса M, на котором может быть расположена корневая вершина выделяемого поддерева, во второй – минимальное число узлов N в разбиваемом поддереве. Для оценки числа узлов в поддереве использовался метод Монте-Карло. Экспериментальный анализ показал, что вторая версия алгоритма не дает предполагаемого выигрыша по сравнению с первой. При хорошем подборе значений M и N в первой и второй версиях в среднем получаются одинаковые результаты по времени, но худшие, чем для МНП.

Задача о назначении, состоящая в нахождении для заданных положительных чисел a_{ij} перестановки $(k_1, k_2, \ldots, k_n)$ чисел 1, 2, …, n с минимальным значением целевой функции $W = \sum_{i=1}^{n} a_{ik_i}$, относится к классу задач поиска оптимального решения. В задачах данного типа в каждой вершине дерева проводится анализ, связанный с полученным на данный момент лучшим решением, по результатам которого продолжается ветвление из этой вершины, либо выполняется возврат на предыдущий уровень. В связи с этим, при нахождении во время параллельного

обхода дерева некоторым из процессов лучшего решения соответствующее ему значение W следует обновить у всех процессов, что в многопроцессорных системах, не имеющих общей памяти, требует явной передачи. Если рабочий процесс находит в своем поддереве назначение со стоимостью $W' < W$, то заменяет W на W' и пересылает W' управляющему процессу. Обновление значения целевой функции в МНП выполняется управляющим процессом только с назначением очередного поддерева, а в МВП сразу же после получения. Результаты экспериментального анализа показали, что при решении данной задачи преимущество имеет МНП.

Глава 2 посвящена параллельному перечислению комбинаторных объектов методом нумерации. Схема алгоритма перечисления строится обычно по следующим правилам: на множестве перечисляемых объектов вводится некоторый порядок, описывается способ преобразования текущего объекта в следующий за ним и формулируется условие окончания перечисления. Для распараллеливания данной схемы множество перечисляемых объектов можно предварительно разбить на классы, мощности которых отвечают производительности процессоров в системе, и перечислять объекты в различных классах независимо и параллельно на соответствующих процессорах. Однако, когда речь идет о множестве комбинаторных объектов, в отличие, скажем, от числового множества, провести такое разбиение в явном виде не представляется возможным. Решение этой проблемы возможно путем сопоставления перечисляемым комбинаторным объектам элементов некоторого “легко разбиваемого” множества. Самый простой способ состоит в нумерации комбинаторных объектов числами 1, 2, … в соответствии с установленным на множестве объектов порядком. Последний, естественно, должен допускать восстановление объекта по его номеру, причем время, требуемое на восстановление, должно быть пренебрежимо мало по сравнению со временем, затрачиваемым на перечисление всех объектов одного класса. Предложенный *метод нумерации* параллельного перечисления комбинаторных объектов включает метод построения объекта по его номеру в заданном порядке.

Пусть объекты задаются векторами конечной длины вида $(a_1, a_2, \ldots, a_n)$ с компонентами из конечных упорядоченных множеств $A_1, A_2, \ldots, A_n$ $(a_i \in A_i)$. Будем считать, что объекты упорядочены в соответствии с лексикографическим порядком представляющих их векторов, и их нумерация начинается с 1. Требуется найти объект с номером I.

Искомый объект строится покомпонентно: вначале вычисляется a_1, затем a_2, и т.д. Пусть $p = (a_1, a_2, \ldots, a_{i-1})$ уже известный префикс искомого объекта (представляющего объект вектора) и N – номер первого объекта с таким префиксом. Обозначим $Z(p) = \{z_1, z_2, \ldots, z_r\} \subseteq A_i$ множество значений, которые может принимать элемент a_i при заданном префиксе p. Множество объектов с префиксом p можно разбить на классы объектов с префиксами (pz_j) для всех $j = 1, \ldots, r$. Пусть $N(p)$ есть число всех векторов, имеющих префикс p. Тогда, если для некоторого целого k между1 и $|Z(p)|$