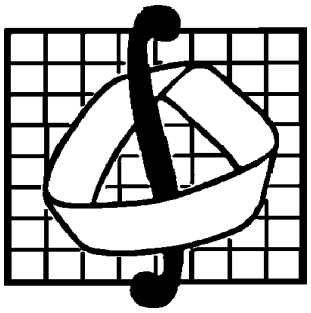
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ
имени М.В.ЛОМОНОСОВА



Механико-математический факультет

Э.Э.Гасанов

Теория сложности информационного поиска

Москва 2005 год

Э.Э.Гасанов

Теория сложности информационного поиска

Учебное пособие написано на основе специальных курсов "Теория баз данных и информационного поиска" и "Теория интеллектуальных систем", читаемых на кафедре математической теории интеллектуальных систем механико-математического факультета МГУ им. М.В.Ломоносова. В книге вводится новый вид представления баз данных, называемый информационно-графовой моделью данных, обобщающий известные ранее модели данных. Рассматриваются основные типы задач поиска информации в базах данных и исследуются проблемы сложности решения этих задач применительно к информационно-графовой модели. Приводятся алгоритмы решения рассматриваемых задач поиска близкие к оптимальным.

Для студентов, аспирантов, специализирующихся в области математической кибернетики, дискретной математики и математической информатики.

Рецензент — доктор физико-математических наук, академик, профессор В. Б. Кудрявцев.

Оглавление

Введение

В последние десятилетия активно развивается новое научное направление, связанное с оптимальным хранением и поиском информации, именуемое теорией информационного поиска. Одним из главных носителей этого направления является теория баз данных. Возникшее под влиянием практических задач, оно и сейчас в основном обслуживает приложения, а собственно теоретическая его часть, как представляется, обретает контуры. Как всякая научная дисциплина это направление должно характеризоваться следующими чертами: предметом исследования, проблематикой, методами и результатами. В развитой теории каждая из этих черт должна иметь достаточно общий характер. В то же время важно отметить, что молодые дисциплины возникают, как правило, через рассмотрение отдельных конкретных важных примеров, которые затем с развитием дисциплин обобщаются как в постановочной, так и в проблемно-методологической частях. Подобных примеров достаточно много в кибернетике, информатике и других разделах науки. К числу характерных из них может быть отнесена теория управляющих систем. В своей практической деятельности человек столкнулся с конкретными видами таких систем, которые далее играли роль модельных управляющих систем. В инженерном деле — это вентильные и контактные схемы, схемы из функциональных элементов и некоторые другие, в математике — формулы, алгоритмы и т.д., в биологии — нейроны, нейронные сети, автоматы и т.п. Для этих видов управляющих систем рассматривались две такие главные задачи анализа и синтеза соответствующих управляющих систем. Первая состояла в изучении "по-

ведения"таких систем, а вторая — в создании соответствующей системы с заданным "поведением". На первом этапе эти постановки связывались непосредственно с модельными системами и для каждой из них разрабатывался конкретный метод решения. Со временем наступил этап, когда большинство из этих систем могли уже рассматриваться с единых позиций и исследование указанных общих задач достигалось уже с помощью общих методов решения [3]. Хотя по-прежнему модельные управляющие системы, имея свою специфику, продолжают оставаться в центре внимания теории управляющих систем.

Аналогичный путь развития проходит теория информационного поиска. В ней также первоначально возникли конкретные примеры способов хранения и представления данных и соответствующих этим способам алгоритмов поиска информации. К их числу относятся: лексикографические, древовидные, реляционные и др. Задачи поиска для них имеют конкретные виды и модификации, такие как: задача поиска идентичных объектов, задача о близости, включающий поиск и др. Они играют роль модельных задач для выбранных способов хранения информации и изучались на протяжении многих лет, привлекая каждый раз для своего решения специальные исследовательские средства, которые носили ограниченный по своим возможностям характер.

Тем самым можно считать, что современное состояние теории информационного поиска напоминает то состояние теории управляющих систем, которое соответствовало первому этапу развития последней, когда накапливались данные лишь о конкретных видах модельных управляющих систем. В то же время, как выяснилось, опыт развития теории управляющих систем в своей методологической части давал возможность сделать попытку с более общих позиций провести исследование как модельных баз данных, так и модельных задач для них, с соответствующей разработкой достаточно общей теории.

Это и составляет содержание исследований данной книги. В ней предлагается новая модель данных (частными случаями которой могут считаться уже известные) с наследственно определенными средствами поиска информации, с соответствующими понятиями сложности такого поиска, а также разрабатываются

основы теории решения базовых задач поиска применительно к этой модели. Если продолжить аналогию с теорией синтеза управляющих систем, то можно отметить, что различным видам управляющих систем соответствуют различные виды хранения и представления данных (модели данных), классам функций, исследуемым в теории синтеза, соответствуют типы задач поиска, исследуемые в теории информационного поиска. И в теории синтеза и в теории поиска вводятся понятия сложности и ставятся задача оптимального синтеза и задача исследования функций сложности шенноновского типа. Таким образом, мы стремимся к тому, чтобы приблизить состояние теории информационного поиска по степени продвинутости к современному состоянию теории управляющих систем.

Уточним сказанное. Во-первых, мы предлагаем новую формализацию понятия задачи поиска. Тип задач поиска охватывает класс однотипных вопросов к базе данных. Тип задач поиска заключает в своем определении три объекта. Множество запросов, множество записей и бинарное отношение, заданное на декартовом произведении этих множеств, называемое отношением поиска. Здесь запись — это поисковый образ элемента данных, т.е. поле или множество полей элемента данных, которые представляют интерес в данном типе вопросов. Запрос — это минимальный элемент, содержащий суть вопроса. Запрос совместно с отношением очерчивает тот круг объектов, которые отвечают на данный вопрос. Задача поиска заданного типа получается выделением из множества записей конечного подмножества, называемого библиотекой. А именно, задача поиска состоит в том, чтобы по произвольному запросу перечислить, все записи из библиотеки, находящиеся в заданном отношении с запросом (удовлетворяющие запросу). При фиксации отношения поиска каждая запись задает предикат, определенный на множестве запросов, который равен 1, если данная запись удовлетворяет запросу — аргументу функции. Поэтому если вернуться к аналогии с теорией синтеза управляющих систем, то тип задач поиска есть способ описания некоторого конкретного класса предикатов, задаваемых на множестве запросов, а задача поиска — это конкретное подмножество предикатов из этого класса.

Во-вторых, мы предлагаем новую управляющую систему, называемую информационным графом, которая в общей иерархии теории управляющих систем находится в не очень высоких слоях залегания и является в некотором смысле обобщением контактных схем. Фактически нам нужны лишь графы, дискретные функции и вычисление волновых процессов на графах, и этого хватает, чтобы с достаточно общих позиций посмотреть на ту разрозненную картину, которая наблюдается в теории информационного поиска.

В предлагаемой информационно-графовой модели данных структура данных задается ориентированным графом (называемым информационным), ребра и вершины которого нагружены элементами данных и функциями, определенными на множестве запросов. В графе выделена одна вершина, называемая корнем и ассоциируемая со входом, а вершины графа, нагруженные элементами данных ассоциируются с выходами. Этот же граф описывает алгоритм поиска, на вход которого поступает запрос, а на выходе получается некоторое подмножество данных. При этом процесс поиска начинается с корня и распространяется в зависимости от значений нагрузочных функций на запросе возможно сразу по нескольким направлениям. Если этот волновой процесс на графе достигает элементов данных, то эти элементы включаются в ответ алгоритма на исходный запрос. Информационный граф будет решать некоторую задачу поиска, если для произвольного запроса ответ на этот запрос содержит все те и только те записи из библиотеки, которые удовлетворяют запросу. Таким образом информационный граф с одной стороны дает новую концепцию хранения данных, а с другой стороны предлагает новый подход к поиску информации как волнового процесса на графах, управляемого нагрузочными функциями. Нагрузочные функции, которые называются базовыми, разделены на два класса — предикаты и переключатели, и являются одним из основных управляющих параметров модели. Нагрузочные функции по сути определяют функции проводимости между вершинами графа, и проблема нахождение решения задачи поиска сводится к проблеме синтеза информационного графа, реализующего систему функций, задаваемую задачей поиска.

Информационные графы позволяют ввести новое понятие сложности поиска. Это понятие новое как с точки зрения теории управляющих систем, так и с точки зрения теории баз данных. В теории управляющих систем обычно под сложностью понимается или число ребер, или число элементов-функций, а здесь сложность понимается как часть графа, захваченного волновым процессом, и существенно зависит от значений нагрузочных функций, и тем самым не является просто количественной характеристикой графа, такой как число ребер или вершин. Новизна же в теории баз данных заключается в том, что такое введение сложности после осреднения по множеству запросов адекватно соответствует среднему времени поиска — традиционно трудной для изучения характеристики алгоритмов поиска информации. Кроме того, в информационных графах совсем просто контролируется такой важный управляющий параметр в задачах информационного поиска, как объем памяти, который в данном случае характеризуется количеством ребер графа.

На основе информационно-графовой модели можно предложить новую технология проектирования физической организации баз данных (БД). По этой технологии на начальном этапе выделяются классы однотипных вопросов к БД, оформляемые в виде типов задач поиска. Множество данных БД задает конкретную задачу поиска данного типа. Для каждой задачи поиска выделяется множество элементарных операций над запросами, оформляемое в виде базового множества, и решается задача синтеза оптимального информационного графа, решающего данную задачу поиска. Полученный информационный граф описывает оптимальную структуру данных, соответствующую заданным целям оптимизации (среднему времени поиска, времени поиска в худшем случае, объему памяти).

Тем самым один информационный граф описывает структурную часть БД, обрабатывающую один класс однотипных вопросов к БД. А сама БД в информационно-графовой модели представляется как совокупность нескольких информационных графов, охватывающих весь спектр вопросов к базе данных.

Учитывая тот факт, что в работе рассматриваются наиболее распространенные типы задач поиска, то решение проблемы оптимального синтеза для данных задач позволяет для боль-

шинства случаев, возникающих при проектировании физической организации баз данных, иметь готовые рекомендации.

Содержание книги составил материал, который использовался при чтении обязательных и специальных курсов в МГУ им. М.В.Ломоносова по кафедре Математической теории интеллектуальных систем на факультетах механико-математическом факультет и вычислительной математики и кибернетики. Отдельные главы читались как специальные курсы в РГГУ (факультет защиты информации) и в НУЦ МГУ – ВЦ РАН – РГГУ "Интеллектуальные системы и нечеткие технологии"(ФЦП "Интеграция", грант №431, направление 2.1).

Автор выражает особую благодарность В.Б.Кудрявцеву и А. С. Подколзину, предложения которых, советы и обсуждения отличались характерной для них щедростью.

Автору приятно поблагодарить весь коллектив кафедры Математической теории интеллектуальных систем механико- математического факультета Московского государственного университета им. М. В. Ломоносова, чье постоянное внимание, доброжелательная критика и советы способствовали появлению этой книги.

Глава 1

Информационно-графовая модель данных

1.1 Понятие информационного графа (ИГ)

Управляющая система, функционирующая в среде, — это центральное понятие кибернетики.

Управляющие системы можно разбить на два класса: те, которые действуют без памяти (рефлекторно), и те, которые имеют память. Второй класс неизмеримо богаче, чем первый, и изучается в первую очередь. Примеры управляющих систем без памяти: вирусы, микробы, разменные аппараты. Высшие животные, человек, в технике — адаптивные системы, компьютеры — управляющие системы с памятью. Естественно возникает вопрос оптимальной организации памяти в таких системах, которую можно отождествить с содержательным пониманием баз данных.

База данных (БД) — формализованное представление информации, удобное для хранения и поиска данных в нем. Понятие БД возникло в 60-е годы 20 века и связано с развитием

вычислительной техники и информатики. Тематика теории БД связана с поиском удобного представления, компактного хранения, быстрого поиска, защищенности и других свойств данных.

Одно из основных направлений теории БД связано с вопросами сложности алгоритмов обработки данных. И именно этому направлению посвящена данная книга.

Это направление связано с проблематикой физической организации баз данных. При физической организации баз данных мы имеем дело не с представлением данных в прикладных программах, а с их размещением на запоминающих устройствах.

Критерии, определяющие выбор физической организации, отличаются от тех, которые определяют выбор логической организации данных. При выборе физической организации решающим фактором является эффективность, причем согласно Дж. Мартину [21] на первом месте стоит обеспечение эффективности поиска, далее идут эффективность операций занесения и удаления и затем обеспечение компактности данных.

На первом шаге нам необходимо ввести понятие задачи информационного поиска.

В понятие задачи поиска исследователи вкладывают по крайней мере 3 различных смысла.

Специалисты в теории исследования операций понимают задачи поиска как задачи управления сближением одной системы (поисковой) с другой (искомым объектом) по неполной априорной информации. Понимается, что цель поиска — это обнаружение искомого объекта, определяемое как выполнение определенных терминальных условий. Интенсивно проблемой поиска подвижных объектов начали заниматься в период Второй мировой войны. Интерес к этой проблеме в тот период был вызван необходимостью разработки тактики борьбы против подводных лодок. Среди работ, посвященных этой тематике, можно выделить, например, работы О. Хеллмана [30] и Д. П. Кима [14].

Другое понимание задач поиска можно найти в книге Р.Альсведе, И.Вегенера "Задачи поиска"[2]. Приведем поясняющий пример из этой книги.

Во время Второй мировой войны все призываемые в армию США подвергались проверке на реакцию Вассермана. При этом проверялось, есть ли в крови обследуемого определенные анти-

тела, имеющиеся только у больных. Во время этого массового обследования было замечено, что разумнее анализировать пробу крови целой группы людей. Если такая объединенная проба крови не содержит антител, то, значит, ни один из обследуемых не болен. В противном случае среди них есть хотя бы один больной. Хороший алгоритм поиска для этой задачи — это тот, который для типичной выборки людей позволяет "наискорейшим образом"выявлять множество всех больных.

Другой пример мы находим в статье Д.Ли, Ф.Препараты "Вычислительная геометрия"[18]. Дано множество горизонтальных и вертикальных отрезков. Надо найти все точки пересечения отрезков.

Эти два примера объединяет то, что в обоих случаях поиск производится однократно. Это порождает свои особенности таких задач поиска. Как правило, данные в этих задачах не имеют сложной организации. Фиксация множества, в котором производится поиск, однозначно определяет множество найденных объектов. "Хорошесть"алгоритма поиска определяется при варьировании множества, в котором производится поиск.

В третьем понимании задачи поиска предполагается многократное обращение к одним и тем же данным, но возможно каждый раз с разными требованиями к искомым объектам, то есть с разными запросами на поиск. Такие задачи поиска обычно возникают в системах, использующих базы данных. Многократное использование порождает особую проблему — проблему специальной организации данных, направленной на последующее ускорение поиска. Процесс такой специальной организации данных, проводимый до того, как осуществляется поиск, называется предобработкой и часто может занимать очень большое время, которое затем окупается сторицей в результате многократности поиска. Простейшим примером предобработки является сортировка. Построение "хорошего"алгоритма поиска в этом случае сводится к нахождению хороших структур данных, то есть к осуществлению такой хорошей предобработки данных, которая обеспечила бы хорошую скорость поиска. "Хорошесть"алгоритма поиска в этом случае определяется варьированием запроса на поиск, например, как среднее время поиска на запросе.

В зависимости от того, является ли база данных фиксированной или изменяется на протяжении времени работы с ней, мы имеем два типа организации баз данных: *статическую* и *динамическую* соответственно.

Задачи поиска, возникающие в статических базах данных и предполагающие многократное обращение к одним и тем же данным, и являются объектом исследования в данной работе.

Приведем примеры таких задач поиска.

Простейшим и самым распространенным примером задачи поиска, встречающейся в любой базе данных, является задача поиска по ключу. Суть ее состоит в том, что любой объект в базе данных имеет свой уникальный ключ. Это может быть порядковый номер, уникальное имя, или уникальное значение некоторого поля, например, номер паспорта. Задача состоит в том, чтобы по заданному в запросе ключу найти в базе данных объект с этим ключом (если такой объект в базе имеется). Более формально эту задачу можно поставить следующим образом. Имеется некоторое конечное множество ключей. Имеется некоторое более широкое множество запросов. Требуется по произвольному запросу из множества запросов найти в множестве ключей ключ идентичный (равный) ключу-запросу. В такой постановке эта задача называется задачей поиска идентичных объектов.

Другой пример взят из систем машинной графики, обработки изображений и систем автоматизированного проектирования Дано конечное множество точек из отрезка $[0, 1]$ вещественной прямой. Множество запросов есть множество всех отрезков, содержащихся в $[0, 1]$. Надо для произвольного запроса $[u, v]$ из множества запросов перечислить все точки из нашего множества точек, которые попадают в отрезок $[u, v]$. Эта задача носит название одномерной задачи интервального поиска.

Если мы хотим всерьез изучать сложность алгоритмов поиска, то без введения математической модели объекта изучения нам не обойтись.

Поэтому начнем с формализации понятия задачи информационного поиска (ЗИП). В работах [4, 26, 27, 40] вводились различные формализации ЗИП. Так в [40] вводилась формализация наиболее близкая к той, которую мы будем использовать

в данной работе. Формализация ЗИП в [40] вводилась следующим образом: вопрос к базе данных представляет некоторый запрос, имеющий тип $T1$, сама база данных состоит из элементов типа $T2$, а ответ на вопрос — значение типа $T3$, например, $T3$ может иметь логический тип, если предполагается, что ответ на запрос должен быть "да"или "нет", $T3$ может совпадать с $T2$, если ответом на запрос является элемент базы данных, и наконец $T3$ может быть множеством элементов типа $T2$, если в ответ на запрос надо перечислять некоторые элементы из базы данных. Вопрос Q рассматривается как отображение из $T1$ и множества подмножеств $T2$ в $T3$, то есть $Q: \ T1 \times 2^{T2} \to T3$.

Мы будем рассматривать только такие задачи поиска, в которых в ответ на запрос надо перечислить элементы базы данных, удовлетворяющие запросу. ЗИП такого типа называются задачами на перечисление. С учетом этого факта мы и дадим собственную формализацию ЗИП, более удобную для использования в дальнейшем.

Из приведенных примеров видно, что в задачах поиска имеется некий универсум, из которого берутся объекты поиска (элементы базы данных). В первом примере таким универсумом является множество всевозможных ключей, а во втором — отрезок $[0, 1]$ вещественной прямой. Этот универсум обозначим через Y и будем называть множеством объектов поиска или *множеством записей*, а элементы множества Y, будем называть, *записями*.

Далее в задачах поиска всегда имеется *множество запросов*. В первом примере множество запросов совпадает с множеством объектов поиска и является множеством всевозможных ключей, а во втором примере множество запросов есть множество всех отрезков, содержащихся в $[0, 1]$, то есть множество $\{(u, v) : 0 \leq u \leq v \leq 1\}$. Множество запросов будем обозначать через X.

На декартовом произведении $X \times Y$ имеется бинарное отношение, которое позволяет устанавливать, когда запись из Y удовлетворяет запросу из X. Это отношение будем называть *отношением поиска*. В первом примере отношение поиска есть отношение идентичности (равенства), то есть запись удовлетворяет запросу, если они идентичны. Во втором примере отноше-

ние поиска, которое обозначим через ρ_{int}, определяется соотношением

$$(u, v)\rho_{int}y \Longleftrightarrow u \leq y \leq v, \tag{1.1}$$

где $(u, v) \in X,\ y \in Y$.

Тройку $S = \langle X, Y, \rho\rangle$, где X — множество запросов, Y — множество записей, ρ — отношение поиска, заданное на $X \times Y$, будем называть *типом*, или иногда более развернуто *типом задач информационного поиска*.

Тройку $I = \langle X, V, \rho\rangle$, где X — множество запросов; V — некоторое конечное подмножество множества Y, в дальнейшем называемое *библиотекой*; ρ — отношение поиска, заданное на $X \times Y$, будем называть *задачей информационного поиска* (ЗИП) типа $S = \langle X, Y, \rho\rangle$. Содержательно будем считать, что задача $I = \langle X, V, \rho\rangle$ состоит в перечислении для произвольно взятого запроса $x \in X$ всех тех и только тех записей из V , которые находятся в отношении ρ с запросом x, то есть удовлетворяют запросу x.

Тем самым мы формализовали понятие ЗИП.

Для полной определенности отметим, что поскольку библиотека V есть множество, то в ней все элементы различные, то есть в ней нет повторяющихся элементов.

Следующий шаг, который мы обязаны сделать — это выбрать математическую модель для алгоритмов поиска.

Отметим, что здесь и всюду далее, используя термин "алгоритм", мы часто будем подменять им понятие "условного алгоритма"(или "относительного алгоритма", см. [20, с. 44–45]), то есть будем рассматривать алгоритмы, которые выполняются при условии, что мы умеем выполнять некоторые операции из описанного заранее множества. В качестве таких операций могут выступать, например, некоторые операции над вещественными числами, но так или иначе мы всегда будем явно оговаривать операции, относительно которых рассматривается каждый описываемый условный алгоритм.

Переберем возможных кандидатов на роль математической модели для алгоритмов поиска.

1) Алгоритмы поиска информации можно описывать на языке программирования, например на Си++ [29]. Положительным фактором такого подхода является то, что любое описание

алгоритма одновременно будет и его реализацией. Отрицательным — то, что помимо описания алгоритма нам понадобится по алгоритму определять его сложность, а в этом случае это сделать весьма трудно, более того неясно, как сделать этот переход формально. Как следствие, мы не можем использовать этот язык для получения нижних оценок сложности алгоритмов.

2) Алгоритмы поиска можно описывать с помощью машин Тьюринга [17, 33, 64]. Если под сложностью алгоритма понимать время поиска, этот подход обладает тем же недостатком, что и первый, но плюс к этому добавляется трудность программирования для машин Тьюринга.

3) Для описания алгоритмов поиска можно использовать схемы алгоритмов Янова [13, 34] или стандартные схемы программ Котова [16]. Неудобство использования этих схем определяется тем, что они предназначены для других целей, а именно для исследования проблемы эквивалентности алгоритмов. Поэтому если мы даже сможем адаптировать эти понятия для наших нужд, то пропадает главное преимущество, которое дает использование известных моделей — это наработанный аппарат.

4) В качестве модели вычислений можно использовать машину с произвольным доступом к памяти, аналогичную описанной в [4] с добавлением возможности выполнения арифметических операций над действительными числами. Это значит, что в такой машине каждая ячейка памяти может содержать действительное число, а каждая арифметическая операция, такая, как сложение, умножение и деление, может быть выполнена за единицу времени. В зависимости от решаемой задачи машина может иметь некоторые другие примитивные операции, о которых предполагается, что они выполняются за постоянное время.

Известной моделью, используемой для исследования сложности алгоритмов, является алгебраическое дерево вычислений [36, 46, 60, 61].

Пусть $\mathbf{R}$ — множество действительных чисел. *Алгебраическое дерево вычислений* (АДВ) [36, Бен-Ор] на множестве переменных $W = \{x_1, x_2, \ldots, x_n\}$, где $x_i \in \mathbf{R}$, — бинарное дерево D, размеченное следующим образом.

1. Каждой вершине v, имеющей в точности одного сына (про-

стая вершина), приписывается операция вида $f_v := f_{v_1} \# f_{v_2}$, или $f_v := f_{v_1} \# c$, или $f_v := \sqrt{f_{v_1}}$, где v_i $(i = 1, 2)$ — предок вершины v в дереве D, или $f_{v_i} \in W$, $\# \in \{+, -, \times, /\}$, а $c \in \mathbf{R}$ является константой.

2. Каждой вершине v, имеющей в точности двух сыновей (вершина ветвления), приписывается операция сравнения вида $f_{v_1} > 0$, или $f_{v_1} < 0$, или $f_{v_1} = 0$, где v_1 — предок вершины v в дереве D, или $f_{v_1} \in W$.

3. Каждому листу дерева приписывается одно из значений ДА или НЕТ.

Приведем пример использования АДВ, например, для задачи поиска идентичных объектов $I = \langle X, V, = \rangle$ в случае, когда множества запросов $X = [0, 1] \subset \mathbf{R}$ и $V \subset [0, 1]$. Для каждого заданного запроса $x \in X$ программа прокладывает в дереве D путь $P(x)$, начинающийся в корне. При прохождении каждой простой вершины выполняется арифметическая операция, приписанная этой вершине, а в каждой вершине ветвления происходит ветвление в соответствии с результатом сравнения, приписанного вершине. При достижении листа дерева возвращается ответ ДА или НЕТ. Считается, что АДВ D решает задачу I, если возвращаемый ответ верен для любого запроса $x \in X$. Сложность дерева D, обозначаемая $C(D)$, определяется как максимум величины $cost(x, D)$ по всем значениям x, где $cost(x, D)$ — число вершин, проходимых путем $P(x)$ в дереве D. Сложность задачи I, обозначаемая $C(I)$, есть минимум $C(D)$ по всем алгебраическим деревьям вычислений D, решающих задачу I.

Разновидностью алгебраического дерева вычислений является так называемое (алгебраическое) *дерево решений порядка* d [63]. Дерево решений порядка d — это дерево, каждой вершине которого соответствует сравнение вида $f(x)?0$, где f имеет полиномиальную сложность относительно входных данных с показателем степени не более d и $? \in \{>, <, =\}$. В случае, когда d равно 1, получается *линейное дерево решений*, с использованием которого получены доказательства ряда нижних оценок сложности [46, 61, 66]. Серьезные исследования временной сложности алгебраических деревьев решений проводились М. Ю. Мошковым [23].

В чем заключаются недостатки использования алгебраического дерева вычислений Бен-Ора или алгебраического дерева решений порядка d для моделирования алгоритмов поиска? Во-первых, хотелось бы более полно учитывать специфику алгоритмов поиска. Во-вторых, использование алгебраического дерева вычислений Бен-Ора или алгебраического дерева решений порядка d приводит к некоторым заблуждениям. Так в рамках АДВ (так же как и в рамках линейного дерева решений) задача поиска идентичных объектов $I = \langle X, V, = \rangle$ в случае, когда множества запросов $X = [0,1] \subset \mathbf{R}$ и $V \subset [0,1]$, имеет нижнюю оценку сложности $\underline{O}(\log_2 n)$, где n — число точек в библиотеке V. Это так называемая *теоретико-информационная оценка* [43]. Понятно откуда получается эта оценка — бинарное дерево с n листьями должно иметь высоту как минимум $\log_2 n$. Сила влияния этой оценки настолько велика и заблуждение о неизбежности такого времени для решения настолько глубоко, что часто алгоритмы с оценкой $\underline{O}(\log_2 n)$ называют оптимальными. Тогда как эта оценка всего лишь следствие бинарности дерева решений, и если отказаться от бинарности, то не будет и оценки. С помощью этой оценки получается нижняя оценка сложности порядка $\underline{O}(n \log_2 n)$ для задачи сортировки множества из n элементов [4]. На основе оценки сложности сортировки с помощью метода сведения одной задачи к другой можно показать, что большое количество задач требуют для своего решения время $\underline{O}(n \log_2 n)$ в рамках АДВ-модели вычислений в том числе задача построения выпуклой оболочки n точек на плоскости, задача построения евклидова минимального остовного дерева и т. д.

Но если отказаться от АДВ-модели вычислений, то для нашей задачи поиска идентичных объектов $I = \langle X, V, = \rangle$ легко предложить алгоритм, который решает эту задачу за константное время, при условии, что можно осуществлять предобработку и использовать дополнительную память. Пусть $V = \{y_1, \ldots, y_n\}$ — упорядоченное по возрастанию множество точек. Найдем расстояние между двумя ближайшими точками. Пусть оно равно d. Поделим отрезок $[0,1]$ на $m =]1/d[$ равных частей, так что i-ая часть ($i = \overline{0, m-1}$) имеет вид $(i/m, (i+1)/m]$. Тогда в каждую часть попадет не более одной точки из V. Заведем целочисленный массив A длины m, i-ый элемент которого

($i = \overline{0, m-1}$) содержит 0, если в i-ой части нет точек из V, и номер j, если в i-ую часть попала точка y_j из V. После такой предобработки поиск по произвольному запросу $x \in [0,1]$ будем осуществлять следующим образом. Вычислим $j = [x \cdot m]$. Номер j дает номер части, в которую попала точка x. Если $A_j = 0$, то точки x в библиотеке V нет, иначе сравниваем y_{A_j} с x и если они равны, то точка x найдена.

Мы откажемся от использования АДВ-модели, в частности, чтобы не быть зажатыми теоретико-информационной оценкой [43].

5) Прежде чем предложить еще одну модель алгоритмов поиска взглянем на алгоритм, решающий задачу поиска с функциональной точки зрения.

Рассмотрим произвольную ЗИП $I = \langle X, V, \rho \rangle$. Алгоритм поиска решает ЗИП, если на любой запрос из множества запросов X он выдает все те и только те записи из V, которые удовлетворяют запросу. Возьмем произвольную запись $y \in Y$. Для нее можно ввести *характеристическую функцию*

$$\chi_{y,\rho}(x) = \begin{cases} 1, & \text{если } x \rho y \\ 0 & \text{в противном случае} \end{cases},$$

то есть она равна 1 на тех запросах, которым удовлетворяет запись y. Тогда можно сказать, что алгоритм, решающий ЗИП $I = \langle X, V, \rho \rangle$, где $V = \{y_1, \ldots, y_k\}$, — ни что иное как алгоритм, реализующий систему функций $\{\chi_{y_1,\rho}, \ldots, \chi_{y_k,\rho}\}$. Следовательно, управляющая система, моделирующая алгоритм, решающий ЗИП $I = \langle X, V, \rho \rangle$, должна представлять собой многополюсник, реализующий множество характеристических функций $\{\chi_{y_1,\rho}, \ldots, \chi_{y_k,\rho}\}$.

Рассмотрим случай когда множество запросов $X = B^n$ — n-мерный единичный куб. Тогда каждая из функций $\chi_{y_i,\rho}$ будет функцией алгебры логики и, значит, в классе контактных схем [19] можно построить многополюсник, реализующий множество функций $\{\chi_{y_1,\rho}, \ldots, \chi_{y_k,\rho}\}$ как функций проводимости.

Если множество запросов не является n-мерным единичным кубом, то можно вместо множества $\{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$ использовать для нагрузки ребер некоторое множество F предикатов, определенных на множестве запросов X. Тогда при удачном выборе множества F и правильной нагрузке ребер много-

полюсника предикатами из F мы можем в качестве функций проводимости между полюсами получить характеристические функции $\chi_{y_i,\rho}$ $(i = \overline{1,k})$. Но если вводить сложность полученной многополюсной сети так же как в контактных схемах (то есть как количество ребер сети), то эта сложность скорее будет характеризовать объем памяти, соответствующего алгоритма, но не время поиска. Чтобы сложность сети характеризовала время поиска, ее надо вводить на подобии того, как это делалось в АДВ-модели. Для этого будем считать, что сеть у нас ориентированная, что в сети есть такой полюс, который называется *корнем*, и каждая из характеристических функций записей $\chi_{y_i,\rho}$ $(i = \overline{1,k})$ реализуется как функция проводимости между корнем и своим полюсом, который отметим приписыванием ему записи y_i. Теперь если взять произвольный запрос $x \in X$, то алгоритм функционирования сети на запросе x можно описать аналогично алгоритму разметки графа [13, 16]: считаем, что в начальный момент все вершины сети, кроме корня, неотмеченные, а некоторое упорядоченное множество вершин сети, которое назовем *множеством активных вершин*, содержит только корень сети. На каждом очередном шаге делаем следующее. Если множество активных вершин не пусто, то выбираем первую активную вершину и удаляем ее из множества активных вершин. Если выбранная вершина является полюсом, то соответствующую ей запись включаем в ответ на запрос x. Просматриваем в некотором порядке все ребра, исходящие из выбранной вершины, и если предикат, приписанный просматриваемому ребру, на запросе x принимает значение 1 и конец просматриваемого ребра неотмеченная вершина, то отмечаем конец просматриваемого ребра и включаем его в множество активных вершин (если мы включим его в начало множества активных вершин, то получим алгоритм обхода "сначала вглубь", а если в конец — то "сначала вширь"). Алгоритм завершает работу в тот момент, когда множество активных вершин окажется пустым.

Легко видеть, что если между корнем сети и некоторой вершиной на запросе x есть проводимость, то есть из корня в эту вершину ведет некоторая цепочка ребер, каждому из которых приписан предикат, принимающий значение 1 на запросе x, то

обязательно в результате описанного алгоритма обхода сети данная вершина окажется отмеченной, то есть алгоритм обхода посетит ее.

Описанный алгоритм обхода сети можно считать соответствующим сети алгоритмом поиска на запросе x, а сложность сети на запросе x можно считать равной, как и в АДВ-модели, числу просмотренных ребер, или, что то же самое числу вычисленных алгоритмом обхода предикатов. Такая сложность будет характеризовать время работы соответствующего алгоритма поиска на запросе x.

Описанную сеть с описанным функционированием будем называть *предикатным информационным графом.*

Предикатный информационный граф можно рассматривать как обобщение контактных схем и АДВ-модели, причем от контактных схем заимствована структура в виде сети с нагрузкой ребер функциями и использование функций проводимости, а от АДВ-модели — способ введения сложности сети. Кроме того вводится алгоритм обхода сети, похожий на алгоритм разметки графа [13, 16], который можно рассматривать как соответствующий сети алгоритм поиска. Причем в каждой вершине сети, до которой на некотором запросе x дошел алгоритм поиска, каждое из ребер, исходящее из этой вершины, задает возможное направление поиска, и если на этом запросе x значение предиката, соответствующего ребру, равно 1, то в направлении данного ребра поиск продолжается, а если равно 0, то по этому направлению поиск прекращается. Таким образом, из одной вершины сети может возникнуть сразу несколько направлений, по которым поиск продолжается. Но в задачах поиска часто встречается ситуация, когда из нескольких направлений поиска выбирается точно одно, например, так как это происходит в вершине ветвления алгебраического дерева вычислений, когда в зависимости от результата сравнения выбирается одно из двух направлений движения. В таких случаях гораздо удобнее приписать вершине функцию-переключатель и в зависимости от ее значения на запросе выбрать то направление, на которое указывает значение переключателя. Поэтому слегка усложним понятие предикатного информационного графа. Для этого введем множество G переключателей, то есть функций, определенных

на множестве запросов X и принимающих значения из конечного подмножества натурального ряда. Теперь если дана ненагруженная многополюсная ориентированная сеть, то нагрузку сети функциями будем осуществлять следующим образом. Сначала выберем в сети некоторое количество вершин, которые назовем *точками переключения*, и каждой точке переключения припишем некоторый переключатель из G и затем занумеруем подряд, начиная с 1, ребра, исходящие из нее. Такие ребра, исходящие из точек переключения, назовем *переключательными*. Остальные ребра назовем *предикатными* и нагрузим их предикатами из множества F. Нагрузим все полюса сети, кроме корня, записями, причем можем считать, что одна и та же запись может быть приписана нескольким полюсам. Алгоритм обхода сети на запросе в этом случае полностью аналогичен описанному ранее за тем исключением, что если активная выбранная вершина является точкой переключения, то вычисляем значение переключателя, соответствующего вершине, на запросе x, и если среди ребер, исходящих из этой вершины, есть ребро с номером, равным вычисленному значению, то включаем конец этого ребра в множество активных вершин, если только этот конец не был отмеченной вершиной. Полученную нагруженную сеть с описанным функционированием будем называть *информационным графом* (ИГ).

Прежде чем дать строгое формальное определение ИГ, введем некоторые вспомогательные понятия и обозначения.

Пусть M — некоторое конечное множество. Через $|M|$ обозначим число элементов во множестве M, называемое *мощностью множества* M.

Через $\{\overline{1,m}\}$ договоримся обозначать множество $\{1,2,\ldots,m\}$.

Некоторые оценки мы будем приводить с точностью до главного члена, поэтому введем обозначения, обычно принятые при описании асимптотических оценок.

Будем писать $\alpha(n)=\bar{o}(1)$, если $\lim\limits_{n\to\infty}\alpha(n)=0$; $A(n)=\bar{o}(B(n))$, если $A(n)=B(n)\cdot\bar{o}(1)$. Скажем, что $A(n)$ *асимптотически не превосходит* $B(n)$ при $n\to\infty$ и обозначим $A\lesssim B$, если существует $\alpha(n)=\bar{o}(1)$ такое, что начиная с некоторого номера n_0, $A(n)\le(1+\alpha(n))\cdot B(n)$. Если $A\lesssim B$ и $B\lesssim A$, то будем говорить, что A и B *асимптотически равны* при $n\to\infty$ и обо-

значать $A \sim B$. Будем писать $A \precsim B$, если существует такая положительная константа c, что, начиная с некоторого номера n_0, $A(n) \leq c \cdot B(n)$. Если $A \precsim B$ и $B \precsim A$, то будем говорить, что A и B *равны по порядку* при $n \to \infty$ и обозначать $A \asymp B$ или $A = \underline{O}(B)$.

Через $\binom{n}{k}$ будем обозначать *число сочетаний из n элементов по k*. Если r — действительное число, то через $[r]$ будем обозначать максимальное целое, не превышающее r, а через $]r[$ — минимальное целое, не меньшее, чем r. Значок $\stackrel{\text{def}}{=}$ будем понимать как "по определению равно". Математическое ожидание будем обозначать значком $\mathbf{M}$, а значок $\mathbf{M}_x$ будем понимать как среднее значение при вариации переменной x.

Договоримся также о теоретико-графовой терминологии.

Пусть нам дан ориентированный граф. В ориентированном ребре (α, β) вершину α будем называть *началом ребра*, а β — *концом*. Скажем, что ориентированное ребро графа *исходит из вершины* β (*входит в вершину* β), если β — начало (конец) данного ребра. Скажем, что ребро *инцидентно* вершине, если эта вершина является одним из концов данного ребра. *Полустепенью исхода* (*захода*) вершины графа назовем число ребер, исходящих из данной вершины (входящих в данную вершину). *Степенью инцидентности вершины* назовем число инцидентных ей ребер. Вершину графа назовем *концевой*, если полустепень ее исхода равна 0. Остальные вершины графа назовем *внутренними*.

Последовательность ориентированных ребер графа

$$(\alpha_1, \alpha_2), (\alpha_2, \alpha_3), \ldots, (\alpha_{m-1}, \alpha_m)$$

назовем *ориентированной цепью* от вершины α_1 к вершине α_m.

Если f — одноместный предикат, определенный на X , то есть $f : X \to \{0, 1\}$, то множество $N_f = \{x \in X : f(x) = 1\}$ назовем *характеристическим множеством предиката* f.

Множество $O(y, \rho) = \{x \in X : x \rho y\}$ назовем *тенью записи* $y \in Y$.

Функцию $\chi_{y,\rho} : X \to \{0, 1\}$ такую, что $N_{\chi_{y,\rho}} = O(y, \rho)$, назовем *характеристической функцией записи* y.

В формальном определении понятия ИГ используются 4 множества:

- множество запросов X;
- множество записей Y;
- *множество F одноместных предикатов*, заданных на множестве X;
- *множество G одноместных переключателей*, заданных на множестве X (*переключатели* — это функции, область значений которых является начальным отрезком натурального ряда).

Пару $\mathcal{F} = \langle F, G \rangle$ будем называть *базовым множеством*.

Определение понятия ИГ разбивается на два шага. На первом шаге раскрывается структурная (схемная) часть этого понятия, на втором — функциональная.

Определение ИГ с точки зрения его структуры.

Пусть нам дана ориентированная многополюсная сеть.

Выделим в ней один полюс и назовем его *корнем*, а остальные полюса назовем *листьями*.

Выделим в сети некоторые вершины и назовем их *точками переключения* (полюса могут быть точками переключения).

Если β — вершина сети, то через ψ_β обозначим *полустепень исхода* вершины β.

Каждой точке переключения β сопоставим некий символ из G. Это соответствие назовем *нагрузкой точек переключения*.

Для каждой точки переключения β ребрам, из нее исходящим, поставим во взаимно однозначное соответствие числа из множества $\{\overline{1, \psi_\beta}\}$. Эти ребра назовем *переключательными*, а это соответствие — *нагрузкой переключательных ребер*.

Ребра, не являющиеся переключательными, назовем *предикатными*.

Каждому предикатному ребру сети сопоставим некоторый символ из множества F. Это соответствие назовем *нагрузкой предикатных ребер*.

Сопоставим каждому листу сети некоторую запись из множества Y. Это соответствие назовем *нагрузкой листьев*.

Полученную нагруженную сеть назовем *информационным графом* над базовым множеством $\mathcal{F} = \langle F, G \rangle$.

Определение функционирования ИГ.

Скажем, что предикатное ребро проводит запрос $x \in X$, если предикат, приписанный этому ребру, принимает значение 1 на запросе x; переключательное ребро, которому приписан номер n, проводит запрос $x \in X$, если переключатель, приписанный началу этого ребра, принимает значение n на запросе x; ориентированная цепочка ребер проводит запрос $x \in X$, если каждое ребро цепочки проводит запрос x; запрос $x \in X$ проходит в вершину β ИГ, если существует ориентированная цепочка, ведущая из корня в вершину β, которая проводит запрос x; запись y, приписанная листу α, попадает в ответ ИГ на запрос $x \in X$, если запрос x проходит в лист α. *Ответом ИГ U на запрос* x назовем множество записей, попавших в ответ ИГ на запрос x, и обозначим его $\mathcal{J}_U(x)$. Эту функцию $\mathcal{J}_U(x) : X \to 2^Y$ будем считать результатом функционирования ИГ U и называть *функцией ответа ИГ U*.

Понятие ИГ полностью определено.

Проиллюстрируем приведенное определение на примере одномерной задачи интервального поиска. В этом случае 4 множества, определяющие ИГ, имеют вид:

- множество запросов $X_{int} = \{(u, v) : 0 \le u \le v \le 1\}$;
- множество записей $Y_{int} = [0, 1]$;
- множество предикатов
  $F = F_1 \bigcup F_2$, где $F_1 = \{f^1_{\le,a} : a \in [0,1]\}$, $F_2 = \{f^2_{\ge,a} : a \in [0,1]\}$,
  $$f^1_{\le,a}(u,v) = \begin{cases} 1, & \text{если } u \le a \\ 0, & \text{если } u > a \end{cases}, \qquad (1.2)$$
  $$f^2_{\ge,a}(u,v) = \begin{cases} 1, & \text{если } v \ge a \\ 0, & \text{если } v < a \end{cases}, \qquad (1.3)$$
- множество переключателей
  $G = G_1 \bigcup G_2 \bigcup G_3$, где $G_1 = \{g_{\cdot,m} : m \in \mathbf{N}\}$, $G_2 = \{g_{-,m} : m \in \mathbf{N}\}$, $G_3 = \{g_{\le,a} : a \in (0,1]\}$,
  $$g_{\cdot,m}(u,v) = \max(1,]u \cdot m[), \qquad (1.4)$$

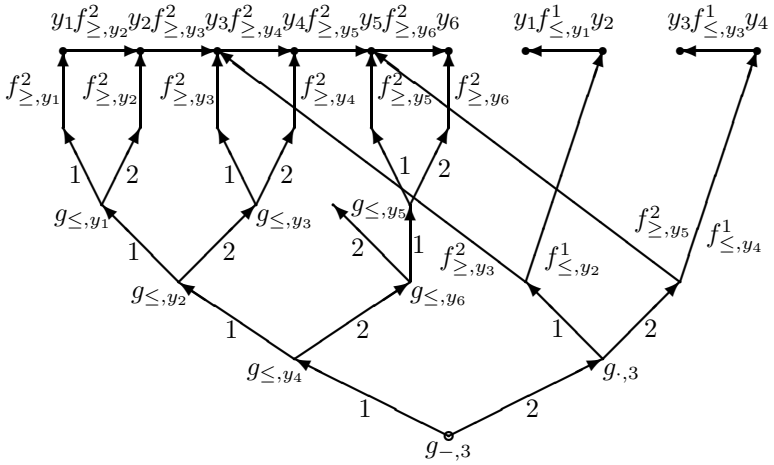


Рис. 1.1: Решение одномерной задачи интервального поиска

$$g_{-,m}(u,v) = \begin{cases} 1, & \text{если } v-u < 1/m \\ 2, & \text{если } v-u \geq 1/m \end{cases}, \qquad (1.5)$$

$$g_{\leq,a}(u,v) = \begin{cases} 1, & \text{если } u \leq a \\ 2, & \text{если } u > a \end{cases}. \qquad (1.6)$$

В информационном графе, приведенном на рисунке 1.1, корень изображен полым кружком. Листья изображены жирными точками, а записи, приписанные листьям, — это символы y с индексами. На рисунке имеется 8 переключательных вершин (им приписаны символы g с индексами) и 17 предикатных ребра (им приписаны символы f с индексами).

Если n — натуральное число, а $g(x)$ — некий переключатель, то через $\xi^n_g(x)$ обозначим предикат, определенный на X , такой, что

$$N_{\xi^n_g} = \{x \in X : g(x) = n\}.$$

Обозначим

$$\widehat{G} = \{\xi^n_g : g \in G, n \in \mathbf{N}\},$$

где $\mathbf{N}$ — множество натуральных чисел.

Если c — ребро ИГ, то через $[c]$ обозначим его нагрузку.

В соответствии с приведенными выше определениями введем функции проводимости.

Проводимостью ребра (α, β) назовем предикат, равный $[(\alpha, \beta)]$, если ребро предикатное, и $\xi_g^{[(\alpha,\beta)]}$, если ребро переключательное, где g — переключатель, соответствующий вершине α.

Проводимостью ориентированной цепи назовем конъюнкцию проводимостей ребер цепи.

Если зафиксировать запрос x, то цепь, проводимость которой на запросе x равна 1, назовем *проводящей цепью на запросе* x.

В ИГ по аналогии с контактными схемами введем для каждой пары вершин α и β *функцию проводимости* $f_{\alpha\beta}$ от вершины α к вершине β следующим образом:

- если $\alpha = \beta$, то $f_{\alpha\beta}(x) \equiv 1$ $(x \in X)$;
- если $\alpha \neq \beta$ и в ИГ не существует ориентированных цепей от α к β, то $f_{\alpha\beta}(x) \equiv 0$;
- если $\alpha \neq \beta$ и множество ориентированных цепей от α к β не пусто, то $f_{\alpha\beta}(x)$ равно дизъюнкции проводимостей всех ориентированных цепей от α к β.

Функцию проводимости от корня ИГ к некоторой вершине β ИГ назовем *функцией фильтра вершины* β и обозначим $\varphi_\beta(x)$.

Через $\mathcal{R}(U), \mathcal{P}(U), \mathcal{L}(U)$ (или просто $\mathcal{R}, \mathcal{P}, \mathcal{L}$) обозначим множества вершин, точек переключения и листьев ИГ U соответственно.

Пусть $\mathcal{N}$ — некоторая подсеть (то есть произвольное подмножество вершин и ребер) ИГ U. Через $\langle\mathcal{N}\rangle$ обозначим множество записей, соответствующих листьям этой подсети (если α — некоторый лист ИГ U, то под $\langle\alpha\rangle$ будем понимать запись, соответствующую листу α).

Легко видеть, что функция ответа ИГ U определяется соотношением

$$\mathcal{J}_U(x) = \langle\{\alpha \in \mathcal{L}(U) : \varphi_\alpha(x) = 1\}\rangle.$$

Из определения функционирования ИГ видно, что ИГ как управляющая система может рассматриваться в качестве модели алгоритма поиска, работающего над данными, организованными в структуру, определяемую структурой ИГ.

В случае, когда базовое множество переключателей G пусто, то есть в графах нет переключателей, то ИГ называются *предикатными информационными графами* (ПИГ).

ПИГ, различным листьям которого соответствуют различные записи, называется *однозначным информационным графом* (ОИГ).

ОИГ, имеющий вид дерева, листья которого совпадают с концевыми вершинами дерева, назовем *информационным деревом* (ИД).

ИД удобны и интересны тем, что структуры данных, им соответствующие, практичны и их гораздо проще реализовать на ЭВМ.

Приведем пример еще одного ИГ. Пусть $I = \langle X, V, = \rangle$ — задача поиска идентичных объектов, где на множестве $V = \{y_1, \ldots, y_7\}$ задан линейный порядок и записи упорядочены в порядке возрастания, то есть $y_1 \leq y_1 \leq \cdots \leq y_7$. Пусть множество предикатов имеет вид

$$F = \{f_{=,a}(x) = \begin{cases} 0, & \text{если } x \neq a \\ 1, & \text{если } x = a \end{cases} : a \in X\}, \tag{1.7}$$

а множество переключателей — вид

$$G = \{g_{\leq,a}(x) = \begin{cases} 1, & \text{если } x \leq a \\ 2, & \text{если } x > a \end{cases} : a \in X\}. \tag{1.8}$$

Бинарным поиском или *методом деления пополам* (см., например, [4, 15, 21]) называется алгоритм поиска в упорядоченном массиве, при котором массив делится пополам, запрос сравнивается со средней точкой и в зависимости от результата сравнения поиск рекурсивно повторяется в одной из половин.

На рисунке 1.2 приведен ИГ над базовым множеством $\mathcal{F} = \langle F, G \rangle$, решающий ЗИП I, соответствующий бинарному поиску в версии Боттенбрука [44, с. 214], в которой вопрос о равенстве записи и запроса откладывается до самого последнего момента.

Введение управляющей системы, то есть схемы с функционированием, для моделирования алгоритмов поиска позволяет использовать аппарат теории сложности управляющих систем для исследования сложности алгоритмов поиска.

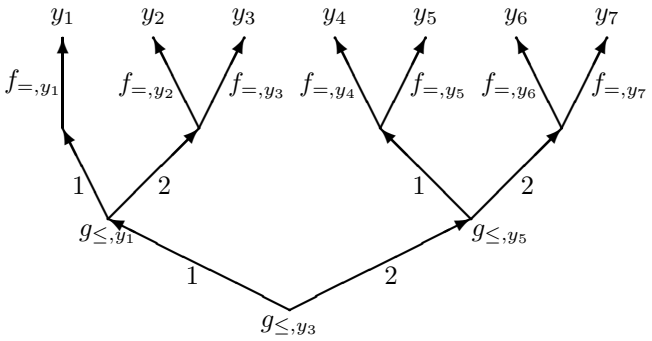


Рис. 1.2: Информационный граф бинарного поиска

Упражнения

1.1. По аналогии с одномерной задачей интервального поиска приведите тип, описывающий n-мерные задачи интервального поиска.

1.2. Опишите тип, задающий задачи интервального поиска на n-мерном булевом кубе, которые состоят в поиске в конечном подмножестве n-мерного булевого куба всех тех точек, которые попадают в подкуб, задаваемый запросом. Какова мощность множества запросов у данного типа?

1.3. Задача о метрической близости состоит в том, чтобы по произвольно взятой точке-запросу единичного n-мерного куба n-мерного евклидового пространства найти в конечном подмножестве этого куба (библиотеке) все точки, находящиеся на расстоянии не более, чем R от точки запроса. Опишите тип, задающий задачи о метрической близости.

1.4. Опишите тип, задающий задачи включающего поиска. Напомним, что в задаче включающего поиска имеется некоторое конечное множество свойств, и каждый элемент библиотеки (множества данных) обладает или не обладает каждым из этих свойств. Запрос задает некоторое подмножество множества свойств, и необходимо найти все элементы библиотеки, которые обладают всеми свойствами из запроса.

1.5. Рассмотрим следующую задачу поиска, которая может возникнуть, например, при разгадывании кроссвордов. Элементы библиотеки (записи) есть слова фиксированной длины n в алфавите $\{0,1\}$. Запрос задает некоторый набор позиций и значения букв в этих позициях. Необходимо найти в библиотеке все записи, у которых в позициях, задаваемых запросом, стоят буквы, совпадающие с соответствующими значениями позиций запроса. Опишите тип, задающий эти задачи поиска. Сравните полученный тип с типом задач интервального поиска на булевом кубе (см. упражнение 1.2).

1.2 Критерий допустимости ИГ

Пусть нам дана ЗИП $I = \langle X, V, \rho \rangle$.

Скажем, что ИГ U *решает* ЗИП $I = \langle X, V, \rho \rangle$, если для любого запроса $x \in X$ ответ на этот запрос содержит все те и только те записи из V, которые удовлетворяют запросу x, то есть

$$\mathcal{J}_U(x) = \{y \in V : x \rho y\}.$$

ИГ U, решающий ЗИП I, будем также называть *допустимым* для задачи I.

Пусть U — некоторый ИГ, y — запись из Y. Через $L_U(y)$ обозначим множество листьев ИГ U, которым соответствует запись y.

Справедлива следующая теорема [6].

Теорема 1. *ИГ U решает ЗИП $I = \langle X, V, \rho \rangle$ тогда и только тогда, когда для любой записи $y \in V$, такой, что $O(y, \rho) \neq \emptyset$, справедливо $L_U(y) \neq \emptyset$ и $\bigvee\limits_{\alpha \in L_U(y)} \varphi_\alpha = \chi_{y,\rho}$, а для любой записи $y \in V$, такой, что $O(y, \rho) = \emptyset$, справедливо либо $L_U(y) = \emptyset$, либо $\bigvee\limits_{\alpha \in L_U(y)} \varphi_\alpha \equiv 0$.*

Доказательство. **Достаточность**.

Возьмем произвольный запрос $x \in X$.

Возьмем произвольную запись $y \in V$.

Если $O(y, \rho) = \emptyset$ и, следовательно, $x \notin O(y, \rho)$, то по предположению либо $L_U(y) = \emptyset$, либо $\bigvee\limits_{\alpha \in L_U(y)} \varphi_\alpha \equiv 0$. Откуда следует,

что $y \notin \mathcal{J}_U(x)$.

Теперь рассмотрим случай, когда $O(y,\rho) \neq \emptyset$.

Если $x\rho y$, то $\chi_{y,\rho}(x) = 1$, и согласно предположению

$$\bigvee_{\alpha \in L_U(y)} \varphi_\alpha(x) = 1.$$

Откуда следует, что существует лист $\alpha \in \mathcal{L}(U)$ такой, что $\varphi_\alpha(x) = 1$. Следовательно, $y \in \mathcal{J}_U(x)$.

Если $x \notin O(y,\rho)$, то $\chi_{y,\rho}(x) = 0$, и согласно предположению $\bigvee_{\alpha \in L_U(y)} \varphi_\alpha(x) = 0$. Откуда следует, что $y \notin \mathcal{J}_U(x)$.

Таким образом, мы показали, что

$$\mathcal{J}_U(x) = \{y \in V : x\rho y\}$$

и тем самым доказали достаточность.

Необходимость.

Возьмем произвольную запись $y \in V$.

Если $O(y,\rho) = \emptyset$ и, следовательно, для любого запроса $x \in X$ $x \notin O(y,\rho)$, значит, для любого $x \in X$ $y \notin \mathcal{J}_U(x)$. Откуда следует, что либо $L_U(y) = \emptyset$, либо $\bigvee_{\alpha \in L_U(y)} \varphi_\alpha \equiv 0$.

Теперь рассмотрим случай, когда $O(y,\rho) \neq \emptyset$.

Предположим, что для данной записи y не выполняются предположения теоремы, то есть существует такой запрос x, что

$$\bigvee_{\alpha \in L_U(y)} \varphi_\alpha(x) \neq \chi_{y,\rho}(x).$$

Но это означает, что y принадлежит в точности одному из множеств $\mathcal{J}_U(x)$ или $\{y \in V : x\rho y\}$.

Следовательно, при этом x

$$\mathcal{J}_U(x) \neq \{y \in V : x\rho y\},$$

и, значит, ИГ U не решает ЗИП I.

Тем самым теорема доказана.

По сути теорема 1 говорит, что если нам дана ЗИП $I = \langle X, V, \rho\rangle$, и мы хотим построить ИГ, решающий эту ЗИП, мы должны построить многополюсник, который между корнем и остальными полюсами реализует как функции проводимости все функции $\chi_{y,\rho}$, где $y \in V$.

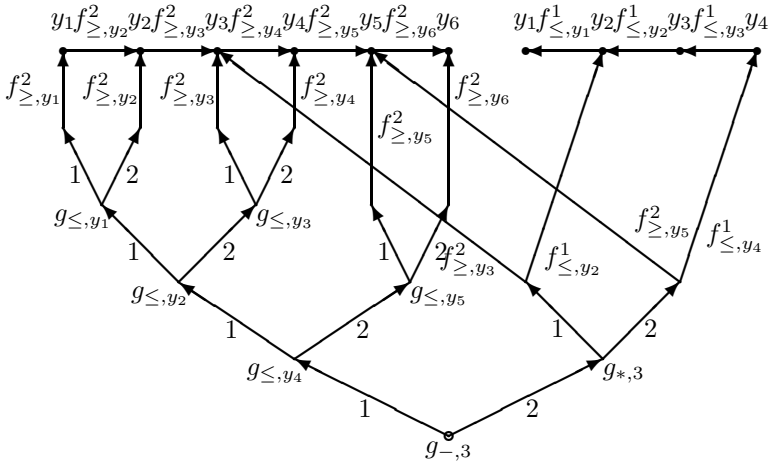


Рис. 1.3:

Упражнения

1.6. Пусть $S = \langle X, X, = \rangle$ — тип поиска идентичных объектов, множество предикатов F задается соотношением (1.7), базовое множество имеет вид $\mathcal{F} = \langle F, \emptyset \rangle$, $V = \{y_1, y_2, \ldots, y_k\} \subseteq X$. Приведите пример информационного графа над базовым множеством $\mathcal{F}$, решающего ЗИП $I = \langle X, V, = \rangle$.

1.7. Пусть $S = \langle X, X, = \rangle$ — тип поиска идентичных объектов, множество переключателей имеет вид

$$G = \{g_{=,a}(x) = \begin{cases} 1, & \text{если } x = a \\ 2, & \text{если } x \neq a \end{cases} : a \in X\}, \tag{1.9}$$

базовое множество имеет вид $\mathcal{F} = \langle \emptyset, G \rangle$, $V = \{y_1, y_2, \ldots, y_k\} \subseteq X$. Приведите пример информационного графа над базовым множеством $\mathcal{F}$, решающего ЗИП $I = \langle X, V, = \rangle$.

1.8. Пусть $X = \{1, 2, ..., N\}$, $S = \langle X, X, = \rangle$ — тип поиска идентичных объектов, множество переключателей имеет вид

$$G = \{g_a(x) = \begin{cases} 1, & \text{если } x < a \\ 2, & \text{если } x = a \\ 3, & \text{если } x > a \end{cases} : a \in X\}, \tag{1.10}$$

базовое множество имеет вид $\mathcal{F} = \langle \emptyset, G \rangle$, $V = \{3, 5, 7, 11, 13, 17, 19\}$. Постройте информационный граф над базовым множеством $\mathcal{F}$, ре-

шающий ЗИП $I = \langle X, V, = \rangle$.

1.9. Пусть $X = \{1, 2, ..., N\}$, $S = \langle X, X, = \rangle$ — тип поиска идентичных объектов, $V = \{y_1, y_2, \ldots, y_k\} \subseteq X$. Предположим, что $y_1 < y_2 < \cdots < y_k$. Метод блочного поиска с размером блока m, решающий задачу $I = \langle X, V, = \rangle$, состоит в следующем. Если на вход алгоритма поиска подается запрос $x \in X$, то, начиная с $i = 1$ до $i = k/m$, просматриваем записи $y_{i \cdot m}$. Если $x > y_{i \cdot m}$, то увеличиваем i на 1, иначе по очереди просматриваем записи $y_{(i-1)m+1}, y_{(i-1)m+2}, \ldots, y_{i \cdot m}$ и сравниваем их с запросом x. При равенстве мы нашли нужную запись, если же ни для какой записи равенства не наблюдается, то ответ на запрос x пуст. Опишите базовое множество и постройте информационный граф над этим базовым множеством, который бы решал ЗИП $I = \langle X, V, = \rangle$ методом блочного поиска.

1.10. Пусть $X = \{1, 2, ..., N\}$, $V \subseteq X$, ρ_c — отношение поиска, задаваемое на $X \times V$ и определяемое соотношением

$$x \rho_c y \Longleftrightarrow (y \in V) \& (x \leq y) \& (\neg(\exists y')((y' \in V) \& (x \leq y') \& (y' < y))), \tag{1.11}$$

т.е. $x \rho_c y$, если $y \in V$, ближайшее справа к x. При выполнении этих условий ЗИП $I = \langle X, V, \rho_c \rangle$ называется задачей о близости. Пусть базовое множество имеет вид $\mathcal{F} = \langle \emptyset, G \rangle$, где множество переключателей G задается соотношением (1.8). Постройте информационный граф над базовым множеством $\mathcal{F}$, решающий ЗИП $I = \langle X, V, \rho_c \rangle$, если $V = \{3, 5, 7, 11, 13, 17, 19\}$.

1.11. Одномерная задача о доминировании задается типом $S_{dom1} = \langle [0, 1], [0, 1], \geq \rangle$. Пусть $V = \{y_1, y_2, \ldots, y_k\} \subseteq [0, 1]$. Опишите некоторое базовое множество и постройте какой-либо информационный граф над этим базовым множеством, который бы решал ЗИП $I = \langle [0, 1], V, \geq \rangle$.

1.12. Пусть $S_{int} = \langle X_{int}, Y_{int}, \rho_{int} \rangle$ — тип одномерного интервального поиска, где отношение ρ_{int} определяется соотношением (1.1), $V = \{y_1, y_2, \ldots, y_6\}$, где $y_1 = 1/6$, $y_2 = 1/4$, $y_3 = 3/8$, $y_4 = 2/5$, $y_5 = 3/4$, $y_6 = 7/8$. Решает ли информационный граф, изображенный на рисунке 1.3, где функции определяются соотношениями (1.2)–(1.6), задачу информационного поиска $I = \langle X_{int}, V, \rho_{int} \rangle$? Обоснуйте ответ.

1.13. Докажите, что информационный граф, изображенный на рисунке 1.1, решает одномерную задачу интервального поиска $I = \langle X_{int}, V, \rho_{int} \rangle$, где $V = \{y_1, y_2, y_3, y_4, y_5, y_6\}$ — библиотека, изображенная на рисунке 1.4.

1.14. Пусть $S_{int} = \langle X_{int}, Y_{int}, \rho_{int} \rangle$ — тип одномерного интервального поиска, где отношение ρ_{int} определяется соотношением (1.1),

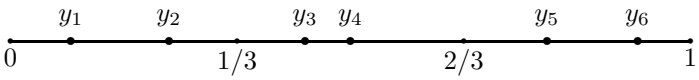


Рис. 1.4:

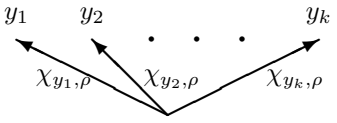


Рис. 1.5: Информационный граф переборного алгоритма

$V = \{1/8, 1/7, 1/5, 3/7,\ 3/5, 4/5, 7/8\}$. Опишите некоторое базовое множество и постройте какой-либо информационный граф над этим базовым множеством, который бы решал ЗИП $I = \langle X_{int}, V, \rho_{int}\rangle$.

1.3 Полнота для информационных графов

Если нам дана ЗИП $I = \langle X, V, \rho\rangle$ и базовое множество $\mathcal{F} = \langle F, G\rangle$, то возникает вопрос, а можно ли построить информационный граф над базовым множеством $\mathcal{F}$, решающий эту задачу I? Если для любой записи $y_i \in V = \{y_1, \ldots, y_k\}$ $\chi_{y_i,\rho} \in F$, то ответ на этот вопрос положительный, и граф, изображенный на рисунке 1.5, решает задачу I.

В данном разделе дается более полный ответ на этот вопрос.

Пусть нам дан тип $S = \langle X, Y, \rho\rangle$, где X — множество запросов, Y — множество записей, ρ — отношение поиска, заданное на $X \times Y$, и базовое множество $\mathcal{F} = \langle F, G\rangle$.

Скажем, что базовое множество $\mathcal{F}$ *полно* для типа $S = \langle X, Y, \rho\rangle$, если для любой ЗИП $I = \langle X, V, \rho\rangle$ типа S существует ИГ U над базовым множеством $\mathcal{F}$, решающий ЗИП I.

Справедлив следующий результат, относящийся к проблеме полноты для ИГ [6].

Теорема 2. *Пусть заданы множества запросов X, записей Y, отношение поиска ρ на $X \times Y$ и базовое множество $\mathcal{F} = \langle F, G \rangle$, такое, что предикат тождественная 1 принадлежит множеству F. Тогда $\mathcal{F}$ будет полным для типа $S = \langle X, Y, \rho \rangle$ тогда и только тогда, когда для любой записи $y \in Y$ такой, что $O(y, \rho) \neq \emptyset$, функцию $\chi_{y,\rho}(x)$ можно представить формулой вида*

$$\chi_{y,\rho}(x) = \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m_i} f_{ij}(x),$$

где $f_{ij} \in F \cup \widehat{G}$.

Доказательство. **Достаточность**.

Пусть нам дана произвольная ЗИП $I = \langle X, V, \rho \rangle$, где $V = \{y_1, \ldots, y_k\} \subseteq Y$. По предположению каждую из функций $\chi_{y_l,\rho}(x)$ $(l = \overline{1,k})$ можно представить формулой

$$\chi_{y_l,\rho}(x) = \bigvee_{i=1}^{n_l} \bigwedge_{j=1}^{m_{li}} f_{lij}(x),$$

где $f_{lij} \in F \cup \widehat{G},\ l = \overline{1,k}$.

Без ограничения общности можно считать, что каждая конъюнкция $\bigwedge_{j=1}^{m_{li}} f_{lij}(x)$ содержит предикат из F, причем этот предикат стоит первым в конъюнкции. В противном случае мы всегда можем добавить предикат тождественная 1.

ИГ, решающий ЗИП I, будем строить следующим образом.

Сначала возьмем $k+1$ вершину и объявим одну из них корнем, а остальные объявим листьями и мысленно перенумеруем, начиная с 1 до k. Затем для каждого $l \in \{\overline{1,k}\}$ проделаем следующее.

Припишем l-му листу (обозначим его α_l) запись y_l.

Если $O(y_l, \rho) \neq \emptyset$, то проведем из корня в лист α_l n_l ориентированных цепей, причем i-я цепь $(i = \overline{1,n_l})$ будет состоять из m_{li} ребер. Теперь для каждого i $(i \in \{\overline{1,n_l}\})$ проделаем следующее. Если $f_{lij} \in F$, то j-е ребро i-й цепи объявим предикатным и припишем ему предикат f_{lij}. Если $f_{lij} \in \widehat{G}$ (то есть $f_{lij} = \xi_g^n$, где $g \in G$, а $n \in \mathbf{N}$), то вершину β, из которой исходит j-е ребро i-й цепи, объявим переключательной припишем ей переключатель g, j-му ребру i-й цепи припишем число n, из вершины β

выпустим еще $r-1$ ребро, где r — мощность области значений переключателя g, и сопоставим им взаимно однозначно числа из множества $\{\overline{1,r}\} \setminus \{n\}$.

Нетрудно видеть, что в этом случае

$$\varphi_{\alpha_l}(x) = \bigvee_{i=1}^{n_l} \bigwedge_{j=1}^{m_{li}} f_{lij}(x) = \chi_{y_l,\rho}(x).$$

Поскольку это условие выполняется для всех листьев, то согласно теореме 1 построенный ИГ решает ЗИП I.

Необходимость.

Пусть $\mathcal{F}$ полно для отношения ρ. Возьмем произвольную запись $y \in Y$ такую, что $O(y,\rho) \neq \emptyset$. Пусть V — любая библиотека, содержащая запись y.

Так как $\mathcal{F}$ полно, то существует ИГ U, решающий ЗИП $I = \langle X, V, \rho\rangle$. Рассмотрим множество $L_U(y)$ листьев ИГ U, которым соответствует запись y. Так как U решает задачу I, то согласно теореме 1

$$\bigvee_{\alpha \in L_U(y)} \varphi_\alpha(x) \equiv \chi_{y,\rho}(x).$$

Поскольку каждая из функций $\varphi_\alpha(x)$ есть функция проводимости от корня к листу α, а функция проводимости по определению есть дизъюнкция конъюнкций некоторых предикатов из $F \cup \widehat{G}$, то необходимость доказана, что и доказывает теорему.

Упражнения

1.15. Пусть $S = \langle X, X, =\rangle$ — тип поиска идентичных объектов, базовое множество имеет вид $\mathcal{F} = \langle \emptyset, G\rangle$, где множество переключателей G задается соотношением (1.8). Будет ли полно базовое множество $\mathcal{F}$ для типа S?

1.16. Задача включающего поиска, описанная в упражнении 1.4, может быть задана типом $S_{bool} = \langle B^n, B^n, \overset{b}{\succeq}\rangle$, где B^n — n-мерный булев куб, $\overset{b}{\succeq}$ — отношение поиска на $B^n \times B^n$, определяемое следующим соотношением

$$(x_1,\ldots,x_n)\overset{b}{\succeq}(y_1,\ldots,y_n) \Longleftrightarrow x_i \geq y_i,\ i = \overline{1,n}. \qquad (1.12)$$

Приведите пример базового множества, полного для типа S_{bool}. Приведите пример минимального по мощности базового множества, полного для типа S_{bool}.

1.4 Сложность информационных графов

Из определения функционирования ИГ естественным образом вытекает, что каждому ИГ U можно сопоставить следующую процедуру поиска.

Предполагается, что эта процедура хранит в своей (внешней) памяти структуру ИГ U. Входными данными процедуры является запрос. Выходными данными является множество записей.

Пусть на вход процедуры поступил запрос x. Вводим понятие активного множества вершин и вносим в него в начальный момент корень ИГ U и помечаем его. Далее по очереди просматриваем вершины из активного множества и для каждой из них проделываем следующее:

- если рассматриваемая вершина — лист, то запись, приписанную вершине, включаем в ответ;
- если рассматриваемая вершина переключательная, то вычисляем на запросе x переключатель, соответствующий данной вершине, и если конец ребра, исходящего из рассматриваемой вершины, нагрузка которого равна значению переключателя, непомеченная вершина, то помечаем его и включаем в множество активных вершин;
- если рассматриваемая вершина предикатная, то просматриваем по очереди исходящие из нее ребра и вычисляем значения предикатов, приписанных этим ребрам, на запросе x. Концы ребер, которым соответствуют предикаты со значениями, равными 1, если они непомеченные, помечаем и включаем в множество активных вершин;
- исключаем рассматриваемую вершину из активного множества.

Процедура завершается по исчерпании активного множества.

Заметим, что если ИГ решает задачу I, то множество, полученное на выходе процедуры, будет содержать все те и только те записи библиотеки $\langle U \rangle$, которые удовлетворяют запросу x. То есть полученная процедура решает ЗИП $I = \langle X, V, \rho \rangle$, где $V = \langle U \rangle$, и, значит, является алгоритмом поиска.

Таким образом, ИГ как управляющая система может рассматриваться как модель алгоритма поиска, работающего над данными, организованными в структуру, определяемую структурой ИГ.

В данном разделе мы введем понятия сложности ИГ, которые будут характеризовать такие общепринятые меры сложности алгоритмов поиска как объем памяти, время поиска в худшем случае и время поиска в среднем.

Отметим, что в большинстве работ, посвященных исследованию сложности алгоритмов поиска, под сложностью алгоритма понимается время поиска в худшем случае [15, 22, 25, 37, 38, 39, 41, 45, 48, 55, 56], и в сравнительно редких случаях (см., например, [12, 15, 42, 47, 57, 58, 62, 65]) исследуется среднее время поиска, хотя для задач поиска, используемых в базах данных, для которых характерны массовость и многократность, исследование среднего времени поиска представляется более актуальным. Некоторое объяснение крена в сторону изучения сложности в худшем случае можно найти в цитате из [25, стр.20]: "К сожалению, анализ поведения в среднем значительно более сложная вещь, чем анализ худшего случая, по двум причинам: во-первых существенные математические трудности возникают, даже если удачно выбрано исходное распределение; во-вторых, часто с трудом достигается согласие в том, что именно выбранное распределение является реальной моделью изучаемой ситуации. Вот почему преобладающее большинство результатов связано с анализом худших случаев."

Определим понятие сложности ИГ на запросе.

Будем считать, что время вычисления любого переключателя из G примерно одинаково и характеризуется числом a, а время вычисления любого предиката из F — числом b.

Пусть нам дан некий ИГ U и произвольно взятый запрос $x \in X$.

Сложностью ИГ U на запросе x назовем число

$$T(U,x) = a \cdot \sum_{\beta \in \mathcal{P}} \varphi_\beta(x) + b \cdot \sum_{\beta \in \mathcal{R} \setminus \mathcal{P}} \psi_\beta \cdot \varphi_\beta(x).$$

Величина $T(U,x)$ характеризует время работы описанной выше процедуры поиска, сопоставленной ИГ U, поскольку $T(U,x)$ равно количеству переключателей, вычисленных данной процедурой при подаче на ее вход запроса x, умноженное на a, плюс количество вычисленных предикатов, умноженное на b.

Сложность ИГ можно вводить двумя способами. Во-первых, как максимальную сложность на запросе

$$\widehat{T}(U) = \max_{x \in X} T(U,x)$$

(здесь мы берем max, а не sup, так как $T(U,x)$ принимает целые значения, и, значит, sup всегда достигается). Эта величина характеризует время поиска в худшем случае соответствующим ИГ алгоритмом и ее будем называть *B-сложностью ИГ* (верхней сложностью). Эта величина исследуется в большинстве работ, посвященных проблемам сложности задач поиска.

Во-вторых, можно вводить сложность ИГ как среднее значение сложности ИГ на запросе, взятое по множеству всех запросов. С этой целью введем *вероятностное пространство* над множеством запросов X, под которым будем понимать тройку $\langle X, \sigma, \mathbf{P} \rangle$, где σ — некоторая *алгебра подмножеств* множества X, $\mathbf{P}$ — *вероятностная мера* на σ, то есть аддитивная мера, такая, что $\mathbf{P}(X) = 1$.

В связи с тем, что мы ввели вероятностное пространство над множеством запросов, уточним понятие типа. А именно, под *типом* будем понимать тройку $S = \langle X, Y, \rho \rangle$, считая, что множество запросов X рассматривается вместе со своим вероятностным пространством $\langle X, \sigma, \mathbf{P} \rangle$. В тех же случаях, когда мы хотим явно выделить рассматриваемое вероятностное пространство над X, мы будем представлять тип пятеркой $S = \langle X, Y, \rho, \sigma, \mathbf{P} \rangle$.

Скажем, что базовое множество $\mathcal{F} = \langle F, G \rangle$ *измеримое*, если алгебра σ содержит все множества N_f, где $f \in F \cup \widehat{G}$.

Справедлива следующая лемма.

Лемма 1. *Если базовое множество $\mathcal{F} = \langle F, G\rangle$ измеримое, то для любого ИГ U над базовым множеством $\mathcal{F}$ функция $T(U, x)$, как функция от x, является случайной величиной.*

Доказательство. Нам необходимо доказать, что для любого ИГ U над базовым множеством $\mathcal{F}$ и любого действительного числа r множество $\{x \in X : T(U, x) < r\} \in \sigma$.

Покажем, что $(\beta \in \mathcal{R}(U)) \to (N_{\varphi_\beta} \in \sigma)$.

Пусть $\beta \in \mathcal{R}(U)$. Пусть $\mathcal{C}_\beta$ — множество всех ориентированных цепей ИГ U, ведущих из корня в вершину β. Пусть C — некоторая цепь, а c — некоторое ребро. Через $\theta(c)$ обозначим проводимость ребра c.

Нетрудно видеть, что

$$\varphi_\beta = \bigvee_{C \in \mathcal{C}_\beta} \bigwedge_{c \in C} \theta(c).$$

Учитывая, что $N_{f \vee g} = N_f \cup N_g$, $N_{f \wedge g} = N_f \cap N_g$, имеем

$$N_{\varphi_\beta} = \bigcup_{C \in \mathcal{C}_\beta} \bigcap_{c \in C} N_{\theta(c)} \in \sigma.$$

Введем следующее обозначение:

$$\mathcal{M}_r = \{\mathcal{B} \subset \mathcal{R}(U) : |\{\mathcal{B} \cap \mathcal{P}\}| + \sum_{\beta \in \mathcal{B} \setminus \mathcal{P}} \psi_\beta < r\}.$$

Тогда, как нетрудно видеть,

$$\{x \in X : T(U, x) < r\} = \bigcup_{\mathcal{B} \in \mathcal{M}_r} ((\bigcap_{\beta \in \mathcal{B}} N_{\varphi_\beta}) \bigcap (\bigcap_{\beta \in \mathcal{R} \setminus \mathcal{B}} (X \setminus N_{\varphi_\beta}))) \in \sigma.$$

Тем самым лемма доказана.

Далее всюду будем предполагать, что базовое множество измеримо.

Сложностью ИГ U назовем математическое ожидание величины $T(U, x)$, то есть число

$$T(U) = \mathbf{M}_x\, T(U, x).$$

Если (β, α) — ребро ИГ, то *сложностью этого ребра* назовем число

- $b \cdot \mathbf{P}(N_{\varphi_\beta})$ — если (β, α) — предикатное ребро;
- $a \cdot \mathbf{P}(N_{\varphi_\beta})/\psi_\beta$ — если это ребро переключательное.

Если β — вершина ИГ, то число $\mathbf{P}(N_{\varphi_\beta})$ назовем *сложностью вершины* β.

Нетрудно показать, что сложность ИГ равна сумме сложностей ребер ИГ. В самом деле

$$\begin{aligned} T(U) &= \mathbf{M}_x\, T(U,x) = \int_X T(U,x)\mathbf{P}(dx) = \\ &= \int_X (b \cdot \sum_{\beta\in\mathcal{R}\setminus\mathcal{P}} \psi_\beta \cdot \varphi_\beta(x) + a \cdot \sum_{\beta\in\mathcal{P}} \varphi_\beta(x))\mathbf{P}(dx) = \\ &= b \cdot \sum_{\beta\in\mathcal{R}\setminus\mathcal{P}} \psi_\beta \int_X \varphi_\beta(x)\mathbf{P}(dx) + a \cdot \sum_{\beta\in\mathcal{P}} \int_X \varphi_\beta(x)\mathbf{P}(dx) = \\ &= b \cdot \sum_{\beta\in\mathcal{R}\setminus\mathcal{P}} \psi_\beta \mathbf{P}(N_{\varphi_\beta}) + a \cdot \sum_{\beta\in\mathcal{P}} \mathbf{P}(N_{\varphi_\beta}). \end{aligned}$$

Далее всюду будем предполагать, что $a = b = 1$.

Пусть нам дан ИГ U.

Объемом $Q(U)$ ИГ U назовем число ребер в ИГ U.

В качестве примера мы можем подсчитать сложность ИГ U, изображенного на рисунке 1.5. Легко видеть, что $Q(U) = k$ и $T(U) = k$, то есть объем графа минимально возможный, а время максимальное. Это и не удивительно, так как ИГ U соответствует переборному алгоритму поиска.

Пусть нам дана некая ЗИП I.*Сложностью задачи* I *при базовом множестве* $\mathcal{F}$ *и заданном объеме* q назовем число

$$T(I,\mathcal{F},q) = \inf\{T(U) : U \in \mathcal{U}(I,\mathcal{F}) \text{ и } Q(U) \le q\},$$

где $\mathcal{U}(I,\mathcal{F})$ — множество всех ИГ над базовым множеством $\mathcal{F}$, решающих ЗИП I.

Соответственно *B-сложностью задачи* I *при базовом множестве* $\mathcal{F}$ *и заданном объеме* q назовем число

$$\widehat{T}(I,\mathcal{F},q) = \min\{\widehat{T}(U) : U \in \mathcal{U}(I,\mathcal{F}) \text{ и } Q(U) \le q\}$$

(здесь мы берем min, а не inf, так как $\widehat{T}(U)$ принимает целые значения, и, значит, inf всегда достигается).

Число

$$T(I,\mathcal{F}) = \inf\{T(U) : U \in \mathcal{U}(I,\mathcal{F})\}$$

назовем *сложностью задачи* I *при базовом множестве* $\mathcal{F}$.

Соответственно *B-сложностью задачи I при базовом множестве $\mathcal{F}$* назовем число

$$\widehat{T}(I,\mathcal{F}) = \min\{\widehat{T}(U) : U \in \mathcal{U}(I,\mathcal{F})\}.$$

Если k — натуральное число, S — тип задач поиска, то обозначим

$$\mathcal{I}(k,S) = \{I = \langle X, V, \rho\rangle \in S : |V| = k\}.$$

Будем исследовать функции, характеризующие сложность класса ЗИП $\mathcal{I}(k,S)$, такие как функции Шеннона:

$$\widehat{\mathcal{T}}(k,S,\mathcal{F}) = \max_{I\in\mathcal{I}(k,S)} \widehat{T}(I,\mathcal{F}),$$

$$\mathcal{T}(k,S,\mathcal{F}) = \sup_{I\in\mathcal{I}(k,S)} T(I,\mathcal{F}),$$

(в первом случае мы берем max, а не sup, так как $\widehat{T}(I,\mathcal{F})$ принимает целые значения, и, значит, sup всегда достигается).

Если существует такой ИГ $U \in \mathcal{U}(I,\mathcal{F})$, что $T(U) = T(I,\mathcal{F})$, то ИГ U будем называть *оптимальным* для ЗИП I. Соответственно если $\widehat{T}(U) = \widehat{T}(I,\mathcal{F})$, то ИГ U будем называть *B-оптимальным* для ЗИП I.

Можно привести **пример такой ЗИП и такого базового множества, для которых не существует оптимального ИГ**.

Пусть $X = Y = [0,1]$ и на X задана равномерная вероятностная мера. Пусть отношение поиска есть отношение "$\geq$"для действительных чисел. Пусть библиотека состоит из одной записи $V = \{3/4\}$. Пусть базовое множество имеет вид $\mathcal{F} = \langle F, \emptyset\rangle$, где

$$F = \{f^1\} \cup \{f^2_a : a \in (0,1/2)\},$$

$$f^1(x) = \begin{cases} 0, & \text{если } x \in [0,1/2] \\ 1, & \text{если } x \in (1/2,1] \end{cases},$$

$$f^2_a(x) = \begin{cases} 0, & \text{если } x \in (a,3/4) \\ 1, & \text{если } x \in [0,a] \cup [3/4,1] \end{cases}.$$

Рассмотрим ЗИП $I = \langle X, V, \geq\rangle$. Поскольку

$$\chi_{3/4,\geq}(x) = \begin{cases} 0, & \text{если } x \in [0,3/4) \\ 1, & \text{если } x \in [3/4,1] \end{cases},$$

то $\chi_{3/4,\geq} = f^1 \& f^2_a$, для любого $a \in (0, 1/2)$. Рассмотрим ИГ U_a, состоящий из двух последовательно соединенных ребер, начало первого из которых есть корень ИГ, а конец второго — лист, которому приписана запись $3/4$, первому ребру соответствует предикат f^2_a, а второму — f^1. Нетрудно видеть, что $T(U_a) = 1 + a + 1/4 = 5/4 + a$. Очевидно, что $T(I, \mathcal{F}) = \inf\{T(U_a) : a \in (0, 1/2)\} = 5/4$, но не существует ИГ, чья сложность равна $5/4$.

Скажем, что некая вершина ИГ *достижима из корня на запросе* $x \in X$ или просто *достижимая на запросе* $x \in X$, если функция фильтра этой вершины на запросе x равна 1.

Скажем, что некая вершина ИГ *достижима из корня* или просто *достижимая*, если функция фильтра этой вершины не равна тождественному нулю, в противном случае вершину называем *недостижимой*. Скажем, что ребро ИГ *несущественное*, если выполняется хотя бы одно из следующих условий

- ребро исходит из недостижимой вершины,
- ребро является предикатным и входит в корень или в недостижимую вершину,
- ребро является предикатным и не принадлежит ни одной цепи, ведущей из корня в какой либо лист,
- ребро является переключательным, и начало этого ребра не принадлежит ни одной цепи, ведущей из корня в какой либо лист,
- ребро является переключательным, и число, приписанное этому ребру, больше максимально возможного значения переключателя, соответствующего началу этого ребра,
- начало и конец ребра совпадают.

В противном случае ребро называем *существенным*.

Легко заметить, что удаление несущественных ребер из ИГ не изменяет функционирования ИГ и не увеличивает его сложность. Поэтому всегда в дальнейшем мы будем рассматривать ИГ с точностью до несущественных ребер, а точнее будем считать, что все ребра в ИГ — существенные.

Теорема 3 (о существовании оптимальных графов). *Если множество запросов X конечно, то для любой ЗИП $I = \langle X, Y, \rho\rangle$ и любого базового множества $\mathcal{F} = \langle F, G\rangle$ такого, что $\mathcal{U}(I, \mathcal{F}) \neq \emptyset$, существует оптимальный ИГ над базовым множеством $\mathcal{F}$.*

Доказательство. Заметим, что из за конечности множества X множества F и G конечны. В самом деле, если $|X| = m$, то число предикатов, определенных на X не больше, чем 2^m. Следовательно, $|F| \leq 2^m$. Так как область значений любого переключателя есть начальный отрезок натурального ряда, то любой переключатель над X принимает не более m значений, следовательно $|G| < m^m$.

Для произвольного ИГ U обозначим через $N(U)$ подграф графа U, состоящий из ребер, имеющих ненулевую сложность.

Возьмем произвольный ИГ $U_0 \in \mathcal{U}(I, \mathcal{F})$. Обозначим $\mathcal{U}' = \{U \in \mathcal{U}(I, \mathcal{F}) : T(U) \leq T(U_0)\}$, $\mathcal{N}' = \{N(U) : U \in \mathcal{U}'\}$. Очевидно, что

$$T(I, \mathcal{F}) = \inf_{U \in \mathcal{U}(I,\mathcal{F})} T(U) = \inf_{U \in \mathcal{U}'} T(U) = \inf_{U \in \mathcal{N}'} T(U).$$

Для доказательства существования оптимального ИГ для ЗИП I нам достаточно показать конечность множества $\mathcal{N}'$.

Пусть $|F \cup \widehat{G}| = n$. Пусть M — множество, состоящее из тождественно нулевого предиката и всех предикатов, полученных из предикатов множества $F \cup \widehat{G}$ с помощью операций конъюнкции и дизъюнкции. Понятно, что $|M| \leq \min(2^m, 2^{2^n})$. Обозначим $M' = \{f \in M : \mathbf{P}(N_f) > 0\}$. Пусть $\min\limits_{f \in M'} \mathbf{P}(N_f) = r$. По определению $r > 0$. Поскольку для любого ИГ над $\mathcal{F}$ функции фильтров вершин принадлежат множеству M, то сложность любого предикатного ребра ИГ над $\mathcal{F}$ либо нулевая, либо не меньше чем r, а сложность любого переключательного ребра с ненулевой сложностью не меньше чем r/m. Отсюда в любом ИГ из множества $\mathcal{N}'$ число ребер не больше чем $T(U_0) \cdot m/r$.

Так как из конечности множеств F и G следует конечность числа различных нагрузок ИГ, то, значит, множество $\mathcal{N}'$ конечно.

Что и доказывает теорему.

Упражнения

1.17. Пусть $X = \{1, 2, ..., N\}$, $S = \langle X, X, =, \mathbf{P}, \sigma \rangle$ — тип поиска идентичных объектов, где $\sigma = 2^X$, $\mathbf{P}$ — равномерная вероятностная мера, то есть для любого $x \in X$ выполняется $\mathbf{P}(x) = 1/N$.

1. Посчитайте сложность, B-сложность и объем информационного графа, полученного при решении упражнения 1.6.
2. Посчитайте сложность, B-сложность и объем информационного графа, полученного при решении упражнения 1.7.
3. Посчитайте сложность, B-сложность и объем информационного графа, полученного при решении упражнения 1.8. Для базового множества и ЗИП, приведенных в упражнении 1.8, постройте информационный граф со сложностью, не большей, чем 1.48.
4. Посчитайте сложность, B-сложность и объем информационного графа, полученного при решении упражнения 1.9, если $N = 100$. Для какого значения параметра m сложность будет минимальна. Для какого значения параметра m B-сложность будет минимальна. Для какого значения параметра m объем будет минимальным.

1.18. Если $X = \{1, 2, ..., N\}$ и на X задана равномерная вероятностная мера, то посчитайте сложность, B-сложность и объем информационного графа, полученного при решении упражнения 1.10.

1.19. Пусть на множестве запросов $X = [0, 1]$ задана равномерная вероятностная мера. Посчитайте сложность, B-сложность и объем информационного графа, полученного при решении упражнения 1.11.

1.20. Пусть на множестве запросов $X_{int} = \{(u, v) : 0 \le u \le v \le 1\}$ задана равномерная вероятностная мера. Посчитайте сложность, B-сложность и объем информационного графа, изображенного на рисунке 1.1.

1.5 Мощностная нижняя оценка

В работе [25, стр. 92] бездоказательно, просто как очевидный факт утверждается, что время поиска по крайней мере не меньше чем время необходимое на перечисление ответа. В нашей модели этот факт находит свое доказательство и носит название

мощностной нижней оценки. В связи повсеместностью применимости мощностной нижней оценки часто при оценке времени алгоритма поиска оценивают только разность между временем поиска и временем перечисления ответа (см., например, [25]).

Пусть нам даны произвольные множества запросов X, записей Y и отношение поиска ρ на $X \times Y$. Причем на множестве запросов задано вероятностное пространство $\langle X, \sigma, \mathbf{P} \rangle$.

Скажем, что базовое множество $\mathcal{F}$ *допустимо для ЗИП* I, если существует ИГ над базовым множеством $\mathcal{F}$, который решает ЗИП I.

Следующий результат, называемый мощностной нижней оценкой, справедлив для любой ЗИП при минимальных ограничениях. Смысл этого результата заключается в том, что время поиска не может быть меньше чем время, необходимое на перечисление ответа.

Справедлива следующая теорема [6].

Теорема 4 (мощностная нижняя оценка). *Пусть* $I = \langle X, V, \rho \rangle$ *— произвольная ЗИП, такая, что существует такая запись* $y \in V$*, что* $O(y, \rho) \neq \emptyset$*,* $\mathcal{F}$ *— измеримое базовое множество, допустимое для* I*, тогда*

$$T(I, \mathcal{F}) \geq \max(1, \sum_{y \in V} \mathbf{P}(O(y, \rho))),$$

$$\widehat{T}(I, \mathcal{F}) \geq \max_{x \in X} |\mathcal{J}_I(x)|.$$

Доказательство. Возьмем произвольный ИГ U, решающий задачу I. Такой граф существует, так как $\mathcal{U}(I, \mathcal{F}) \neq \emptyset$.

Возьмем произвольный запрос $x \in X$. Так как ИГ U решает ЗИП I, то ответ на запрос x

$$\mathcal{J}_U(x) = \mathcal{J}_I(x) = \{y \in V : x \rho y\}.$$

Возьмем произвольную запись $y \in \mathcal{J}_U(x)$. Поскольку запись y попала в ответ, то, значит, в ИГ U существует некий лист α, которому приписана запись y и такой, что $\varphi_\alpha(x) = 1$. А так как $\varphi_\alpha(x) = 1$ и никакой лист не совпадает с корнем, то существует цепь, ведущая из корня в лист α, проводимость которой равна 1, и в этой цепи есть ребро, ведущее в α, с проводимостью 1. Это

ребро назовем проводящим ребром записи y. Понятно, что разным записям из $\mathcal{J}_U(x)$ соответствуют разные проводящие ребра, так как эти ребра ведут в разные листья. Если проводящее ребро записи предикатное, предикат, приписанный проводящему ребру, обязательно был вычислен перед тем, как мы попали в лист. Если проводящее ребро записи переключательное, то обязательно был вычислен переключатель, приписанный вершине, из которой исходит проводящее ребро. Причем такие переключатели для разных записей из $\mathcal{J}_U(x)$ будут разными, так как только одно из переключательных ребер, исходящих из одной вершины, может иметь проводимость, равную 1. Таким образом, каждой записи из $\mathcal{J}_U(x)$ можно сопоставить переключатель или предикат, вычисляемый непосредственно перед попаданием в соответствующий записи лист. Причем разным записям будут сопоставлены разные переключатели или предикаты. Отсюда следует, что

$$T(U,x) \geq |\mathcal{J}_I(x)|.$$

Следовательно,

$$\begin{aligned}
\widehat{T}(U) &= \max_{x\in X} T(U,x) \geq \max_{x\in X} |\mathcal{J}_I(x)|, \\
T(U) &= \mathbf{M}_x\, T(U,x) \geq \mathbf{M}_x\, |\mathcal{J}_I(x)| = \\
&= \int_X |\mathcal{J}_I(x)| \mathbf{P}(dx) = \int_X |\{y \in V : x\rho y\}| \mathbf{P}(dx) = \\
&= \sum_{y\in V} \int_{O(y,\rho)} \mathbf{P}(dx) = \sum_{y\in V} \mathbf{P}(O(y,\rho)).
\end{aligned}$$

А так как это неравенство выполняется для любого графа $U \in \mathcal{U}(I,\mathcal{F})$, то

$$\widehat{T}(I,\mathcal{F}) \geq \max_{x\in X} |\mathcal{J}_I(x)|,$$

$$T(I,\mathcal{F}) \geq \sum_{y\in V} \mathbf{P}(O(y,\rho)),$$

Так как в библиотеке V существует такая запись y, что $O(y,\rho) \neq \emptyset$, то в любом ИГ, решающем ЗИП I, существует хотя бы одна цепь, и соответственно хотя бы одно ребро исходит из корня. Следовательно, $T(I,\mathcal{F}) \geq 1$.

Тем самым теорема доказана.

Упражнения

1.21. Пусть $X = \{1, 2, ..., N\}$, $S = \langle X, X, =, \mathbf{P}, \sigma\rangle$ — тип поиска идентичных объектов, где $\sigma = 2^X$, $\mathbf{P}$ — равномерная вероятностная мера, то есть для любого $x \in X$ выполняется $\mathbf{P}(x) = 1/N$, $V = \{3, 5, 7, 11, 13, 17, 19\}$. Приведите мощностную нижнюю оценку для ЗИП $I = \langle X, V, =\rangle$.

1.22. Пусть $S_{dom1} = \langle [0, 1], [0, 1], \geq, \mathbf{P}, \sigma\rangle$ — тип одномерной задачи о доминировании, где $\mathbf{P}$ — равномерная вероятностная мера на $[0, 1]$, $V = \{y_1, y_2, \ldots, y_k\} \subseteq [0, 1]$. Приведите мощностную нижнюю оценку для ЗИП $I = \langle [0, 1], V, \geq\rangle$.

1.23. Пусть $S_{int} = \langle X_{int}, Y_{int}, \rho_{int}\rangle$ — тип одномерного интервального поиска и на множестве запросов $X_{int} = \{(u, v) : 0 \leq u \leq v \leq 1\}$ задана равномерная вероятностная мера. $V = \{y_1, y_2, \ldots, y_k\} \subseteq [0, 1]$. Приведите мощностную нижнюю оценку для ЗИП $I = \langle X_{int}, V, \rho_{int}\rangle$. Оцените сверху полученную величину.

1.6 Случай оптимальности перебора

Теорема 5. *Если $I = \langle X, V, \rho\rangle$ — ЗИП и $\mathcal{F} = \langle F, \emptyset\rangle$ — измеримое базовое множество, такие, что для любой записи $y \in V$ выполняется $\chi_{y,\rho} \in F$ и $O(y, \rho) \setminus (\bigcup\limits_{f \in F \setminus \chi_{y,\rho}} N_f) \neq \emptyset$, то*

$$\widehat{T}(I, \mathcal{F}) = T(I, \mathcal{F}) = |V|.$$

Доказательство. Пусть $V = \{y_1, y_2, \ldots, y_k\}$. Каждому $i \in \{\overline{1, k}\}$ сопоставим некоторый запрос $x_i \in O(y, \rho) \setminus (\bigcup\limits_{f \in F \setminus \chi_{y,\rho}} N_f)$. Легко видеть, что для любых таких $i, j \in \{\overline{1, k}\}$, что $i \neq j$, справедливо $x_i \neq x_j$.

Понятно, что $U \in \mathcal{U}(I, \mathcal{F}) \neq \emptyset$, так как ИГ, изображенный на рисунке 1.5, соответствующий алгоритму перебора, решает ЗИП I. Обозначим этот ИГ через U_0.

Возьмем произвольный ИГ $U \in \mathcal{U}(I, \mathcal{F})$. Так как U решает ЗИП I, то для каждого $i \in \{\overline{1, k}\}$ существуют лист α_i, которому приписана запись y_i, и цепочка ребер C_i, ведущая из корня в лист α_i, которая проводит запрос x_i. Нетрудно заметить, что

для любых таких $i, j \in \{\overline{1,k}\}$, что $i \neq j$, цепочки C_i и C_j не пересекаются по ребрам, поскольку, если бы существовало ребро, принадлежащее и C_i, и C_j, то нагрузка этого ребра принимала бы значение 1 одновременно и на x_i, и на x_j, но по определению x_i и x_j таких функций в множестве F не существует. Отсюда сразу следует, что из корня ИГ U исходит по крайней мере k ребер, и, следовательно, для любого $x \in X$ $T(U,x) \geq k$, откуда $T(U) \geq k$ и $\widehat{T}(U) \geq k$, а в силу произвольности ИГ U $T(I,\mathcal{F}) \geq k$ и $\widehat{T}(I,\mathcal{F}) \geq k$. Но так как для ИГ U_0 справедливо $T(U_0) = \widehat{T}(U_0) = k$, то $T(I,\mathcal{F}) = \widehat{T}(I,\mathcal{F}) = k$, что и требовалось доказать.

1.7 Основные задачи

В данной работе для исследования были выбраны следующие модельные классы ЗИП:

1. ЗИП, в которых вероятность появления в ответе более одной записи равна нулю, или задачи поиска с коротким ответом;

2. поиск идентичных объектов, то есть случай когда отношение поиска есть отношение равенства, и, значит, в библиотеке надо найти записи равные запросу;

3. ЗИП, когда отношение поиска является отношением частичного порядка, заданное на конечном множестве, в частности включающий поиск (это задача поиска на булевом кубе точек не больших по-компонентно, чем запрос);

4. одномерная задача интервального поиска, которая состоит в поиске в конечном подмножестве вещественной прямой всех тех точек, которые попадают в интервал-запрос.

Для перечисленных базовых задач поиска ставится проблема оптимального синтеза, которая состоит в построении для заданной задачи информационного поиска информационного графа, который решает эту задачу и имеет наименьшую или близкую к ней сложность. Причем в качестве сложности используются как время поиска в худшем случае, так и среднее время

поиска, но все же основным объектом исследования является среднее время решения базовых задач при разных заданных базовых множествах и при заданных ограничениях на объем, то есть фактически исследуется функция зависимости среднего времени поиска от объема памяти при различных ограничениях на множество доступных средств (функций из базового множества).

Предлагаемые в работе методы синтеза дают верхние оценки сложности исследуемых функций, а для подтверждения их оптимальности доказываются нижние оценки.

Глава 2

Задачи поиска с коротким ответом

Класс задач с коротким ответом — это совокупность задач поиска, в которых ответ на любой запрос содержит малое число записей. Здесь мы будем рассматривать, когда в ответе не более одной записи. Наиболее известными задачами из данного класса являются задача поиска идентичных объектов и задача поиска ближайшего объекта.

2.1 Некоторые свойства задач поиска с коротким ответом

Результат, описываемый в данном разделе, относится к классу предикатных информационных графов (ПИГ). Напомним, что ПИГ — это такие ИГ, базовое множество которых не содержит переключателей, то есть ИГ, в которых имеются только предикатные ребра.

Напомним также, что ПИГ, различным листьям которого соответствуют различные записи, называется однозначным информационным графом (ОИГ), однозначный информационный граф, имеющий вид дерева, листья которого совпадают с концевыми вершинами дерева, называется информационным деревом

(ИД).

ИД удобны и интересны тем, что структуры данных, им соответствующие, практичны и их гораздо проще реализовать на ЭВМ. Тогда как ПИГ обладают большими возможностями и охватывают более широкий класс алгоритмов. Поэтому представляет интерес выявление классов задач информационного поиска, для которых оптимальные (то есть с минимальной сложностью) ПИГ находятся в классе ИД.

Один из таких классов приводится в данном разделе. По сути это такой класс задач поиска, в которых мера множества запросов, содержащих в ответе задачи более одного элемента, равна 0.

Пусть нам даны множества запросов X, записей Y и отношение поиска ρ на $X \times Y$. Причем на множестве запросов задано вероятностное пространство $\langle X, \sigma, \mathbf{P} \rangle$.

Скажем, что ЗИП $I = \langle X, V, \rho \rangle$ *обладает A-свойством*, если

- для любой записи $y \in V$ $O(y, \rho) \in \sigma$ и $\mathbf{P}(O(y, \rho)) \neq 0$;
- для любых $y, y' \in V$, таких, что $y \neq y'$ $\mathbf{P}(O(y, \rho) \cap O(y', \rho)) = 0$.

Класс задач, обладающих A-свойством, и будет исследоваться в данном разделе.

2.2 Существование древовидного оптимального ИГ для задач поиска с коротким ответом

В этом пункте мы докажем теорему о существовании древовидного оптимального графа для задач, обладающих A-свойством.

Скажем, что *вершина α графа схемно достижима из вершины β*, если из β в α существует ориентированная цепь.

Пусть β — вершина некоторого ИГ. Обозначим через V_β множество записей, соответствующих листьям, схемно достижимым из вершины β.

Скажем, что ИГ *обладает C-свойством*, если для любой вершины β ИГ, за исключением корня $\varphi_\beta = \bigvee\limits_{y\in V_\beta} \chi_{y,\rho}$.

Пусть $I = \langle X, V, \rho\rangle$ — некоторая ЗИП, где $V = \{y_1, y_2, \ldots, y_k\}$, тогда обозначим

$$F_0^I = \{\bigvee_{j=1}^{m} \chi_{y_{i_j},\rho} : m = \overline{1,k},\ 1 \le i_1 < i_2 < \cdots < i_m \le k\}.$$

Скажем, что ИД над базовым множеством $\mathcal{F} = \langle F_0^I, \emptyset\rangle$ *обладает D_I-свойством*, если оно решает ЗИП I, обладает C-свойством и у любой вершины ИД, не являющейся полюсом, полустепень исхода больше 1.

Обозначим через $\mathcal{D}^I$ множество всех ИД, обладающих D_I-свойством.

Справедлива следующая теорема [7].

Теорема 6. *Пусть $I = \langle X, V, \rho\rangle$ — ЗИП, $\mathcal{F} = \langle F, \emptyset\rangle$ — произвольное измеримое базовое множество, допустимое для I, U — произвольный ПИГ над базовым множеством $\mathcal{F}$, решающий ЗИП I. Тогда,*

- *если I обладает A-свойством, то существует ИД $D \in \mathcal{D}^I$, такое, что $T(D) \le T(U)$,*
- *если I обладает B_1-свойством, то существует ИД $D \in \mathcal{D}^I$, такое, что $\widehat{T}(D) \le \widehat{T}(U)$.*

Доказательство. Обозначим через

$$O'(y,\rho) = O(y,\rho) \setminus (\bigcup_{\substack{y'\in V\\ y'\ne y}} O(y',\rho)).$$

Понятно, что если I обладает B_1-свойством, то $O'(y,\rho) = O(y,\rho)$, а если I обладает A-свойством, то $\mathbf{P}(O'(y,\rho)) = \mathbf{P}(O(y,\rho))$.

Обозначим через F_1 следующее бесконечное множество предикатов

$$F_1 = \{f_A : N_{f_A} = A, A \in \sigma\}.$$

Отметим, что так как $\mathcal{F}$ измеримо, то $F \subseteq F_1$. Если I обладает A-свойством, то поскольку σ — алгебра, то $F_0^I \subseteq F_1$.

Скажем, что ПИГ над базовым множеством $\langle F_1, \emptyset \rangle$ *обладает E-свойством*, если он решает ЗИП I и для любого листа ПИГ полустепень исхода этого листа равна 0.

Покажем, что существует ОИГ U_0, обладающий E-свойством, такой, что $T(U_0) \leq T(U)$.

Пусть $C = (\alpha_1, \alpha_2)(\alpha_2, \alpha_3) \cdots (\alpha_{r-1}, \alpha_r)$ — цепь в ПИГ, где α_1 — корень ПИГ. Множество ребер, исходящих из вершин $\alpha_1, \alpha_2, \ldots, \alpha_{r-1}$, назовем *следом цепи* C, а множество ребер, исходящих из вершин $\alpha_2, \ldots, \alpha_{r-1}$ — *усеченным следом цепи* C. Соответственно число $n = \sum_{i=2}^{r-1} \psi_{\alpha_i}$ будет *мощностью усеченного следа цепи* C.

Пусть $V = \{y_1, y_2, \ldots, y_k\}$.

Рассмотрим сначала запись y_1.

Обозначим через $\mathcal{C}_{y_1} = \{C_1, C_2, \ldots, C_m\}$ — множество цепей, ведущих из корня в листья множества $L_U(y_1)$. Пусть $f_1, \ldots, f_m$ функции проводимости цепей $C_1, \ldots, C_m$ соответственно. Так как U решает I, то согласно теореме 1 $\bigvee_{i=1}^{m} f_i = \chi_{y_i,\rho}$, или $\bigcup_{i=1}^{m} N_{f_i} = O(y_i, \rho)$.

Выберем в $\mathcal{C}_{y_1}$ подмножество цепей, такое, что характеристические множества их функций проводимости образуют тупиковое покрытие $O'(y_1, \rho)$. Без ограничения общности можно считать, что это первые s цепей ($s \leq m$), то есть $\bigcup_{i=1}^{s} N_{f_i} \supseteq O'(y_1, \rho)$, но удаление любого множества из объединения в левой части нарушает данное соотношение.

Пусть $N'_i \subseteq N_{f_i} (i = \overline{1,s})$, такие, что $N'_i \cap N'_j = \emptyset$, если $i \neq j$ $(i, j \in \{\overline{1,s}\})$ и $\bigcup_{i=1}^{s} N'_i = O'(y_1, \rho)$.

Понятно, что такие N'_i $(i = \overline{1,s})$ можно подобрать.

Обозначим через n_i мощность усеченного следа цепи C_i $(i = \overline{1,m})$.

Пусть c ребро ПИГ, через $[c]$ будем обозначать его нагрузку, то есть предикат, приписанный этому ребру.

Для каждого ребра c графа U заменим его нагрузку $[c]$ на

$f_{N_{[c]}\setminus O'(y_1,\rho)}$. Так как $N_{[c]} \in \sigma$ и $O'(y_1,\rho) \in \sigma$, то $f_{N_{[c]}\setminus O'(y_1,\rho)} \in F_1$.

После такой операции функции фильтра листьев, не принадлежащих $L_U(y_1)$, не изменятся, а дизъюнкция функций фильтра листьев из $L_U(y_1)$ станет равной $f_{O(y_1,\rho)\setminus O'(y_1,\rho)}$, при этом сложность графа уменьшится по крайней мере на $\sum_{i=1}^{s} n_i \cdot \mathbf{P}(N_i')$.

Пусть $n_j = \min\limits_{1\le i\le s} n_i$.

Пусть цепь C_j ведет в некоторый лист $\alpha_1 \in L_U(y_1)$. Все листья из $L_U(y_1)$, отличные от α_1, объявим обычными вершинами и уберем приписанную им нагрузку y_1. После этой операции множество $L_U(y_1)$ будет состоять только из одного листа α_1.

Для каждого ребра c цепи C_j заменим его нагрузку $[c]$ на $[c]\vee \chi_{y_1,\rho}$. После этой операции функция фильтра листа α_1 станет равной $\varphi_{\alpha_1} = \chi_{y_1,\rho}$. Таким образом, полученный граф снова решает задачу I.

В результате последней операции сложность графа увеличится на $n_j \cdot \mathbf{P}(O(y_1,\rho))$.

Заметим, что

$$\sum_{i=1}^{s} n_i \cdot \mathbf{P}(N_i') \ge n_j \cdot \sum_{i=1}^{s} \mathbf{P}(N_i') = n_j \cdot \mathbf{P}(O'(y_1,\rho)) = n_j \cdot \mathbf{P}(O(y_1,\rho)).$$

Отсюда следует, что полученный граф по сложности не больше чем $T(U)$.

Переобозначим цепь C_j на C_1'.

Проделаем выше описанную процедуру для всех остальных записей $y_i,\ i = \overline{2,k}$, и для каждой записи y_i получим цепь C_i', ведущую из корня в некоторый лист α_i (причем α_i будет единственным листом в $L_U(y_i)$) и имеющую проводимость $\chi_{y_i,\rho}$.

Удалим из полученного графа все ребра, не принадлежащие ни одной из цепей $C_1', \ldots, C_k'$. Граф U', получающийся после этого удаления, будет решать задачу I и иметь сложность, не превышающую $T(U)$.

Граф U' является ОИГ, так как каждой записи соответствует ровно один лист. Покажем, что в графе U' полустепень исхода любого листа равна 0.

Предположим, что это не так. Так как граф U' состоит только из ребер, принадлежащих цепям $C'_1, \ldots, C'_k$, то, значит, некоторая цепь C'_j ($j \in \{\overline{1,k}\}$) проходит через некоторый лист α_m, где
$m \in \{\overline{1,k}\}$ и $m \neq j$. Так как граф U' решает ЗИП I, то $\varphi_{\alpha_m} = \chi_{y_m,\rho}$. Следовательно, проводимость f_j цепи C'_j такая, что $N_{f_j} \subseteq O(y_m, \rho)$. Но этого не может быть, так как $N_{f_j} = O(y_j, \rho)$ и $\mathbf{P}(O(y_j, \rho) \cap O(y_m, \rho)) = 0$, и $\mathbf{P}(O(y_j, \rho)) \neq 0$.

Таким образом, мы доказали, что U' обладает E-свойством и, значит, его можно взять в качестве искомого графа U_0.

Теперь в графе U_0 изменим нагрузку всех ребер следующим образом. Пусть c произвольное ребро графа, такое, что оно принадлежит цепям $C'_{i_1}, \ldots, C'_{i_m}$, тогда в качестве нагрузки этого ребра возьмем $\bigvee_{j=1}^{m} \chi_{y_{i_j},\rho} \in F_0^I$. В частности нагрузка ребра, ведущего в некоторый лист α_i, будет равна $\chi_{y_i,\rho}$.

Поскольку эта замена не меняет проводимости ни одной из цепей $C'_1, \ldots, C'_k$, то функционирование графа не меняется.

Граф, полученный после осуществления такой замены нагрузки для всех ребер, обозначим через U_1. $T(U_1) \leq T(U_0)$, так как в графе U_0 для каждого ребра, принадлежащего C'_i ($i \in \{\overline{1,k}\}$), конъюнкция его нагрузки и функции $\chi_{y_i,\rho}$ равна $\chi_{y_i,\rho}$.

ОИГ U_1 является графом над F_0^I, $T(U_1) \leq T(U)$, U_1 обладает E-свойством, причем в каждый лист графа U_1 ведет единственное ребро, и еще, если β вершина графа U_1, отличная от корня, через которую проходят некоторые цепи $C'_{i_1}, \ldots, C'_{i_s}$, то $\varphi_\beta = \bigvee_{j=1}^{s} \chi_{y_{i_j},\rho}$. Причем так как через эту вершину β не проходит других цепей, ведущих в листья, то $V_\beta = \{y_{i_1}, \ldots, y_{i_s}\}$, откуда следует, что ОИГ U_1 обладает C-свойством.

Предположим, что в U_1 есть вершины с полустепенью захода более 1. Пусть этих вершин m штук.

Рассмотрим произвольную вершину β с полустепенью захода, превышающей 1.

Пусть через нее проходит s цепей $C'_{i_1}, \ldots, C'_{i_s}$. Согласно за-

мечанию $\varphi_\beta = \bigvee_{j=1}^{s} \chi_{y_{i_j},\rho}$.

Отметим, что ребра, входящие в вершину β не принадлежат никаким другим цепям, кроме цепей $C'_{i_1}, \ldots, C'_{i_s}$.

Обозначим через C''_{i_j} и C'''_{i_j} части цепи C'_{i_j} ($j \in \{\overline{1,s}\}$ соответственно от корня до вершины β и от вершины β до листа α_{i_j}, а через n'_{i_j} — мощность усеченного следа цепи C''_{i_j}. Обозначим через G_β подграф графа U_1, состоящий из цепей $C''_{i_1}, \ldots, C''_{i_s}$, а через $O_\beta = \bigcup_{j=1}^{s} O'(y_{i_j}, \rho)$.

Для каждого ребра c подграфа G_β заменим его нагрузку $[c]$ на $f_{N_{[c]} \setminus O_\beta}$. Не трудно заметить, что после этой операции проводимости цепей C'_{i_m}, таких, что $m \notin \{\overline{1,s}\}$, не изменится, более того нагрузка ребер из этих цепей, как и прежде, будет принадлежать F_0^I. При этом сложность графа уменьшится по крайней мере на $\sum_{i=1}^{s} n'_{i_j} \cdot \mathbf{P}(O'(y_{i_j}, \rho))$.

Пусть $n'_{i_l} = \min\limits_{1 \le j \le s} n'_{i_j}$.

Для каждого ребра c цепи C''_{i_l} ($j = \overline{1,s}$) заменим его нагрузку $[c]$ на $[c] \vee \bigvee_{j=1}^{s} \chi_{y_{i_j},\rho}$. Эта замена увеличит сложность графа на $n'_{i_j} \cdot \mathbf{P}(O(y_{i_j}, \rho))$, но поскольку это не больше чем $\sum_{j=1}^{s} n'_{i_j} \cdot \mathbf{P}(O'(y_{i_j}, \rho))$, то полученный граф по сложности не превышает $T(U_1)$.

Объявим цепью C'_{i_j} цепь, составленную из C''_{i_l} и C'''_{i_j} ($j = \overline{1,s}$). Нетрудно заметить, что проводимость новообъявленных цепей C'_{i_j} по-прежнему равна $\chi_{y_{i_j},\rho}$.

Теперь удалим все ребра, не принадлежащие ни одной из цепей C'_j ($j = \overline{1,k}$). В частности мы удалим все ребра, входящие в вершину β, кроме ребра, принадлежащего C''_{i_l}, в силу сделанного выше замечания о ребрах, входящих в β.

Полученный таким образом граф обозначим U_2. Мы получили, что $T(U_2) \le T(U_1)$, U_2 решает задачу I, и число вершин с

полустепенью захода более 1 в U_2 по крайней мере на 1 меньше чем в U_1, поскольку теперь в вершину β ведет единственное ребро, а новых вершин с полустепенью захода более 1 образоваться не могло.

Нетрудно заметить, что $\widehat{T}(U_2) \leq \widehat{T}(U_1)$. В самом деле, для любого $x \in X \setminus O_\beta$ $T(U_2, x) = T(U_1, x)$, а для любого $j \in \{\overline{1,s}\}$ и для любого $x \in O(y_{i_j}, \rho)$ $T(U_2, x) = T(U_1, x) - n'_{i_j} + n'_{i_l} \leq T(U_1, x)$.

Применяя вышеописанную процедуру к графу U_2, мы получим граф U_3 с еще меньшим числом вершин с полустепенью захода более 1.

Применив данную процедуру нужное количество раз, мы получим некий граф U_r ($r \leq m+1$), в котором полустепень захода любой вершины равна 1.

Поскольку плюс к этому полустепень исхода любого листа U_r равна 0, то U_r является информационным деревом. По построению для любой некорневой вершины графа U_r выполняется условие $\varphi_\beta = \bigvee\limits_{y \in V_\beta} \chi_{y,\rho}$. Предположим, что в U_r есть неполюсная вершина β, полустепень исхода которой равна 1. Тогда нагрузка ребер, входящего в β и исходящего из β, одинакова. Следовательно, ребро, исходящее из β, можно удалить без ущерба для функционирования и сложности. Эту операцию повторим для каждой неполюсной вершины, полустепень исхода которой равна 1. После этого U_r будет ИД, обладающим D_I-свойством. Поскольку $T(U_r) \leq T(U)$ и $\widehat{T}(U_r) \leq \widehat{T}(U)$ по построению, то U_r можно взять в качестве искомого дерева D.

Тем самым теорема доказана.

Следствие 1. *Если ЗИП I обладает A-свойством $\mathcal{F} = \langle F, \emptyset \rangle$ — измеримое базовое множество, допустимое для I, такое, что $F_0^I \subseteq F$, то существует оптимальный ПИГ U для ЗИП I, принадлежащий классу $\mathcal{D}^I$.*

Доказательство. Так как $F_0^I \subseteq F$, то $\mathcal{D}^I \subseteq \mathcal{U}(I, \mathcal{F})$, то есть оптимальные ИГ, если они существуют, надо искать согласно теореме 6 в классе $\mathcal{D}^I$. Чтобы показать существование оптимального ИГ, достаточно заметить, что $\mathcal{D}^I$ — конечное множество. В самом деле, ИГ из $\mathcal{D}^I$ — это деревья с k концевыми вершинами (здесь k — количество записей в библиотеке задачи

I), нагрузка ребер которых берется из конечного множества F_0^I, значит, $\mathcal{D}^I$ — конечное множество.

Тем самым следствие доказано.

2.3 Нижняя оценка сложности ИГ для задач поиска с коротким ответом и равномощными тенями записей

Результат теоремы 6 помимо того, что представляет самостоятельный интерес, позволяет получить нижнюю оценку сложности информационных графов для задач поиска с равномощными слабо пересекающимися тенями записей, причем нижнюю оценку значительно лучшую, чем мощностная нижняя оценка.

Дадим строгое определение исследуемого в данном разделе класса задач поиска.

Скажем, что ЗИП $I = \langle X, V, \rho \rangle$ *обладает F-свойством*, если она обладает A-свойством и для любых $y, y' \in V$ справедливо $\mathbf{P}(O(y, \rho)) = \mathbf{P}(O(y', \rho))$.

Обозначим

$$R(k) = 3k[\log_3 k] + 4(k - 3^{[\log_3 k]}) + \max(0, k - 2 \cdot 3^{[\log_3 k]}),$$

где $k \geq 1$.

Справедлива следующая теорема [7].

Теорема 7. *Если $I = \langle X, V, \rho \rangle$ — ЗИП, обладающая F-свойством, $\mathcal{F} = \langle F, \emptyset \rangle$ — измеримое базовое множество, допустимое для I, то*

$$T(I, \mathcal{F}) \geq \mathbf{P}(O(y, \rho)) \cdot R(|V|),$$

где $y \in V$.

Прежде чем доказывать теорему, докажем ряд вспомогательных утверждений.

Лемма 2. *Пусть $0 \leq t \leq t_1 \leq t_2$, $t_1 - t \geq 1$ тогда*

$$R(t_1) + R(t_2) \leq R(t_1 - t) + R(t_2 + t).$$

Доказательство. Нетрудно убедиться, что $R(k)$ есть непрерывная выпуклая функция, представляющая собой ломаную

линию, причем точки 3^l и $2 \cdot 3^l$ $(l = 0, 1, 2, \ldots)$ являются точками излома. А из свойств выпуклых функций легко выводится утверждение леммы.

Обозначим через $\mathbf{N}$ — множество натуральных чисел.

Лемма 3. *Пусть* $k \in \mathbf{N}$, $k \geq 2$, $l = [\log_3 k]$, *тогда*

$$r_1(k) \stackrel{\text{def}}{=} \min\{R(t_1) + R(t_2) : t_1 + t_2 = k; t_1, t_2 \in \mathbf{N}\} =$$
$$= \begin{cases} R(k) + 4 \cdot 3^{l-1} - 3k, & \textit{если } 3^l \leq k \leq 4 \cdot 3^{l-1} \\ R(k) - 2k, & \textit{если } 4 \cdot 3^{l-1} \leq k \leq 2 \cdot 3^l \\ R(k) - k - 2 \cdot 3^l, & \textit{если } 2 \cdot 3^l \leq k \leq 3 \cdot 3^l. \end{cases}$$

Доказательство. Из леммы 2 следует, что

$$r_1(k) = \begin{cases} 2 \cdot R(k/2), & \text{если } k \text{ — четно} \\ R((k-1)/2) + R((k+1)/2), & \text{если } k \text{ — нечетно.} \end{cases}$$

Пусть $l = [log_3 k]$, $c = k - 3^{[\log_3 k]}$, то есть $k = 3^l + c$, где $0 \leq c \leq 2 \cdot 3^l$, тогда по определению функции $R(k)$

$$R(k) = \begin{cases} 3 \cdot l \cdot k + 4 \cdot c, & \text{если } 0 \leq c \leq 3^l \\ 3 \cdot l \cdot k + 5 \cdot c - 3^l, & \text{если } 3^l \leq c \leq 2 \cdot 3^l. \end{cases}$$

Докажем утверждение леммы при четном k.

1) Пусть $0 \leq c \leq 3^{l-1}$, тогда

$$\begin{aligned} r_1(k) &= 2 \cdot R(k/2) = 2 \cdot R(3^{l-1} + (3^{l-1} + c)/2) = \\ &= 2 \cdot (3 \cdot (l-1) \cdot k/2 + 2 \cdot (3^{l-1} + c)) = \\ &= 3 \cdot l \cdot k + 4 \cdot c - 3 \cdot k + 4 \cdot 3^{l-1} = R(k) + 4 \cdot 3^{l-1} - 3 \cdot k, \end{aligned}$$

так как в этом случае $(3^{l-1} + c)/2 \leq 3^{l-1}$.

2) Пусть $3^{l-1} \leq c \leq 3^l$, тогда $3^{l-1} \leq (3^{l-1} + c)/2 \leq 2 \cdot 3^{l-1}$ и

$$\begin{aligned} r_1(k) &= 2 \cdot R(k/2) = \\ &= 2 \cdot (3 \cdot (l-1) \cdot k/2 + 5 \cdot (3^{l-1} + c)/2 - 3^{l-1}) = \\ &= 3 \cdot l \cdot k + 5 \cdot c - 3 \cdot k + 3^l = R(k) - 2 \cdot k. \end{aligned}$$

3) Пусть $3^l \leq c \leq 2 \cdot 3^l$, тогда $k/2 = 3^l + (c - 3^l)/2$, $(c - 3^l)/2 \leq 3^l$ и

$$\begin{aligned} r_1(k) &= 2 \cdot R(k/2) = 2 \cdot (3 \cdot l \cdot k/2 + 2 \cdot (c - 3^l)) = \\ &= 3 \cdot l \cdot k + 4 \cdot c - 4 \cdot 3^l = R(k) - k - 2 \cdot 3^l. \end{aligned}$$

Для нечетного k лемма доказывается аналогично.

Тем самым лемма доказана.

Следствие 2. *$r_1(k) \geq R(k) - 2 \cdot k$, причем равенство достигается только при $4 \cdot 3^{[\log_3 k]-1} \leq k \leq 2 \cdot 3^{[\log_3 k]}$.*

Лемма 4. *Пусть $k \in \mathbf{N}$, $k \geq 3$, тогда*

$$r_2(k) \stackrel{\text{def}}{=} \min\{R(t_1) + R(t_2) + R(t_3) : \sum_{i=1}^{3} t_i = k; t_1, t_2, t_3 \in \mathbf{N}\} =$$
$$= R(k) - 3 \cdot k.$$

Доказательство. Из леммы 2 следует, что

$$r_2(k) = \begin{cases} 3 \cdot R(k/3), & \text{если } k = 3s, \\ 2 \cdot R((k-1)/3) + R((k+2)/3), & \text{если } k = 3s+1, \\ R((k-2)/3) + 2 \cdot R((k+1)/3), & \text{если } k = 3s+2. \end{cases}$$

Докажем утверждение леммы при $k = 3s$.

Пусть $l = [\log_3 k]$, $c = k - 3^{[\log_3 k]}$, то есть $k = 3^l + c$, $k/3 = 3^{l-1} + c/3$.

1) Пусть $0 \leq c \leq 3^l$, тогда

$$r_2(k) = 3 \cdot R(k/3) = 3 \cdot (3 \cdot (l-1) \cdot k/3 + 4 \cdot c/3) =$$
$$= 3 \cdot l \cdot k + 4 \cdot c - 3 \cdot k = R(k) - 3 \cdot k.$$

2) Пусть $3^l \leq c \leq 2 \cdot 3^l$, тогда

$$r_2(k) = 3 \cdot R(k/3) = 3 \cdot (3 \cdot (l-1) \cdot k/3 + 5 \cdot c/3 - 3^{l-1}) = R(k) - 3 \cdot k.$$

Для случаев $k = 3s+1$ и $k = 3s+2$ лемма доказывается аналогично.

Тем самым лемма доказана.

Рассмотрим функцию $M(D) = \sum_{\beta \in \mathcal{R}(D)} |\beta| \cdot \psi_\beta$, где $D \in \mathcal{D}_0$, $\mathcal{D}_0$ — множество связных ориентированных деревьев с корнем, в которых каждое ребро принадлежит хотя бы одной цепи, ведущей из корня к какой-либо концевой вершине, $|\beta|$ — количество концевых вершин, схемно достижимых из β.

Пусть $N(k) = \min\{M(D) : D \in \mathcal{D}_0 \text{ и } |D| = k\}$, где $|D|$ — количество концевых вершин в дереве D.

Дерево $D \in \mathcal{D}_0$ назовем *простым*, если $M(D) = N(|D|)$.

Лемма 5. *Для любого натурального k существует простое дерево с k концевыми вершинами, полустепень исхода любой внутренней вершины которого равна либо 2, либо 3.*

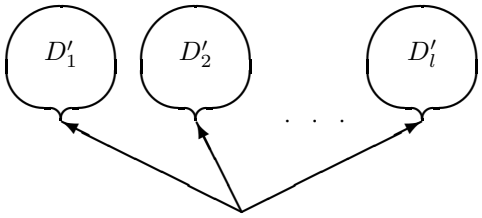


Рис. 2.1: Ветвь с большой полустепенью исхода корня

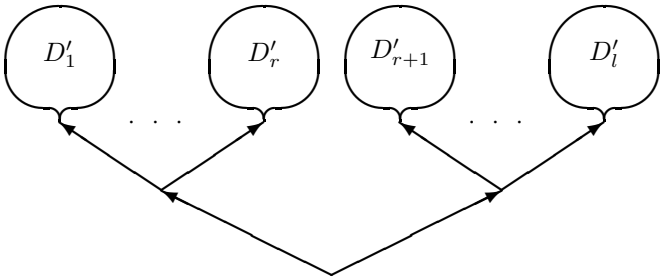


Рис. 2.2: Уменьшение полустепени исхода ветвей

Доказательство. Предположим, что дерево D содержит ветвь D_1 вида изображенного на рисунке 2.1, где $l \geq 4$.

Рассмотрим дерево D_2 вида, указанного на рисунке 2.2, где $r = [l/2]$.

Покажем, что $M(D_2) \leq M(D_1)$.

Пусть $t_i = |D'_i|$ $(i = \overline{1,l})$, $t_i \geq 1$.

Возможны два случая:

1) l — четно, то есть $l = 2 \cdot r$, где $r \geq 2$, тогда

$$\begin{aligned} M(D_2) &= 2 \cdot \sum_{i=1}^{l} t_i + r \cdot \sum_{i=1}^{l} t_i + \sum_{i=1}^{l} M(D'_i) \leq M(D_1) = \\ &= l \cdot \sum_{i=1}^{l} t_i + \sum_{i=1}^{l} M(D'_i), \end{aligned}$$

причем равенство достигается лишь при $r = 2$.

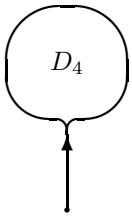


Рис. 2.3: Ветвь с единичной полустепенью исхода корня

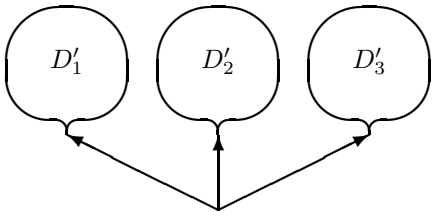


Рис. 2.4: Дерево D_1

2) l — нечетно, то есть $l = 2 \cdot r + 1$, где $r \geq 2$, тогда

$$M(D_2) = 2 \cdot \sum_{i=1}^{l} t_i + r \cdot \sum_{i=1}^{r} t_i + (r+1) \cdot \sum_{i=r+1}^{l} t_i + \sum_{i=1}^{l} M(D'_i) < M(D_1).$$

Таким образом, мы показали, что если в дереве есть ветвь вида D_1, то ее всегда можно заменить на ветвь вида D_2, и дерево от этого не усложнится.

Осталось заметить, что ветвь вида D_3, изображенного на рисунке 2.3, всегда можно заменить на ветвь D_4, не усложняя при этом дерева.

Тем самым лемма доказана.

Лемма 6. *Для любого натурального k выполняется $N(k) = R(k)$.*

Доказательство. Доказывать будем индукцией по числу k.

Базис индукции. $k = 1;\ N(1) = 0 = R(1)$.
$k = 2;\ N(2) = 4 = R(2).\quad k = 3;\ N(3) = 9 = R(3)$.

Шаг индукции. Пусть для любого $t < k$ $N(t) = R(t)$.

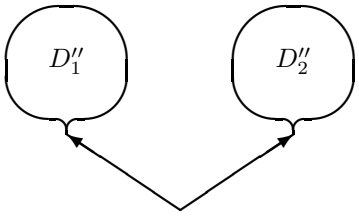


Рис. 2.5: Дерево D_2

Согласно лемме 5, существует простое дерево с k концевыми вершинами, которое имеет либо вид D_1, изображенный на рисунке 2.4, либо вид D_2, изображенный на рисунке 2.5.

Согласно предположению индукции, лемме 4 и следствию 2,

$$\begin{aligned} M(D_1) &= 3\cdot k + M(D_1') + M(D_2') + M(D_3') \geq \\ &\geq 3\cdot k + R(|D_1'|) + R(|D_2'|) + R(|D_3'|) \geq \\ &\geq 3\cdot k + \min\{R(t_1) + R(t_2) + R(t_3) : t_1 + t_2 + t_3 = k, \\ &\quad t_1, t_2, t_3 \in \mathbf{N}\} = 3\cdot k + R(k) - 3\cdot k = R(k), \end{aligned}$$

$$\begin{aligned} M(D_2) &= 2\cdot k + M(D_1'') + M(D_2'') \geq \\ &\geq 2\cdot k + R(|D_1''|) + R(|D_2''|) \geq \\ &\geq 2k + \min\{R(t_1) + R(t_2) : t_1 + t_2 = k, t_1, t_2 \in \mathbf{N}\} \geq \\ &\geq R(k). \end{aligned}$$

С другой стороны, если D_1', D_2', D_3' — простые и такие, что их мощности (то есть количество концевых вершин) различаются не более чем на 1, то $M(D_1) = R(k)$ и D_1 — простое, а если $4\cdot 3^{[\log_3 k]-1} \leq k \leq 2\cdot 3^{[\log_3 k]}$ и D_1'' и D_2'' — простые и такие, что $||D_1''| - |D_2''|| \leq 1$, то $M(D_2) = R(k)$ и D_2 — простое дерево.

Тем самым лемма доказана.

Перейдем к *доказательству теоремы* 7.

Возьмем произвольный ИГ $U \in \mathcal{U}(I, \mathcal{F})$. Согласно теореме 6 существует такое информационное дерево $D \in \mathcal{D}^I$, что

$T(U) \geq T(D)$. Оценим сложность ИД D.

$$\begin{aligned} T(D) &= \sum_{\beta\in\mathcal{R}(D)} \psi_\beta \cdot \mathbf{P}(N_{\varphi_\beta}) \geq \sum_{\beta\in\mathcal{R}(D)} \psi_\beta \cdot \mathbf{P}(N_{\vee_{y\in V_\beta}\chi_{y,\rho}}) = \\ &= \sum_{\beta\in\mathcal{R}(D)} \psi_\beta \cdot \mathbf{P}(\bigcup_{y\in V_\beta} O(y,\rho)) = \\ &= \sum_{\beta\in\mathcal{R}(D)} \psi_\beta \cdot |V_\beta| \cdot \mathbf{P}(O(y',\rho)) = \\ &= \mathbf{P}(O(y',\rho)) \cdot \sum_{\beta\in\mathcal{R}(D)} \psi_\beta \cdot |V_\beta| = \\ &= \mathbf{P}(O(y',\rho)) \cdot M(D) \geq \mathbf{P}(O(y',\rho)) \cdot N(|V|) \; = \\ &= \mathbf{P}(O(y',\rho)) \cdot R(|V|), \end{aligned}$$

где y' некоторая запись из V.

В силу произвольности U теорема доказана.

Следствие 3. *В условиях теоремы 7*

$$T(I,\mathcal{F}) \geq c \cdot [\log_3 |V|],$$

где $c \leq 3$.

Справедливость утверждения следует из того, что

$$k \cdot \mathbf{P}(O(y,\rho)) \leq 1.$$

Для сравнения отметим, что мощностная нижняя оценка сложности, получаемая с помощью теоремы 4 для задач, обладающих F-свойством, равна константе, которая не превышает 1.

Результат следствия 3 может показаться аналогичным теоретико-информационной нижней оценке [43], но он имеет принципиальные отличия: во первых, он касается средней сложности, а не сложности в худшем случае, во-вторых, доказывался для произвольных сетей, а не бинарных деревьев.

Следствие 4. *Если* $I = \langle X, V, \rho\rangle$ *— ЗИП, обладающая* F*-свойством,* $\mathcal{F} = \langle F, \emptyset\rangle$ *— измеримое базовое множество, допустимое для* I*, такое, что* $F_0^I \subseteq F$*, то*

$$\mathbf{P}(O(y,\rho)) \cdot R(k) \leq T(I,\mathcal{F}) \leq \mathbf{P}(O(y,\rho)) \cdot R(k) + 1,$$

где $k = |V|,\ y \in V$.

Доказательство. Нижняя оценка следует из теоремы 7, а верхнюю дает ИД, изображенный на рисунке 2.3, где D_4 — простое дерево с k концевыми вершинами, которые одновременно являются листьями графа, нагрузка листьев записями из V выбрана произвольно, а нагрузка ребер осуществляется по методу, описанному в доказательстве теоремы 6, так, чтобы выполнялось C-свойство.

Тем самым следствие доказано.

Глава 3

Поиск идентичных объектов

В данном разделе мы будем рассматривать задачу поиска идентичных объектов, которая в том или ином виде встречается во всех информационных системах и базах данных. Задача поиска идентичных объектов состоит в поиске в информационном массиве объекта, идентичного объекту-запросу.

Как уже говорилось, для задачи поиска идентичных объектов (как и для задачи о близости) в рамках АДВ-модели справедлива логарифмическая теоретико-информационная нижняя оценка сложности для худшего случая [43]. Поэтому считается, что бинарный поиск является оптимальным по порядку для задачи поиска идентичных объектов, и это как бы закрывает проблему, но несмотря на это имеется много работ, посвященных исследованию алгоритмов поиска идентичных объектов в худшем случае [1, 4, 35, 44, 50]. Связано это, во-первых, с проблемой поддержки сбалансированности бинарного дерева при операциях вставки и удаления, а во-вторых, с тем, что бинарный поиск хорош только тогда, когда целиком вся библиотека помещается во внутренней памяти (*внутренний поиск* [15]), если же библиотека вся или частично расположена на внешних носителях (*внешний поиск* [15]), то эффективность бинарного поиска сразу падает. Также много работ, посвященных изучению ал-

горитмов поиска идентичных объектов, имеющих хорошие временные характеристики в среднем [12, 47, 53, 57, 58, 62, 65]. Связаны они в основном с методом хеширования, на котором мы остановимся чуть подробнее.

Хеширование предполагает наличие хеш-функции. Хеш-функция $h(y)$ определена на множестве записей X и переводит его в множество $\{\overline{1,m}\}$, где m — параметр хеш-функции.

Обычно используется два основных метода хеширования. Первый — предложенный А. Думи [47] и называемый методом цепочек, предполагает наличие m списков. Тогда для поступившего запроса x (он же запись) вычисляется значение хеш-функции $h(x)$, и если оно равно i, то просматривается i-ый список и в нем ищется запись x. Если это поиск с занесением, то в случае неудачного поиска запись x добавляется к i-ому списку.

Второй метод хеширования предложен А. П. Ершовым [12] и называется открытой адресацией. Он предполагает наличие закольцованного массива для m записей и наличие понятия пустой записи. После этого для поступившего запроса x вычисляется значение хеш-функции $h(x)$, и если оно равно i, то просматривается с i-ой позиции массив записей пока запись x будет найдена или пока не встретится пустая запись. Если это поиск с занесением, то в случае неудачного поиска на место встреченной пустой записи помещается запись x.

Методы хеширования хороши тем, что при удачном выборе хеш-функции, равномерно рассеивающей поступающие записи, время поиска в среднем будет очень малым. Вместе с тем Д. Кнут [15, стр. 641–642] усматривает три основных недостатка метода хеширования.

а) После неудачного поиска мы знаем лишь то, что нужной записи нет, тогда как с помощью бинарного поиска мы обнаруживаем ближайших соседей ненайденной записи, что часто бывает важно во многих приложениях.

б) Часто довольно трудно распределить память под хеш-таблицу. Если выделить слишком мало, то она может переполниться, и потребуется тягостное "рехеширование". Если выделить слишком много, то это расточительно.

в) "Наконец, при использовании методов хеширования нужно свято верить в теорию вероятностей, ибо они эффективны

лишь в среднем, а худший случай просто ужасен!— цитата из Д. Кнута [15, стр. 642]. Поэтому они не всегда подходят для работы в реальном масштабе времени, например, для управления движением транспорта, поскольку на карту поставлены человеческие жизни. Алгоритмы, использующие сбалансированные деревья, гораздо безопаснее, ведь они имеют гарантированную верхнюю границу времени поиска.

В данной работе предлагается метод поиска идентичных объектов, который в среднем эффективен, как хорошие методы хеширования, то есть обеспечивает мгновенное решение, а в худшем случае такой же, как метод бинарного поиска, и плюс к этому не обладает недостатком а), то есть в случае неудачного поиска выходит к ближайшим соседям ненайденной записи, что позволяет использовать этот алгоритм для решения задач о близости.

Опишем формально задачу поиска идентичных объектов.

Пусть нам дано множество X, на котором задано отношение линейного порядка $\preceq$, то есть такое бинарное отношение на $X \times X$, которое для любых $x, y, z \in X$ удовлетворяет условиям

- рефлексивности $x \preceq x$;
- транзитивности $(x \preceq y)\&(y \preceq z) \rightarrow (x \preceq z)$;
- антисимметричности $(x \preceq y)\&(y \preceq x) \rightarrow (x = y)$;
- связности $(x \preceq y) \vee (y \preceq x)$.

Рассмотрим следующий тип задач поиска $S_{id} = \langle X, X, \rho_{id} \rangle$, где отношение поиска ρ_{id} есть отношение идентичности, то есть

$$x \rho_{id} y \Longleftrightarrow x = y.$$

Тип S_{id} будем называть *типом поиска идентичных объектов.*

3.1 Бинарный поиск

Если V — конечное подмножество X, то через $\max\limits_{y \in V} y$ будем обозначать такое $y_{max} \in V$, что для любых $y \in V$ $y \preceq y_{max}$, и через $\min\limits_{y \in V} y$ обозначим такое $y_{min} \in V$, что для любых $y \in V$ $y_{min} \preceq y$.

Пусть

$$g_{\preceq,a}(x) = \begin{cases} 1, & \text{если } x \preceq a \\ 2 & \text{в противном случае} \end{cases}, a \in X, \qquad (3.1)$$

$$f_{=,a}(x) = \begin{cases} 0, & \text{если } x \neq a \\ 1, & \text{если } x = a \end{cases}, a \in X. \qquad (3.2)$$

$$G_1 = \{g_{\preceq,a}(x) : a \in X\}, \qquad (3.3)$$

$$F = \{f_{=,a}(x) : a \in X\}, \qquad (3.4)$$

$$\mathcal{F}_{bin} = \langle F, G_1 \rangle. \qquad (3.5)$$

Справедлива следующая теорема.

Теорема 8. *Если $I = \langle X, V, \rho_{id} \rangle$ — задача поиска идентичных объектов, то есть задача типа S_{id}, $\mathcal{F}_{bin}$ — базовое множество, определяемое соотношениями (3.1)–(3.5), то*

$$T(I, \mathcal{F}_{bin}, 2|V| - 1) \leq] \log_2 |V| [+ 1,$$

$$\widehat{T}(I, \mathcal{F}_{bin}, 2|V| - 1) \leq] \log_2 |V| [+ 1.$$

Доказательство: Пусть $|V| = k$.

Построим ИГ $D^1(V)$, имеющий вид дерева, следующим образом.

Возьмем бинарное сбалансированное дерево с k концевыми вершинами, высоты $] \log_2 k [$, все ребра которого ориентированы от корня к концевым вершинам. Если в этом дереве есть внутренняя вершина, из которой исходят ребра, ведущие одно в концевую вершину, а другое во внутреннюю (если такая вершина есть, то она только одна), то из концевой вершины, в которую ведет одно из этих ребер, выпустим одно ребро. Полученное дерево обозначим D. Объявим концевые вершины этого дерева листьями и сопоставим им слева направо в порядке возрастания элементы из V. (Говоря о дереве, мы подразумеваем некоторую его укладку, и направления "влево", "вправо"определяются уже на этой укладке.)

Пусть β — произвольная внутренняя вершина дерева D. Обозначим через V_β множество записей, соответствующих листьям, схемно достижимым из β. Пусть β' — вершина, в которую ведет левое (если оно одно, то единственное) ребро из β. Пусть

$$y_\beta = \max_{y \in V_{\beta'}} y.$$

Объявим все внутренние вершины дерева D, из которых исходят ребра, ведущие во внутренние вершины, вершинами переключения и для каждой такой вершины β левому ребру, из нее выходящему, припишем 1, а правому — 2, а самой вершине припишем переключатель $g_{\preceq,y_\beta}(x)$.

Все ребра дерева D, входящие в листья, объявим предикатными и припишем каждому такому ребру, ведущему в лист с записью y, предикат $f_{=,y}(x)$.

ИГ, полученный из дерева D после определения таким образом нагрузки, обозначим $D^1(V)$ и будем называть первым деревом бинарного поиска.

Покажем, что $D^1(V)$ решает задачу $I = \langle X, V, \rho_{id}\rangle$.

Возьмем произвольную запись $y \in V$. Пусть α — лист $D^1(V)$, которому приписана запись y. В этот лист ведет единственная цепь, состоящая из не более чем $]\log_2 k[$ ребер, причем все эти ребра, кроме последнего, переключательные. Последнее ребро в цепи — предикатное с предикатом $f_{=,y}(x)$, и его проводимость на запросе y равна $f_{=,y}(y) = 1$. Покажем, что проводимость остальных ребер цепи на запросе y равна 1. Возьмем произвольно одно из этих ребер (β, β'). Если (β, β') — левое, исходящее из β , то $y \in V_{\beta'}$ и предикат, приписанный вершине β, $g_{\preceq,y_\beta}(y)=1$, так как $y \preceq y_\beta = \max\limits_{y' \in V_{\beta'}} y'$. Если (β, β') — правое ребро, исходящее из β, то $y \not\preceq y_\beta$ и $g_{\preceq,y_\beta}(y) = 2$, тем самым проводимость ребра (β, β') в обоих случаях равна 1. Следовательно, $\varphi_\alpha(y) = 1$. Так как ребро, ведущее в лист α, имеет нагрузку $f_{=,y}$, то для любого запроса $x \neq y$ $\varphi_\alpha(x) = 0$.

Таким образом, мы показали, что $\varphi_\alpha(x) = f_{=,y}(x)$. Учитывая произвольность листа α и теорему 1, получим, что ИГ $D^1(V)$ решает задачу $I = \langle X, V, \rho_{id}\rangle$.

Подсчитаем объем ИГ $D^1(V)$.

Поскольку $D^1(V)$ есть дерево, в котором полустепень исхода любой вершины равна 2, кроме, быть может, одной вершины, то если в дереве $D^1(V)$ нет вершины с полустепенью исхода 1, то в $D^1(V)$ будет ровно $2 \cdot k - 2$ ребра, а если в дереве $D^1(V)$ есть вершина с полустепенью исхода 1, то в $D^1(V)$ будет $2 \cdot k - 1$ ребро. Отсюда следует, что

$$Q(D^1(V)) \leq 2k - 1. \tag{3.6}$$

Подсчитаем сложность ИГ $D^1(V)$.

Рассмотрим произвольный запрос $x \in X$. Как мы показали выше, активные на этом запросе вершины графа $D^1(V)$ (вершина β активна на запросе, если $\varphi_\beta(x) = 1$) образуют единственную цепь. Причем в этой цепи не более чем $]\log_2 k[-1$ переключательных вершин и одна вершина, из которой исходит не более двух ребер с предикатами. Тем самым

$$T(D^1(V), x) \leq]\log_2 k[+1.$$

Отсюда с учетом (3.6) сразу следует, что

$$T(I, \mathcal{F}_{bin}, 2k-1) \leq]\log_2 k[+1,$$

$$\widehat{T}(I, \mathcal{F}_{bin}, 2k-1) \leq]\log_2 k[+1.$$

Тем самым теорема доказана.

Отметим, что первое дерево бинарного поиска соответствует стандартному алгоритму бинарного поиска в версии Боттенбрука [44, с. 214].

Упражнения

3.1. Пусть $I = \langle X, V, \rho_c \rangle$ — задача о близости, где $X = \{1, 2, ..., N\}$, $V \subseteq X$, ρ_c — отношение поиска, задаваемое соотношением на $X \times V$ и определяемое соотношением (1.11). По аналогии с поиском идентичных объектов постройте ИГ, решающий задачу о близости I методом бинарного поиска. Получите соответствующие оценки сложности.

3.2 Константный в среднем алгоритм поиска

Пусть $\mathbf{N}_0 = \mathbf{N} \cup \{0\}$ — множество целых неотрицательных чисел.

Пусть

$$L_1(l) = \begin{cases} 0, & \text{если } l = 0 \\ ]\log_2 l[+1, & \text{если } l = 1, 2, 3 \\ \log_2 l + 2, & \text{если } l \geq 4 \end{cases} - \qquad (3.7)$$

функция, определенная на множестве $\mathbf{N}_0$.

Докажем следующее вспомогательное утверждение.

Лемма 7. *Пусть $L_1(l)$ — функция, определенная выше, пусть $k, m \in \mathbf{N}$, пусть*

$$r_1(k,m) \stackrel{\text{def}}{=} \max\{\sum_{i=1}^{m} L_1(l_i) : l_1 \in \mathbf{N}_0, \ldots, l_m \in \mathbf{N}_0, \sum_{i=1}^{m} l_i = k\}.$$

Тогда

$$\begin{aligned} r_1(k,m) &= \left(k - \left[\frac{k}{m}\right] \cdot m\right) \cdot L_1\left(\left[\frac{k}{m}\right] + 1\right) + \\ &+ \left(m - k + \left[\frac{k}{m}\right] \cdot m\right) \cdot L_1\left(\left[\frac{k}{m}\right]\right). \end{aligned}$$

Доказательство: Если доопределить функцию $L_1(l)$ на положительную полуось числовой прямой, например, следующим образом:

$$L_1(x) = \begin{cases} x, & \text{если } 0 \le x \le 4 \\ \log_2 x + 2, & \text{если } x \ge 4 \end{cases},$$

то полученная функция будет непрерывной и вогнутой. И, вообще говоря, вогнутость этой функции объясняет результат леммы.

Более подробное доказательство будем вести от противного.

Предположим, что лемма неверна, то есть

$$\begin{aligned} r_1(k,m) &= \sum_{i=1}^{m} L_1(l'_i) > \left(k - \left[\frac{k}{m}\right] \cdot m\right) \cdot L_1\left(\left[\frac{k}{m}\right] + 1\right) + \\ &+ \left(m - k + \left[\frac{k}{m}\right] \cdot m\right) \cdot L_1\left(\left[\frac{k}{m}\right]\right), \end{aligned} \tag{3.8}$$

и среди чисел $l'_i (i = \overline{1,m})$ существует 2 числа, разность которых не меньше двух.

Без ограничения общности можем считать, что $l'_1 - l'_2 \ge 2$.

Пусть

$$l''_1 = \left]\frac{l'_1 + l'_2}{2}\right[, \; l''_2 = \left[\frac{l'_1 + l'_2}{2}\right].$$

$l''_1 + l''_2 = l'_1 + l'_2$ и $l''_1 - l''_2 \le 1$.

Так как функция $L_1(x)$ вогнутая, то по свойству вогнутых функций

$$L_1(l_1'') + L_1(l_2'') \geq L_1(l_1') + L_1(l_2'). \tag{3.9}$$

Если в неравенстве (3.9) неравенство строгое, то получили противоречие, так как $\sum_{i=1}^{m} L_1(l_i')$ — не максимально; если же нет, то обозначим $l_i'' = l_i'$ $(i = \overline{3,m})$ и перейдем к рассмотрению суммы

$$\sum_{i=1}^{m} L_1(l_i'') = \sum_{i=1}^{m} L_1(l_i') = r_1(k,m).$$

Если среди чисел l_i'' $(i = \overline{1,m})$ нет пары чисел, разность которых превышает 1, то получим противоречие с неравенством (3.8), если же есть, то опять выполним операцию, описанную выше, и избавимся от такой пары, и так будем делать до тех пор, пока либо не получим строгое неравенство в неравенстве (3.9), либо не придем к разбиению $l_1^{(n)}, \ldots, l_m^{(n)}$, такому, что

$$\sum_{i=1}^{m} L_1(l_i^{(n)}) = r_1(k,m),$$

и все они отличаются не более чем на 1, и тем самым получим противоречие с неравенством (3.8), что доказывает лемму 7.

Пусть на X задано вероятностное пространство $\langle X, \sigma, \mathbf{P} \rangle$.

Пусть $g_m^1(x)$ — переключатель такой, что

$$g_m^1(x) = i, \text{ если } x \in X_i \ (i = \overline{1,m}), \tag{3.10}$$

где $X_1, \ldots, X_m$ — разбиение множества X (то есть $X = X_1 \cup \cdots \cup X_m$ и $X_i \cap X_j = \emptyset$, если $i \neq j$) такое, что

$$\mathbf{P}(X_i) \leq c/m \ (i = \overline{1,m}), \tag{3.11}$$

где c — постоянная, не зависящая от m.

Так как

$$\mathbf{P}(X) = \sum_{i=1}^{m} \mathbf{P}(X_i) \leq c,$$

то отсюда сразу следует, что $c \geq 1$.

Пусть

$$G_2 = \{g_m^1(x) : m \in \mathbf{N}\}, \tag{3.12}$$

$$\mathcal{F} = \langle F, G_1 \cup G_2 \rangle. \tag{3.13}$$

Справедлива следующая теорема.

Теорема 9. *Пусть $I = \langle X, V, \rho_{id} \rangle$ — задача поиска идентичных объектов, то есть задача типа S_{id}, где $|V| = k$, $\mathcal{F}$ — базовое множество, определяемое соотношениями (3.1)–(3.4), (3.10)–(3.13), m — натуральное число, c — константа, определяемая соотношением (3.11), $L_1(l)$ — функция, определяемая соотношением (3.7). Тогда*

$$\begin{aligned} 1 \quad &\leq \quad T(I, \mathcal{F}, 2 \cdot k + m - 1) \leq \\ &\leq \quad \frac{c}{m} \left(\left(k - \left[\frac{k}{m} \right] \cdot m \right) \cdot L_1 \left(\left[\frac{k}{m} \right] + 1 \right) + \right. \\ &\qquad \left. + \left(m - k + \left[\frac{k}{m} \right] \cdot m \right) \cdot L_1 \left(\left[\frac{k}{m} \right] \right) \right) + 1. \end{aligned}$$

В частности,

$$1 \leq T(I, \mathcal{F}, (2 + c) \cdot k) \leq 2$$

и $T(I, \mathcal{F}) \sim 1$ при $k \to \infty$. Кроме того, для ИГ $U \in \mathcal{U}(I, \mathcal{F})$, на котором достигается верхняя оценка, $\widehat{T}(U) \leq 2 +] \log_2 k [$.

Доказательство: Построим ИГ U_m^0, имеющий вид дерева, следующим образом.

Возьмем вершину β_0 и объявим ее корнем графа U_m^0. Выпустим из β_0 m ребер, припишем им числа от 1 до m, объявим β_0 точкой переключения и припишем ей переключатель $g_m^1(x)$.

Пусть $V_i = X_i \cap V,\ l_i = |V_i|,\ i = \overline{1, m}$.

Конец ребра с номером i обозначим β_i.

Для всех таких i, что $V_i \neq \emptyset$, выпустим из вершины β_i первое дерево бинарного поиска, описанное в разделе 3.1, то есть дерево $D^1(V_i)$.

Полученный ИГ с $|V| = k$ листьями обозначим U_m^0. Это граф над базовым множеством $\mathcal{F}$.

Покажем, что U_m^0 решает задачу $I = \langle X, V, \rho_{id} \rangle$.

Возьмем произвольную запись $y \in V$. Пусть $y \in V_i$ ($i \in \{\overline{1, m}\}$), то есть лист α, которому приписана запись y, принадлежит дереву $D^1(V_i)$. Так как $g_m^1(y) = i$, запрос y пройдет по

i-му ребру, исходящему из корня, в корень дерева $D^1(V_i)$. Так как $D^1(V_i)$ решает задачу поиска идентичных объектов для библиотеки V_i, то только запрос y пройдет в лист α.

Таким образом, мы показали, что $\varphi_\alpha(x) = f_{=,y}(x)$. Учитывая произвольность выбора y и теорему 1, получим, что ИГ U_m^0 решает задачу $I = \langle X, V, \rho_{id} \rangle$.

Подсчитаем объем графа U_m^0.

$$Q(U_m^0) \le m + \sum_{i=1}^{m} \max(0, 2 \cdot l_i - 1) \le 2 \cdot k + m - 1.$$

Подсчитаем сложность ИГ U_m^0.

Рассмотрим произвольный запрос $x \in X$. Пусть $x \in X_i$ $(i = \overline{1,m})$.

$$T(U_0^m, x) = 1 + T(D^1(V_i), x) \le 2 +]\log_2 l_i[\le 1 + L_1(l_i), \quad (3.14)$$

где $L_1(l)$ — функция, определяемая соотношением (3.7).

По определению и учитывая лемму 7,

$$\begin{aligned}
T(U_0^m) &= \mathbf{M}_x\, T(U_0^m, x) = \int_X T(U_0^m, x)\, \mathbf{P}(dx) = \\
&= \sum_{i=1}^{m} \int_{X_i} T(U_0^m, x)\, \mathbf{P}(dx) \le \sum_{i=1}^{m} (1 + L_1(l_i)) \cdot \mathbf{P}(X_i) = \\
&= 1 + \sum_{i=1}^{m} L_1(l_i) \cdot \mathbf{P}(X_i) \le 1 + \frac{c}{m} \sum_{i=1}^{m} L_1(l_i) \le \\
&\le 1 + r_1(k,m)\, \frac{c}{m} = \\
&= \frac{c}{m} \left(\left(k - \left[\frac{k}{m} \right] \cdot m \right) \cdot L_1 \left(\left[\frac{k}{m} \right] + 1 \right) + \right. \\
&\quad \left. + \left(m - k + \left[\frac{k}{m} \right] \cdot m \right) \cdot L_1 \left(\left[\frac{k}{m} \right] \right) \right) + 1.
\end{aligned}$$

Возьмем $m =]c \cdot k[$. При этом

$$Q(U_0^m) \le 2k +]c \cdot k[- 1 \le (2 + c) \cdot k.$$

Так как $c \ge 1$, то $m \ge k$. Если $m = k$, то $k/m = 1$ и

$$r_1(m,k) = 0 \cdot L_1(2) + k \cdot L_1(1) = k \cdot L_1(1).$$

Если $m > k$, то $k/m = 0$ и

$$r_1(m,k) = k \cdot L_1(1) + (m - k) \cdot L_1(0) = k \cdot L_1(1).$$

Следовательно,

$$T(I,\mathcal{F},(2+c)\cdot k) \le T(U_0^m) \le 1 + \frac{c}{]c\,k[}\,k \cdot L_1(1) \le 2.$$

Если взять $m =]k\cdot\gamma(k)[$, где $\gamma(k) \to \infty$ при $k \to \infty$ и $\gamma(k) > 1$ при любом k, то

$$T(I,\mathcal{F}) \le 1 + \frac{c\cdot k}{]k\cdot\gamma(k)[} \lesssim 1.$$

С другой стороны, $T(I,\mathcal{F}) \ge 1$, так как из корня должно исходить хотя бы одно ребро, поскольку $k > 0$.

И наконец, оценим B-сложность ИГ U_0^m, воспользовавшись соотношение (3.14),

$$\widehat{T}(U_0^m) = \max_{x\in X} T(U_0^m, x) \le 2 + \max_{1\le i\le m}]\log_2 l_i[\le 2 +]\log_2 k[.$$

Тем самым теорема доказана.

Следствие 5. *Если $\mathcal{F}$ — базовое множество, определяемое соотношениями (3.1)–(3.4), (3.10)–(3.13), k — натуральное число, то*

$$1 \le T(k, S_{id}, \mathcal{F}) \le \mathcal{T}(k, S_{id}, \mathcal{F}) \le 2$$

и при $k \to \infty$

$$T(k, S_{id}, \mathcal{F}) \sim \mathcal{T}(k, S_{id}, \mathcal{F}) \sim 1.$$

Приведем два примера задачи поиска идентичных объектов.

Пример 1. Пусть $S_{id1} = \langle X, X, \rho_{id}\rangle$ — тип поиска идентичных объектов, где $X = [0,1]$, причем задано вероятностное пространство $\langle X, \sigma, \mathbf{P}\rangle$ над X такое, что $\mathbf{P}$ задается функцией плотностью вероятности $p(x)$.

Пусть

$$\begin{aligned} X_1 &= \{x \in X : 0 \le x \le 1/m\}. \\ X_i &= \{x \in X : \frac{i-1}{m} < x \le \frac{i}{m}\},\ i = \overline{2,m}. \end{aligned} \qquad (3.15)$$

Тогда переключатель $g_m^1(x) = \max(1,]x\cdot m[)$ удовлетворяет соотношению (3.10).

Если $p(x) \le c = const$, то $\mathbf{P}(X_i) \le c/m$, и мы находимся в условиях теоремы 9.

Приведем неформальное описание алгоритма поиска, описанного в доказательстве теоремы 9, применительно к задаче $I = \langle X, V, \rho_{id} \rangle$ типа S_{id1}, где $|V| = k$.

Разобьем отрезок $[0, 1]$ на m равных частей в соответствии с соотношениями (3.15).

Каждой части сопоставим подмножество множества V, состоящее из точек, принадлежащих этой части.

Теперь для произвольной точки-запроса x поиск точки из V, идентичной запросу, будем вести следующим образом.

Определим ту часть отрезка, которой принадлежит запрос x. Ее номер равен $\max(1,]x \cdot m[)$.

Далее во множестве, сопоставленном найденной части, осуществим обычный бинарный поиск.

Согласно лемме 7 наибольшее среднее время поиска будет в том случае, когда точки из множества V равномерно распределены по всем m частям отрезка.

Если в качестве m взять $m = k$, то случай, когда в каждую часть попадет по одной точке из V, будет иметь наибольшую сложность. А поскольку найти или не найти объект во множестве мощности 1 можно за 1 шаг, то в среднем за 2 шага (на первом мы определили номер нужной части) мы можем решить задачу поиска идентичных объектов.

Отметим, что в случае неудачного поиска алгоритм выводит к ближайшим соседям ненайденной точки.

Если внимательно присмотреться к описанному алгоритму, то можно заметить, что по своей структуре он напоминает хеширование по методу цепочек, в котором в качестве хеш-функции взято умножение, а цепочки-списки организованы в виде бинарного дерева.

Пример 2. Пусть $S_{id2} = \langle X, X, \rho_{id} \rangle$ — тип поиска идентичных объектов, где $X = \{1, \ldots, N\}$.

Пусть $\langle X, \sigma, \mathbf{P} \rangle$ — вероятностное пространство над X, где σ — множество всех подмножеств, а вероятностная мера $\mathbf{P}$ определяет равномерную вероятность на множестве запросов X, то есть для любого $x \in X \quad \mathbf{P}(x) = 1/N$.

Пусть задано $m \in \mathbf{N}$.

Пусть $r = N - m \cdot [N/m]$.

Пусть $X_1, \ldots, X_m$ — такое разбиение множества, что

$$\begin{aligned} X_i &= \{x \in X : 1 + (i-1) \cdot ([N/m]+1) \le x \le i \cdot ([N/m]+1)\}, \\ &\quad i = \overline{1,r}, \\ X_i &= \{x \in X : r \cdot ([N/m]+1) + 1 + (i-1-r) \cdot [N/m] \le x \le \\ &\quad \le r \cdot ([N/m]+1) + (i-r) \cdot [N/m]\},\ i = \overline{r+1,m}, \end{aligned}$$

и $g_m^1(x) = i$, если $x \in X_i,\ i \in \{\overline{1,m}\}$. Тогда

$$\mathbf{P}(X_i) \le ([N/m]+1)/N < 2/m,$$

и мы опять находимся в условиях теоремы 9 при $c = 2$.

Упражнения

3.2. Пусть $I = \langle X, V, \rho_c \rangle$ — задача о близости, где $X = \{1, 2, ..., N\}$, $V \subseteq X$, ρ_c — отношение поиска, задаваемое соотношением на $X \times V$ и определяемое соотношением (1.11). По аналогии с поиском идентичных объектов постройте ИГ, решающий задачу о близости I и имеющий константную сложность.

3.3 Константный в худшем случае алгоритм поиска

Теорема 10. *Если $I = \langle X, V, \rho_{id} \rangle$ — задача поиска идентичных объектов, $\mathcal{F} = \langle F, G_2 \rangle$ — базовое множество, определяемое соотношениями (3.2),(3.4),(3.10),(3.12), такое, что существует такое натуральное число m, что для любых различных $y, y' \in V$ $g_m^1(y) \ne g_m^1(y')$, то $1 \le \widehat{T}(I, \mathcal{F}) \le 2$.*

Доказательство: Рассмотрим ИГ U, построенный следующим образом. Возьмем вершину, которую объявим корнем ИГ. Пусть m — число, упомянутое в условиях теоремы. Выпустим из корня m ребер, объявим их переключательными и занумеруем подряд, начиная с 1. Припишем корню переключатель g_m^1. Для каждой записи $y \in V$ из конца ребра с номером $g_m^1(y)$ выпустим предикатное ребро с предикатом $f_{=,y}$, конец этого ребра объявим листом и припишем этому листу запись y. Понятно,

что U решает ЗИП I, и для любого запроса $x \in X$ выполняется $T(U,x) \leq 2$. Следовательно, $\widehat{T}(I,\mathcal{F}) \leq 2$. Но поскольку одно вычисление всегда надо сделать, то $\widehat{T}(I,\mathcal{F}) \geq 1$.

Тем самым теорема доказана.

Следствие 6. *Если $\mathcal{F} = \langle F, G_2 \rangle$ — базовое множество, определяемое соотношениями (3.2),(3.4),(3.10),(3.12), такое, что для любой библиотеки V существует такое натуральное число m, что для любых различных записей $y, y' \in V$ $g^1_m(y) \neq g^1_m(y')$, то для любого натурального k выполнено*

$$1 \leq \widehat{T}(k, S_{id}, \mathcal{F}) \leq \widehat{T}(k, S_{id}, \mathcal{F}) \leq 2.$$

3.4 Оценки памяти константного в худшем случае алгоритма поиска

В данном разделе рассматривается задача поиска идентичных объектов в ее геометрической интерпретации, которая звучит следующим образом. Дано конечное подмножество $V = \{y_1, \ldots, y_k\}$ точек из отрезка $[0,1]$ вещественной прямой. Требуется построить условный алгоритм, который для произвольной точки $x \in [0,1)$ (эта точка называется запросом) позволяет найти номер точки из множества V, которая совпадает с x (если такая точка в V существует), при условии, что мы умеем выполнять следующие операции над вещественными числами: арифметические операции (сложение, вычитание, умножение, деление, взятие целой части вещественного числа), операции сравнения и возможно некоторые другие простейшие операции. При этом допускается предобработка данных, которая может состоять в сортировке данных (множества V), а также в построении некоторых дополнительных структур.

В данном разделе детализируется алгоритм, описанный в предыдущем пункте, применительно к данной геометрической интерпретации, и показывается, что для почти всех задач поиска идентичных объектов (то есть при вариации множества V) данный алгоритм позволяет при объеме памяти порядка k^2 решать задачу в худшем случае за константное число элементарных операций, то есть операций, которые обычно используются

в компьютерах. Здесь под объемом памяти понимается количество ячеек для хранения вещественных чисел, куда можно поместить данные и дополнительные структуры, а худший случай берется по множеству всех возможных значений запроса, то есть по множеству $[0, 1)$. Здесь для упрощения дальнейшего изложения мы выбросили точку 1 из множества запросов.

Результаты данного раздела опубликованы в [11].

Опишем предлагаемый алгоритм. Пусть нам дано множество $V = \{y_1, y_2, \ldots, y_k\}$, в котором производится поиск. Это множество будем называть библиотекой. Выполним следующую предобработку. Упорядочим точки из V в порядке возрастания и, чтобы не усложнять обозначения, далее считаем, что $y_1 < y_2 < \cdots < y_k$. Находим число $d_V = \min\limits_{2 \le i \le k} (y_i - y_{i-1})$. Пусть $m =]1/d_V[$ — наименьшее целое, не меньшее, чем $1/d_V$. Выделим место под массив целых длины m, и элементы этого массива будем обозначать n_i $(i = 0, 1, 2, \ldots, m-1)$. Разделим отрезок $[0, 1]$ на m равных частей:

$$A_i = [i/m, (i+1)/m),\ i = 0, 1, \ldots, m-2,\ A_{m-1} = [(m-1)/m, 1].$$

В каждую часть может попасть не более одной точки из множества V. Теперь заполним массив n_i следующим образом:

$$n_i = \begin{cases} -1, & \text{если в } A_i \text{ не попало ни одной точки из } V \\ q & \text{в противном случае, где } q \text{ — номер точки из } V, \\ & \text{которая попала в } A_i, \end{cases}$$

где $i = 0, 1, 2, \ldots, m-1$.

После того как сделана данная предобработка, поиск будем осуществлять следующим образом. Пусть нам дан запрос $x \in [0, 1)$. Вычислим $j = [x \cdot m]$ — целая часть числа $x \cdot m$. Понятно, что $x \in A_j$. Если n_j равно -1, то в библиотеке V нет числа равного x. В противном случае сравниваем y_{n_j} с x и если они равны, то номер n_j является ответом задачи, иначе ответ пуст. Тем самым в худшем случае мы выполняем одну операцию умножения, одну операцию взятия целой части, одну операцию сравнения целых чисел, одну операцию сравнения вещественных чисел и две операции извлечения элемента массива, всего 6 элементарных операций. Объем памяти, необходимый данному алгоритму, равен сумме объемов массивов целых чисел длины

m и вещественных чисел длины k. Ниже приводятся результаты, оценивающие величину m.

Пусть k — натуральное число, большее 1 и $\xi_1, \xi_2, \ldots, \xi_k$ — независимые равномерно распределенные на отрезке $[0,1]$ случайные величины. Пусть

$$d(\xi_1, \ldots, \xi_k) = \min_{1 \le i < j \le k} |\xi_i - \xi_j|.$$

Пусть r — вещественное число и $f(k,r) = \mathbf{P}(d(\xi_1, \ldots, \xi_k) \ge r)$ — вероятность того, что минимальное расстояние между парами различных точек ξ_i $(i = 1, 2, \ldots, n)$ не меньше r.

Если считать, что библиотеки V получаются случайным независимым бросанием k точек на отрезок $[0,1]$, где вероятность попадания в любую пару отрезков одинаковой длины одинакова, то $f(k,r)$ равна доле множества k-элементных библиотек, у которых минимальное расстояние между любыми двумя точками не меньше чем r, по отношению к множеству всех библиотек мощности k.

Справедлива следующая теорема.

Теорема 11.

$$f(k,r) = \begin{cases} 1 & \text{если } r < 0 \\ (1 - (k-1) \cdot r)^k & \text{если } 0 \le r \le 1/(k-1) \\ 0 & \text{если } r > 1/(k-1). \end{cases}$$

Доказательство:

Пусть l — вещественное число из отрезка $[0,1]$, r — вещественное число, k — натуральное число, большее 1, n — такое натуральное число, что $1 \le n \le k$. Пусть $x_1, x_2, \ldots, x_k$ — независимые равномерно распределенные на отрезке $[0,1]$ случайные величины. Обозначим через $B(n,r,l)$ событие, что точки $x_1, x_2, \ldots, x_n$ попадают в отрезок $[0,l]$, и минимальное расстояние между парами различных точек x_i $(i = 1, 2, \ldots, n)$, не меньше r. Обозначим через $f(n,r,l) = \mathbf{P}(B(n,r,l))$. Понятно, что если $r < 0$, то $f(n,r,l) = l^n$, а при $r > l/(n-1)$, $f(n,r,l) = 0$. Поэтому мы будем рассматривать только случай, когда $0 \le r \le l/(n-1)$.

Лемма 8. *Если* $0 \le r \le l/(n-1)$, *то* $f(n,r,l) = (l - (n-1) \cdot r)^n$.

Доказательство будем вести индукцией по n.

Базис индукции. $n = 2$.

Поскольку возможны два равновероятных события: случай когда $x_1 < x_2$, и когда $x_2 < x_1$, то достаточно рассмотреть первую ситуацию и удвоить полученный результат. Поскольку в этом случае x_1 может меняться от 0 до $l - r$, а x_2 — от $x_1 + r$ до l, то

$$f(2, r, l) = 2 \int_0^{l-r} dx_1 \int_{x_1+r}^{l} dx_2 = 2 \int_0^{l-r} (l - x_1 - r) dx_1 = (l - r)^2.$$

Индуктивный переход. Пусть утверждение леммы справедливо для любого натурального $q < n$ и любого вещественного $l \in [0, 1]$.

Через A_i обозначим событие, что случайная величина x_i, максимальна среди величин $x_1, \ldots, x_n$, здесь $i = 1, \ldots, n$. Понятно, что если $i, j \in \{1, \ldots, n\}$ и $i \neq j$, то $A_i \cap A_j = \emptyset$, кроме того $\mathbf{P}(A_i) = 1/n$, для любого $i \in \{1, \ldots, n\}$.

Легко видеть, что

$$\mathbf{P}(B(n, r, l)) = \sum_{i=0}^{n} \mathbf{P}(A_i \cap B(n, r, l)) = n \cdot \mathbf{P}(A_n \cap B(n, r, l)).$$

Поскольку в случае события $A_n \cap B(n, r, l)$ величина x_n может меняться от $(n - 1)r$ до l, а остальные $n - 1$ величины располагаются на отрезке $[0, x_n - r]$ и должны находиться на расстоянии не менее r, то согласно предположению индукции

$$\begin{aligned} f(n, r, l) &= \mathbf{P}(B(n, r, l)) = n \cdot \mathbf{P}(A_1 \cap B(n, r, l)) = \\ &= n \int_{(n-1)r}^{l} \mathbf{P}(B(n - 1, r, x_n - r)) dx_n = \\ &= n \int_{(n-1)r}^{l} (x_n - r - (n - 2)r)^{n-1} dx_n = \\ &= (l - (n - 1)r)^n. \end{aligned}$$

Тем самым лемма доказана.

Чтобы убедиться в справедливости утверждения теоремы 11 достаточно заметить, что $f(k, r) = f(k, r, 1)$.

Тем самым теорема 11 доказана.

Следующая теорема описывает асимптотическое поведение функции $f(k,r)$.

Теорема 12. *Пусть r_k — последовательность вещественных чисел, такая, что $0 \le r_k \le 1/(k-1)$. Тогда*

$$\lim_{k\to\infty} f(k,r_k) = \begin{cases} 0, & \text{если } 1/r_k = \bar{o}(k^2) \\ e^{-1/c}, & \text{если } 1/r_k \sim c\cdot k^2, \text{ где } c = const \\ 1, & \text{если } k^2 = \bar{o}(1/r_k). \end{cases}$$

Доказательство: Пусть $\alpha_k = \bar{o}(k)$ при $k \to \infty$. Воспользовавшись вторым замечательным пределом, легко получить

$$\begin{aligned} \lim_{k\to\infty}\left(1-\frac{\alpha_k}{k}\right)^k &= \lim_{k\to\infty}\left(\frac{k}{k-\alpha_k}\right)^{-k} = \\ &= \lim_{k\to\infty}\left(\left(1+\frac{\alpha_k}{k-\alpha_k}\right)^{\frac{k-\alpha_k}{\alpha_k}}\right)^{-\frac{\alpha_k k}{k-\alpha_k}} = \\ &= \lim_{k\to\infty} e^{-\frac{k\alpha_k}{k-\alpha_k}} = \lim_{k\to\infty} e^{-\alpha_k}. \end{aligned} \tag{3.16}$$

Рассмотрим случай, когда $1/r_k = \bar{o}(k^2)$ при $k \to \infty$.

Это означает, что для некоторой последовательности $\alpha_k \to \infty$ при $k \to \infty$ выполняется $r_k = \alpha_k/k^2$. Поскольку $r_k \le 1/(k-1)$, то достаточно рассмотреть два подслучая: $\alpha_k \sim c\cdot k$, где c — константа не превышающая 1, и $\alpha_k = \bar{o}(k)$. В первом подслучае

$$f(k,r_k) = \left(1-\frac{(k-1)\alpha_k}{k^2}\right)^k \sim \left(1-\frac{c(k-1)k}{k^2}\right)^k = \bar{o}(1).$$

Так как во втором подслучае $(k-1)\alpha_k/k = \bar{o}(k)$, то согласно (3.16)

$$\begin{aligned} \lim_{k\to\infty} f(k,r_k) &= \lim_{k\to\infty}\left(1-\frac{(k-1)\alpha_k}{k^2}\right)^k = \\ &= \lim_{k\to\infty} e^{-\frac{(k-1)\alpha_k}{k}} = \lim_{k\to\infty} e^{-\alpha_k} = 0. \end{aligned}$$

Рассмотрим случай, когда $1/r_k \sim ck^2$ при $k \to \infty$, где $c = const$.

Поскольку $(k-1)/ck = \bar{o}(k)$, то согласно (3.16)

$$\lim_{k\to\infty} f(k, r_k) = \lim_{k\to\infty} \left(1 - \frac{k-1}{ck^2}\right)^k = \lim_{k\to\infty} e^{-\frac{k-1}{ck}} = e^{-1/c}.$$

И наконец, рассмотрим случай, когда $k^2 = \bar{o}(1/r_k)$ при $k \to \infty$.

Это означает, что для некоторой последовательности $\alpha_k \to 0$ при $k \to \infty$ выполняется $r_k = \alpha_k/k^2$. Поскольку $(k-1)\alpha_k/k = \bar{o}(k)$, то согласно (3.16)

$$\begin{aligned}\lim_{k\to\infty} f(k, r_k) &= \lim_{k\to\infty} \left(1 - \frac{(k-1)\alpha_k}{k^2}\right)^k = \\ &= \lim_{k\to\infty} e^{-\frac{(k-1)\alpha_k}{k}} = \lim_{k\to\infty} e^{-\alpha_k} = 1.\end{aligned}$$

Тем самым теорема 12 доказана.

Отсюда следует, что если в нашем распоряжении имеется объем памяти размера $c \cdot k^2$, то доля библиотек мощности k, для которых описанным алгоритмом мы можем находить ответ за 6 элементарных операций, равна $e^{-1/c}$. А если в нашем распоряжении имеется объем памяти больший по порядку, чем k^2, то для почти всех библиотек мы можем находить ответ за 6 элементарных операций. С другой стороны, если у нас имеется объем памяти, меньший по порядку, чем k^2, то почти всегда мы не сможем воспользоваться описанным выше алгоритмом поиска.

Обозначим через $\overline{d}(k)$ среднее значение описанной выше величины $d(\xi_1, \ldots, \xi_k)$, тогда справедливо следующее утверждение.

Теорема 13. $\overline{d}(k) = 1/(k^2 - 1)$.

Доказательство: Обозначим через $F(x)$ функцию распределения случайной величины $d(\xi_1, \ldots, \xi_k)$.

$$F(x) = \mathbf{P}(d(\xi_1, \ldots, \xi_k) < x) = 1 - \mathbf{P}(d(\xi_1, \ldots, \xi_k) \geq x) = 1 - f(k, x).$$

Тогда так как при $x \leq 0$ $F(x) = 0$, а при $x \geq 1/(k-1)$ $F(x) = 1$, то используя формулу интегрирования по частям, нетрудно

получить

$$\begin{aligned}\overline{d}(k) &= \int_{-\infty}^{\infty} x dF(x) = \int_0^{\frac{1}{k-1}} x dF(x) = \\ &= xF(x)|_0^{\frac{1}{k-1}} - \int_0^{\frac{1}{k-1}} F(x)dx = \\ &= \int_0^{\frac{1}{k-1}} f(k,x)dx = \int_0^{\frac{1}{k-1}} (1-(k-1)x)^k dx = \frac{1}{k^2-1}.\end{aligned}$$

Тем самым теорема 13 доказана.

Отсюда следует, что в типичной ситуации достаточно иметь k^2 памяти, чтобы обеспечить время поиска в 6 элементарных операций.

Глава 4

Включающий поиск

4.1 Задачи поиска на конечных частично-упорядоченных множествах данных

Задачи поиска, в которых отношение поиска, является отношением частичного порядка, встречаются практически во всех системах баз данных и в зависимости от интерпретации носят название включающего поиска [28], дескрипторного поиска [26, 31, 32], поиска по ключевым словам [15], задачи о доминировании [25] и т. п.

В данном разделе мы приведем некоторые свойства отношений частичного порядка, полезные в том случае, когда эти отношения используются в качестве отношений поиска. Результаты, приведенные в данном разделе, были опубликованы в [9]. Мы будем рассматривать типы задач поиска, в которых в качестве отношения поиска выступает отношение частичного порядка, а именно рассмотрим следующий тип $S_{part} = \langle X, X, \succeq \rangle$, где X — некоторое конечное множество, $\succeq$ — некоторое *отношение частичного порядка* на $X \times X$, под которым будем понимать отношение, для любых $x, y, z \in X$ удовлетворяющее условиям

- рефлексивности $x \succeq x$;

- транзитивности $(x \succeq y)\&(y \succeq z) \to (x \succeq z)$;

- антисимметричности $(x \succeq y)\&(y \succeq x) \to (x = y)$.

Пусть $a \in X$, K_a — функция, действующая из X в $\{0,1\}$ такая, что $N_{K_a} = \{x \in X : x \succeq a\}$. Отметим, что $K_a(x)$ есть характеристическая функция записи a. Пусть $\mathcal{K} = \{K_a(x) : a \in X\}$, $\mathcal{M} = \{\bigvee_{i=1}^{n} K_{a_i}(x) : a_i \in X, n \in \mathbf{N}\}$, где $\mathbf{N}$ — множество натуральных чисел.

Справедливы следующие свойства.

Свойство 1. *Если $a, b \in X$ и $K_b(a) = 1$, то $N_{K_a} \subseteq N_{K_b}$.*

Доказательство. Так как $K_b(a) = 1$, то $a \succeq b$. Для любого $c \in N_{K_a}$ справедливо $c \succeq a \succeq b$. Отсюда следует, что $c \succeq b$ и, следовательно, $c \in N_{K_b}$. Что и требовалось доказать.

Свойство 2. *Если $a \in X$, $f \in \mathcal{M}$ и $f(a) = 1$, то $N_{K_a} \subseteq N_f$.*

Доказательство. Пусть $f = \bigvee_{i=1}^{n} K_{a_i}$. Так как $f(a) = 1$, то существует K_{a_i} такое, что $K_{a_i}(a) = 1$, но тогда согласно свойству 1 $N_{K_a} \subseteq N_{K_{a_i}}$ и, следовательно, $N_{K_a} \subseteq N_f$. Что и требовалось доказать.

Свойство 3. *Пусть $a \in X$ и $\bigvee_{i=1}^{n} f_i = K_a$, где $f_i \in \mathcal{M}$. Тогда существует такое $i \in \{\overline{1,n}\}$, что $f_i = K_a$.*

Доказательство. Так как $K_a(a) = 1$, то существует такое i, что $f_i(a) = 1$. Тогда согласно свойству 2 имеем $N_{K_a} \subseteq N_{f_i}$, но из условия следует, что $N_{f_i} \subseteq N_{K_a}$, следовательно, $N_{K_a} = N_{f_i}$, то есть $K_a = f_i$. Что и требовалось доказать.

Свойство 4. *Если $a \in X$, $f = \bigwedge_{i=1}^{n} f_i$, где $f_i \in \mathcal{M}$, и $f(a) = 1$, то $N_{K_a} \subseteq N_f$.*

Доказательство. $N_f = \bigcap_{i=1}^{n} N_{f_i}$. Так как $a \in N_f$, то $a \in N_{f_i}$ для любого $i = 1, \ldots, n$. Отсюда согласно свойству 2 имеем

$N_{K_a} \subseteq N_{f_i}$ для любого $i = 1, \ldots, n$. Следовательно, $N_{K_a} \subseteq \bigcap_{i=1}^{n} N_{f_i} = N_f$. Что и требовалось доказать.

Точку $a \in R \subseteq X$ назовем *нижней единицей множества* R, если не существует точки $b \in R \setminus \{a\}$ такой, что $a \succeq b$.

Свойство 5. *Если* $f = \bigwedge_{i=1}^{n} f_i$, *где* $f_i \in \mathcal{M}$, *то либо* $f \in \mathcal{M}$, *либо* $f \equiv 0$.

Доказательство. $N_f = \bigcap_{i=1}^{n} N_{f_i}$. Допустим, что $\bigcap_{i=1}^{n} N_{f_i} \neq \emptyset$, то есть $f \not\equiv 0$. Просмотрим все точки множества N_f (а их конечное число) и для каждой проверим, является ли она нижней единицей? Пусть $M = \{a_1, \ldots, a_m\}$ — множество всех нижних единиц множества N_f, которое мы в результате получили. Легко видеть, что для любой точки $a \in N_f$ либо $a \in M$, либо существует $a_i \in M$ такое, что $a \succeq a_i$. Из того, что $a \succeq a_i$ следует $K_{a_i}(a) = 1 \Rightarrow a \in N_{K_{a_i}}$. Следовательно, $N_f \subseteq \bigcup_{i=1}^{m} N_{K_{a_i}}$. С другой стороны, согласно свойству 4 (используем то, что $f(a_i) = 1$) для любого $i \in \{1, \ldots, m\}$ имеем $N_{K_{a_i}} \subseteq N_f$. Следовательно, $N_f = \bigcup_{i=1}^{m} N_{K_{a_i}}$ и, значит, $f \in \mathcal{M}$, что и требовалось доказать.

Пусть базовое множество имеет вид $\mathcal{F} = \langle \mathcal{M}, \emptyset \rangle$. Пусть U — ПИГ над базовым множеством $\mathcal{F}$. Подмножество $\{\beta_1, \ldots, \beta_m\}$ вершин ПИГ U назовем *характерным*, если существует такая запись $a \in X$, что $\bigvee_{i=1}^{m} \varphi_{\beta_i} = K_a$.

Пусть U — ПИГ над базовым множеством $\mathcal{F} = \langle \mathcal{M}, \emptyset \rangle$. Пусть $\mathcal{B} = \{\beta_1, \ldots, \beta_m\}$ — характерное подмножество вершин ПИГ U такое, что $\bigvee_{i=1}^{m} \varphi_{\beta_i} = K_a$. Если в ПИГ U существует такая цепь, ведущая из корня в какую-либо вершину множества $\mathcal{B}$, что проводимость этой цепи равна K_a, то эту цепь назовем *главной цепью характерного множества* $\mathcal{B}$.

Теорема 14 (о существовании главных цепей). *Пусть U — произвольный ПИГ над базовым множеством $\mathcal{F} = \langle \mathcal{M}, \emptyset \rangle$. Пусть $\mathcal{B}$ — произвольное характерное подмножество вершин ПИГ U. Тогда в ПИГ U существует главная цепь множества $\mathcal{B}$.*

Доказательство. Рассмотрим некоторую цепь C графа U. Обозначим через f_C проводимость цепи C. Пусть C такая цепь, что $f_C \not\equiv 0$. Так как f_C равна конъюнкции нагрузок ребер цепи C, то согласно свойству 5 $f_C \in \mathcal{M}$. Пусть $\mathcal{C}_\mathcal{B}$ - множество цепей, ведущих из корня в вершины множества $\mathcal{B}$. Так как $\mathcal{B}$ — характерное множество, то существует такая запись $a \in X$, что $\bigvee\limits_{\beta \in \mathcal{B}} \varphi_\beta = \bigvee\limits_{C \in \mathcal{C}_\mathcal{B}} f_C = K_a$. Отсюда согласно свойству 3 существует цепь $C' \in \mathcal{C}_\mathcal{B}$ такая, что $f_{C'} = K_a$. Эта цепь C' и есть главная цепь множества $\mathcal{B}$. Что и требовалось доказать.

Пусть U — ПИГ над базовым множеством $\mathcal{F} = \langle \mathcal{M}, \emptyset \rangle$, решающий некоторую ЗИП $I = \langle X, V, \succeq \rangle$ типа S_{part}. Пусть $y \in V$. Если в графе U существует такая цепь, ведущая из корня в какую-либо вершину множества $L_U(y)$, что проводимость этой цепи равна $\chi_{y,\succeq}$, то эту цепь назовем *главной цепью записи y*.

Следствие 7. *Пусть $I = \langle X, V, \succeq \rangle$ — ЗИП типа S_{part}. Пусть U — ПИГ над базовым множеством $\mathcal{F} = \langle \mathcal{M}, \emptyset \rangle$, решающий ЗИП I. Тогда для любой записи $y \in V$ в графе U существует главная цепь записи y.*

Доказательство. В силу свойства рефлексивности отношения $\succeq$ $O(y, \succeq) \neq \emptyset$. Так как ПИГ U решает ЗИП I, то согласно теореме 1 $\bigvee\limits_{\alpha \in L_U(y)} \varphi_\alpha = \chi_{y,\succeq} = K_y$. Следовательно, множество $L_U(y)$ есть характерное множество, и согласно теореме 14 в графе U существует главная цепь множества $L_U(y)$. Что и требовалось доказать.

4.2 Включающий поиск

Среди задач поиска, в которых в качестве отношения поиска выступают отношения частичного порядка, наверное, наибо-

лее распространенными являются задачи включающего поиска (set-inclusion search problem).

Приведем наиболее распространенную интерпретацию задачи включающего поиска.

Предположим, что мы осуществляем поиск в дескрипторных автоматизированных информационно-поисковых системах, то есть информационный массив состоит из документов, каждый документ описывается множеством дескрипторов (ключевых слов), запрос задает некоторую совокупность дескрипторов, и необходимо перечислить в информационном массиве все документы, содержащие в своем описании все дескрипторы, входящие в запрос. Занумеруем некоторым образом множество всех дескрипторов (*тезаурус*). Каждому документу сопоставим запись, представляющую собой булев вектор длины, равной мощности тезауруса, в i-ой компоненте которого стоит 0 (ноль) в том и только в том случае, когда i-ый дескриптор входит в описание данного документа. Запросы будем описывать аналогичными векторами, то есть в i-ой компоненте запроса будет стоять 0, если i-ый дескриптор входит в запрос. Теперь если в качестве отношения поиска взять отношение $\overset{b}{\succeq}$, определяемое следующим соотношением

$$(x_1, \ldots, x_n) \overset{b}{\succeq} (y_1, \ldots, y_n) \Longleftrightarrow x_i \geq y_i,\ i = \overline{1,n},\ x_i, y_i \in \{0,1\},$$

то получившаяся задача включающего поиска будет полностью соответствовать исходной. В самом деле, каждая компонента запроса должна быть не меньше соответствующей компоненты записи, или если в i-ой компоненте запроса стоит 0 (то есть i-ый дескриптор входит в запрос), то в i-ой компоненте записи тоже должен стоять 0 (то есть запись должна содержать i-ый дескриптор).

В общем случае включающий поиск встречается всегда, когда элементы информационного массива задаются множеством свойств (в частности, в дескрипторных автоматизированных информационно-поисковых системах — свойствами наличия дескриптора в описании документа) и необходимо перечислить в этом массиве элементы с определенным набором свойств, задаваемым запросом.

Хотя задачи включающего поиска имеют широкое примене-

ние, исследование оценок сложности включающего поиска не проводилось.

Для этих задач удалось получить нижнюю оценку в два раза лучшую, чем мощностная нижняя оценка. Кроме того, было доказано существование задач, для которых эта нижняя оценка асимптотически не улучшаема. Эти и некоторые другие результаты приводятся в данном разделе.

Рассмотрим следующий тип задач поиска:

$$S_{bool} = \langle B^n, B^n, \overset{b}{\succeq}, \sigma, \mathbf{P} \rangle,$$

где B^n — *единичный n-мерный куб*, то есть

$$B^n = \{(\alpha_1, \ldots, \alpha_n) : \ \alpha_i \in \{0,1\} \ (i = \overline{1,n})\};$$

$\overset{b}{\succeq}$ — отношение поиска на $B^n \times B^n$, определяемое следующим соотношением $(x_1, \ldots, x_n) \overset{b}{\succeq} (y_1, \ldots, y_n) \Longleftrightarrow x_i \geq y_i, \ i = \overline{1,n}$; σ — алгебра подмножеств B^n, представляющая собой множество всех подмножеств B^n; $\mathbf{P}$ — равномерная вероятностная мера на σ, то есть такая мера, что для любого $x \in B^n \ \ \mathbf{P}(x) = 1/2^n$ и любого $A \subseteq B^n \ \ \mathbf{P}(A) = |A|/2^n$.

Задачи поиска, принадлежащие данному типу, есть разновидность задач, именуемых в литературе задачами включающего поиска (см., например, [27]), поэтому тип S_{bool} мы назовем *типом включающего поиска*, а задачи, принадлежащие этому типу, — *задачами включающего поиска*.

Напомним, что в соответствии с терминологией [5] *гранью* единичного n-мерного куба B^n называется множество

$$\{(\alpha_1, \ldots, \alpha_n) \in B^n : \alpha_{i_1} = \sigma_1, \ldots, \alpha_{i_k} = \sigma_k\},$$

при этом число $n - k$ называется *размерностью* этой грани.

Весом набора $(\alpha_1, \ldots, \alpha_n) \in B^n$ называют число его координат, равных 1. Множество вершин куба, имеющих вес k, называется *k-м слоем куба B^n* и обозначается B^n_k. *Номером набора* $\alpha = (\alpha_1, \ldots, \alpha_n) \in B^n$ назовем число $||\alpha|| = \sum_{i=1}^{n} 2^{n-i} \cdot \alpha_n$. Будем считать, что наборы в слое куба упорядочены в порядке убывания их номеров. Множество, состоящее из m первых наборов k-го слоя куба B^n, будем называть *начальным отрезком* длины m этого слоя.

Формула $x_{i_1}^{\sigma_1}\&x_{i_2}^{\sigma_2}\&\ldots\&x_{i_r}^{\sigma_r}$, где & - знак конъюнкции, $\sigma_k \in \{0,1\}$, $x_{i_k}^0 = \overline{x}_{i_k}$, $x_{i_k}^1 = x_{i_k}$, $i_k \in \{1,2,\ldots,n\}$, $k = 1,2,\ldots,r$ ($r \geq 1$ и $n \geq 1$), называется *конъюнкцией над множеством переменных* $X^n = \{x_1, x_2, \ldots, x_n\}$, а число r — длиной этой конъюнкции. Если $x_{i_j} \neq x_{i_k}$ при $j \neq k$, то конъюнкция называется *элементарной*. Элементарная конъюнкция называется *монотонной*, если она не содержит отрицаний переменных. Множество элементарных монотонных конъюнкций от n переменных будем обозначать через $\mathcal{K}^n$. Функцию тождественная единица будем считать элементарной монотонной конъюнкцией длины 0, то есть будем считать, что она входит в множество $\mathcal{K}^n$, кроме того добавим тождественную единицу и в множество переменных X^n.

Будем говорить, что две элементарные монотонные конъюнкции *пересекаются по переменным*, если в формулах, описывающих эти конъюнкции, встречаются одинаковые переменные.

Функция алгебры логики $f(x_1, \ldots, x_n)$ называется *монотонной*, если для любых двух наборов α и β из B^n таких, что $\alpha \overset{b}{\succeq} \beta$, имеет место неравенство $f(\alpha) \geq f(\beta)$. Дизъюнкция элементарных монотонных конъюнкций есть монотонная функция. Множество монотонных булевых функций от n переменных будем обозначать через $\mathcal{M}^n$.

Рассмотрим произвольную запись $y \in B^n$. Нетрудно заметить, что если $\{i_1, \ldots, i_k\}$ есть множество номеров координат вектора y, которые равны 1, то характеристическая функция записи является элементарной монотонной конъюнкцией

$$\chi_{y,\overset{b}{\succeq}}(x_1, \ldots, x_n) = x_{i_1}\&x_{i_2}\&\ldots\&x_{i_k}.$$

Далее часто знак конъюнкции & в формулах мы будем опускать.

Таким образом, согласно теореме 1 граф, решающий некоторую задачу включающего поиска, представляет собой многополюсник, реализующий некоторую систему элементарных монотонных конъюнкций. Отсюда согласно теореме 2 следует, что каждое из базовых множеств $\langle \mathcal{M}^n, \emptyset \rangle$, $\langle \mathcal{K}^n, \emptyset \rangle$ и $\langle X^n, \emptyset \rangle$ является полным для типа S_{bool}.

Следовательно, тип S_{bool} полностью удовлетворяет условиям теоремы 3, и, значит, для любой ЗИП I типа S_{bool} существует оптимальный ПИГ над любым из базовых множеств $\langle \mathcal{M}^n, \emptyset \rangle$, $\langle \mathcal{K}^n, \emptyset \rangle$ и $\langle X^n, \emptyset \rangle$.

Упражнения

4.1. Пусть $V = \{y_1, y_2, \ldots, y_k\} \subseteq B^n$ и число единиц в наборе y_i равно t_i $(i = 1, 2, \ldots, k)$. Приведите мощностную нижнюю оценку для задачи включающего поиска $I = \langle B^n, V, \overset{b}{\succeq} \rangle$.

4.3 Нижняя оценка сложности включающего поиска

В данном пункте нам понадобится теорема Краскала–Катоны, приведенная, например, в работах [49, 51, 52, 54]. Переформулируем ее в удобных для нас терминах.

Теорема 15 (Краскала–Катоны). *Пусть B^n_m — m-й слой куба B^n и $A \subseteq B^n_m$. Пусть $R(A) = \{\alpha \in B^n_{m+1} : (\exists \beta \in A : \alpha \overset{b}{\succeq} \beta)\}$. Тогда минимум $|R(A)|$ достигается, если A — начальный отрезок слоя B^n_m, причем в этом случае $R(A)$ — начальный отрезок слоя B^n_{m+1}.*

Если $A \in B^n$, то обозначим $O(A, \overset{b}{\succeq}) = \bigcup\limits_{y \in A} O(y, \overset{b}{\succeq})$.

Приведем одно простое следствие.

Следствие 8 (теоремы Краскала–Катоны). *Пусть $A \subseteq B^n_m$, тогда $|O(A, \overset{b}{\succeq})|$ достигает минимума, если A — начальный отрезок слоя B^n_m.*

Лемма 9. *Пусть $I = \langle B^n, V, \overset{b}{\succeq} \rangle$ — ЗИП типа S_{bool}. Пусть U — решающий ЗИП I ПИГ над базовым множеством $\mathcal{F} = \langle \mathcal{M}^n, \emptyset \rangle$, где $\mathcal{M}^n$ — множество монотонных булевых функций от n переменных. Тогда существует такое разбиение библиотеки V на непересекающиеся части, что*

- *каждая часть содержит записи одного веса, и этот вес назовем весом части;*
- *каждой части веса $m > 0$ и мощности t можно поставить в соответствие такую совокупность ребер графа U, что сумма сложностей ребер из этой совокупности не меньше чем $t \cdot 2^{1-m}$, причем образы различных частей при этом соответствии не пересекаются.*

Доказательство. Пусть $V = \{y_1, \ldots, y_k\}$.

Отметим, что если $y_i \in V$ — запись веса m, то $|O(y_i, \overset{b}{\succeq})| = 2^{n-m}$, так как $O(y_i, \overset{b}{\succeq})$ — $(n-m)$-мерная грань куба B^n.

Обозначим через A_m^l начальный отрезок длины l m-го слоя куба B^n. Легко видеть, что если $l \leq n - m + 1$, то

$$|O(A_m^l, \overset{b}{\succeq})| = 2^{n-m+1} \cdot (1 - 2^{-l}).$$

Так как для любой записи $y_i \in V$ выполняется $O(y_i, \overset{b}{\succeq}) \neq \emptyset$, то согласно следствию 7 из теоремы 14 о существовании главных цепей для любой записи $y_i \in V$ в графе U существует главная цепь записи y_i, которую обозначим через C_i. Лист из $L_U(y_i)$, в который ведет эта цепь, будем обозначать α_i $(i = \overline{1,k})$.

Будем строить требуемое в лемме разбиение, последовательно добавляя к нему новые части разбиения, пока они не покроют все множество. Причем будем считать, что в начальный момент разбиение не содержит ни одной части, то есть представляет собой пустое множество.

Рассмотрим произвольную запись $y_i \in V$ $(i \in \{\overline{1,k}\})$, которая на момент рассмотрения не попала ни в какую часть разбиения. Пусть ее вес равен $m > 0$. Пусть $\beta_i \neq \alpha_i$ — ближайшая к α_i вершина в цепи C_i, через которую также проходит главная цепь какой-то другой записи. Такая вершина β_i обязательно существует, в крайнем случае β_i — корень графа U, так как все главные цепи начинаются в корне. Пусть γ_i — вершина графа, в которую ведет ребро цепи C_i, исходящее из β_i.

Разберем возможные случаи.

Случай 1. Пусть β_i — корень графа U. Тогда ребро (β_i, γ_i) имеет сложность 1. Мы заведем новую часть разбиения, состоящую из одной записи y_i, и сопоставим этой части ребро (β_i, γ_i).

Случай 2. Предположим теперь, что β_i не является корнем графа U, но совпадает с каким-либо листом α_j, где $j \in \{\overline{1,k}\} \setminus \{i\}$. Легко заметить, что так как C_i и C_j — главные цепи, то вес записи y_j меньше веса записи y_i и, следовательно, сложность ребра (β_i, γ_i) не меньше чем 2^{1-m}. Мы заведем новую часть разбиения, состоящую из одной записи y_i, и сопоставим этой части ребро (β_i, γ_i).

Случай 3. Пусть теперь β_i не является корнем графа U и не совпадает ни с каким листом α_j. Пусть $C_{j_1}, \ldots, C_{j_q}$ — множество главных цепей записей, проходящих через вершину β_i.

Случай 3.1 Пусть среди записей $y_{j_1}, \ldots, y_{j_q}$ существует запись y_{j_s}, вес которой меньше m. Тогда сложность вершины β_i не меньше чем $\mathbf{P}(O(y_{j_s}, \overset{b}{\succeq})) = |O(y_{j_s}, \overset{b}{\succeq})|/2^n \geq 2^{1-m}$ и, следовательно, сложность ребра (β_i, γ_i) не меньше чем 2^{1-m}. Мы заведем новую часть разбиения, состоящую из одной записи y_i, и сопоставим этой части ребро (β_i, γ_i).

Случай 3.2 Предположим, что среди записей $y_{j_1}, \ldots, y_{j_q}$ не существует записи, вес которой меньше m. Пусть в множестве $\{y_{j_1}, \ldots, y_{j_q}\}$ имеется t записей веса m таких, что в главных цепях этих записей между вершиной β_i и листом, в котором заканчивается цепь, нет вершин пересечения с другими главными цепями записей. Понятно, что $t \geq 1$. Обозначим это множество записей через V' и объявим его новой частью разбиения. Понятно, что все записи из V' до данного момента еще не рассматривались, так как в противном случае запись y_i уже оказалась бы в некоторой части разбиения. Понятно также, что для каждой записи $y_s \in V'$ β_i — ближайшая к α_s вершина в цепи C_s, через которую проходит главная цепь какой-то другой записи, причем, если γ_s — вершина, в которую ведет ребро цепи C_s, исходящее из β_i, то ребро (β_i, γ_s) не принадлежит никакой другой цепи, кроме C_s. Сопоставим V' совокупность ребер, состоящую из t ребер, исходящих из вершины β_i и принадлежащих главным цепям записей из V', и t' ребер, входящих в вершину β_i и принадлежащих главным цепям записей из V' $(t' \leq t)$. Заметим, что сумма сложностей последних t' ребер не меньше чем $\mathbf{P}(O(V', \overset{b}{\succeq}))$, и сложность вершины β не меньше чем $\mathbf{P}(O(V', \overset{b}{\succeq}))$, следовательно, сумма сложностей

ребер совокупности, сопоставленной части V', не меньше чем $(t+1)\cdot\mathbf{P}(O(V',\overset{b}{\succeq})) = 2^{-n}(t+1)\cdot|O(V',\overset{b}{\succeq})|$. Согласно следствию теоремы Краскала–Катоны $|O(V',\overset{b}{\succeq})| \geq |O(A_m^t,\overset{b}{\succeq})|$. Следовательно, если $t \leq n-m+1$, то

$$\begin{aligned} 2^{-n}(t+1)\cdot|O(V',\overset{b}{\succeq})| &\geq 2^{-n}(t+1)\cdot|O(A_m^t,\overset{b}{\succeq})| = \\ &= 2^{-n}(t+1)(1-2^{-t})2^{n-m+1} = \\ &= t\cdot 2^{1-m} + 2^{1-m}(1-(t+1)2^{-t}) \geq \\ &\geq t\cdot 2^{1-m}. \end{aligned}$$

При $t > n-m+1$, поскольку для любой записи $y \in V'$ справедливо $O(y,\overset{b}{\succeq}) \setminus O(V'\setminus y,\overset{b}{\succeq}) \neq \emptyset$, то

$$\begin{aligned} 2^{-n}(t+1)\cdot|O(V',\overset{b}{\succeq})| &\geq 2^{-n}(t+1)\times \\ &\quad \times(|O(A_m^{n-m+1},\overset{b}{\succeq})| + t - n + m - 1) \geq \\ &\geq (t+1)\cdot 2^{1-m} > t\cdot 2^{1-m}. \end{aligned}$$

Теперь осталось проверить, что указанное соответствие различным частям разбиения сопоставляет различные непересекающиеся множества ребер.

Отметим, что по ходу построения соответствия мы для каждой записи $y_i \in V$ ($i \in \{\overline{1,k}\}$), нашли свои вершины β_i и γ_i, описанные выше.

Докажем одно полезное свойство.

Свойство 6. *Если β_i и γ_i — описанные выше вершины, соответствующие записи y_i, то ребро (β_i,γ_i) принадлежит только совокупности ребер, соответствующей части разбиения, содержащей запись y_i.*

Доказательство. Так как для любой записи $y_i \in V$ совокупность ребер, связанная с записью y_i, содержит только ребра либо входящие в β_i, либо исходящие из β_i, то ребро (β_i,γ_i) может попасть в другую совокупность ребер только в том случае, если вершина γ_i совпадает с какой-то другой вершиной β_s. Итак пусть существует $s \in \{\overline{1,k}\}\setminus\{i\}$ такое, что $\gamma_i = \beta_s$. Тогда γ_i принадлежит цепи C_s, отличной от C_i, а так как γ_i в цепи C_i ближе к листу α_i, то согласно определению вершины β_i вершина γ_i должна совпадать с вершиной α_i, а это сразу означает,

что каждая из записей y_s, вершина β_s которой совпадает с γ_i, подпадает под случай 2, и, значит, содержащая y_s часть состоит из одной записи y_s и ей соответствует одно единственное ребро (β_s, γ_s), не совпадающее с (β_i, γ_i).

Тем самым свойство 6 доказано.

Возьмем произвольную запись $y_i \in V$. Покажем, что совокупность ребер, соответствующая части разбиения, содержащей запись y_i, не пересекается с образами никаких других частей разбиения.

Рассмотрим возможные варианты.

Вариант 1. Пусть существует $s \in \{\overline{1,k}\} \setminus \{i\}$ такое, что $\beta_i = \gamma_s$.

Тогда по тем же соображениям, которые описаны в доказательстве свойства 6, запись y_i подпадает под случай 2, и, значит, содержащая y_i часть разбиения состоит из одной записи y_i и ей соответствует одно единственное ребро (β_i, γ_i), которое согласно свойству 6 принадлежит только совокупности ребер, соответствующей части разбиения, содержащей запись y_i.

Вариант 2. Пусть для любого $s \in \{\overline{1,k}\} \setminus \{i\}$ выполняется $\beta_i \neq \gamma_s$.

Тогда возможны следующие подварианты.

Вариант 2.1 Для любого $s \in \{\overline{1,k}\} \setminus \{i\}$ вершина β_i не совпадает с вершиной β_s.

Отсюда следует, что все ребра, входящие в β_i, не принадлежат никаким совокупностям ребер, кроме, быть может, совокупности ребер, соответствующей части разбиения, содержащей запись y_i. Кроме того, отсюда следует, что либо запись y_i подпадает под случай 3.1 алгоритма построения соответствия, и тогда части разбиения, состоящей из одной записи y_i, сопоставляется одно единственное ребро (β_i, γ_i), либо запись y_i подпадает под случай 3.2 алгоритма построения соответствия, где параметр t принимает значение 1, и тогда части разбиения, состоящей из одной записи y_i, сопоставляется два ребра: ребро (β_i, γ_i) и ребро, входящее в β_i и принадлежащее цепи C_i. Следовательно, согласно приведенному выше замечанию и свойству 6, и в этом варианте совокупность ребер, соответствующая части разбиения, содержащей запись y_i, не пересекается с образами

никаких других частей разбиения.

Вариант 2.2 Существует $s \in \{\overline{1,k}\} \setminus \{i\}$ такое, что $\beta_i = \beta_s$.

Пусть $V' = \{y_{j_1}, \ldots, y_{j_l}\}$ — подмножество записей из V таких, что $\beta_{j_r} = \beta_i$ $(r = \overline{1,l})$. Так как мы находимся в условиях варианта 2, то все ребра, входящие в β_i, не принадлежат никаким совокупностям ребер, кроме, быть может, совокупностей ребер, соответствующих частям разбиения, содержащим записи из V'.

Возможны следующие два подварианта.

Вариант 2.2.1 В множестве V' есть запись с весом меньшим, чем вес записи y_i.

Тогда запись y_i подпадает под случай 3.1 алгоритма построения соответствия, и части разбиения, состоящей из одной записи y_i, сопоставляется одно единственное ребро (β_i, γ_i), которое согласно свойству 6 принадлежит только совокупности ребер, соответствующей части разбиения, содержащей запись y_i.

Вариант 2.2.2 В множестве V' нет записей с весом меньшим, чем вес записи y_i.

Тогда запись y_i подпадает под случай 3.2 алгоритма построения соответствия. Пусть $V'' \subset V'$ — множество записей, вес которых совпадает с весом y_i. Согласно алгоритму построения соответствия множество V'' образует одну часть разбиения, которой сопоставляется совокупность ребер, инцидентных вершине β_i и принадлежащих главным цепям записей из V''. Согласно свойству 6 и сделанному в варианте 2.2 замечанию, эти ребра не принадлежат образам других частей разбиения.

Таким образом, мы перебрали все возможные случаи и показали, что совокупность ребер, соответствующая части разбиения, содержащей запись y_i, не пересекается с образами никаких других частей разбиения.

В силу произвольности выбора записи y_i это доказывает, что образы различных частей разбиения взаимно не пересекаются.

Тем самым лемма 9 доказана.

Опираясь на лемму 9, мы можем получить следующую нижнюю оценку сложности включающего поиска.

Теорема 16 (нижняя оценка). *Пусть* $I = \langle B^n, V, \overset{b}{\succeq} \rangle$ *— ЗИП типа* S_{bool}. *Пусть базовое множество имеет вид* $\mathcal{F} = \langle \mathcal{M}^n, \emptyset \rangle$,

где $\mathcal{M}^n$ — множество монотонных булевых функций. Тогда

$$T(I,\mathcal{F}) \geq 2 \cdot \sum_{y \in V} \mathbf{P}(O(y, \overset{b}{\succeq})) - t_0,$$

где t_0 — число записей веса 0 в библиотеке V ($t_0 \in \{0,1\}$).

Доказательство. Обозначим через t_i — число записей веса i в библиотеке V ($i = \overline{1,n}$).

Возьмем произвольный ПИГ U над базовым множеством $\mathcal{F} = \langle \mathcal{M}^n, \emptyset \rangle$, решающий ЗИП I. (Такой ПИГ обязательно существует, так как $\mathcal{F}$ полно для типа S_{bool}.)

Согласно лемме 9 библиотеку V можно разбить на такие непересекающиеся части, что каждая часть содержит записи одного веса, и каждой части веса $i > 0$ и мощности t сопоставляется такая совокупность ребер графа U, что сумма сложностей ребер из этой совокупности не меньше чем $t \cdot 2^{1-m}$, причем образы различных частей при этом соответствии не пересекаются. Отсюда следует, что

$$T(U) \geq \sum_{i=1}^{n} t_i 2^{1-i}.$$

Теперь предположим, что в библиотеке V есть запись $y_0 = (0, \ldots, 0)$ веса 0. Обозначим через C_0 главную цепь, ведущую из корня в множество $L_U(y_0)$ (согласно теореме 14 такая цепь существует). Пусть α_0 — лист, в котором заканчивается цепь C_0, а ребро (β_0, α_0) — последнее ребро цепи C_0. Так как запись y_0 удовлетворяет всем запросам из B^n, то все запросы проходят через ребро (β_0, α_0), и, значит, сложность этого ребра равна 1. Остается заметить, что ребро (β_0, α_0) не принадлежит ни одной совокупности ребер, соответствующей части разбиения веса, большего 0.

Таким образом,

$$T(U) \geq t_0 + \sum_{i=1}^{n} t_i 2^{1-i} = 2 \cdot \sum_{y \in V} \mathbf{P}(O(y, \overset{b}{\succeq})) - t_0.$$

Отсюда в силу произвольности ПИГ U вытекает утверждение теоремы. Что и требовалось доказать.

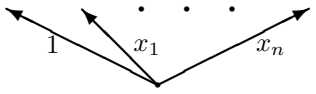


Рис. 4.1: Универсальный многополюсник для множества $\mathcal{K}_1^n$

Замечание. Если мы находимся в условиях теоремы 16, то нижняя оценка, получаемая из этой теоремы, практически в два раза лучше мощностной нижней оценки.

4.4 Верхняя оценка сложности включающего поиска

Теорема 17 (верхняя оценка). *Пусть $I = \langle B^n, V, \overset{b}{\succeq}\rangle$ — ЗИП типа S_{bool}. Пусть базовое множество имеет вид $\mathcal{F} = \langle \mathcal{K}^n, \emptyset\rangle$, где $\mathcal{K}^n$ — множество монотонных элементарных конъюнкций. Пусть $|V| = k$ и m — такое число, что $2\binom{n}{m} < k \leq 2\binom{n}{m+1}$. Тогда*

$$T(I, \mathcal{F}) \leq 1 + \sum_{i=1}^{m} 2^{1-i}\binom{n}{i} + 2^{-m}k.$$

Доказательство. Обозначим через $\mathcal{K}_l^n$ — множество всех монотонных элементарных конъюнкций, содержащих не более l переменных (или другими словами, имеющих длину не более l).

Построим по индукции информационный граф U_l^n, реализующий как функции фильтров каких-либо вершин все функции из $\mathcal{K}_l^n$.

Базис индукции. $l = 1$. $\mathcal{K}_1^n = \{1, x_1, \ldots, x_n\}$.

U_1^n будет иметь вид, изображенный на рисунке 4.1, где жирной точкой изображен корень графа, а на концах ребер реализуются функции из $\mathcal{K}_1^n$.

Шаг индукции. Пусть U_{l-1}^n — граф, реализующий все функции из $\mathcal{K}_{l-1}^n$. Граф U_l^n будем строить, добавляя к U_{l-1}^n вершины и ребра, чтобы реализовать функции из $\mathcal{K}_l^n \setminus \mathcal{K}_{l-1}^n$, следующим образом. Возьмем произвольную монотонную элементар-

ную конъюнкцию длины l $x_{i_1}\&\cdots\&x_{i_l}$. В графе U^n_{l-1} найдем вершину, на которой реализуется функция $x_{i_1}\&\cdots\&x_{i_{l-1}}$ и выпустим из нее ребро, которому припишем переменную x_{i_l}. На вершине, в которую ведет это ребро, будет реализовываться как функция фильтра функция $x_{i_1}\&\cdots\&x_{i_l}$. Перебрав все функции из $\mathcal{K}^n_l\setminus\mathcal{K}^n_{l-1}$ и проделав для каждой эту операцию, получим требуемый граф U^n_l, реализующий все функции из $\mathcal{K}^n_l$.

Отметим, что на первом ярусе графа U^n_l находится $n+1=\binom{n}{1}+1$ ребер, и сложность каждого из них равна 1 (первый ярус — это ребра, исходящие из корня), а на i-ом ярусе ($i\geq 2$) находится $\binom{n}{i}$ ребер, и сложность каждого из них равна 2^{1-i} (i-ый ярус — это ребра, исходящие из концов ребер $(i-1)$-го яруса). Таким образом, сложность графа U^n_l равна

$$T(U^n_l)=1+\sum_{i=1}^{l}\binom{n}{i}2^{1-i}.$$

Вернемся теперь к нашей задаче I.

Для любой записи $y\in V$ характеристическая функция этой записи есть некоторая монотонная элементарная конъюнкция, которую будем обозначать K_y.

Информационный граф U, решающий задачу I, будем строить следующим образом. Возьмем граф U^n_m, описанный выше. Возьмем произвольную запись $y\in V$. Возможны два случая.

1. Характеристическая функция этой записи K_y имеет длину, не превышающую m. Тогда в графе U^n_m находим вершину, на которой реализуется функция K_y, объявляем эту вершину листом и приписываем ей запись y.

2. Длина элементарной конъюнкции K_y больше чем m. Пусть $K_y=x_{i_1}\&\cdots\&x_{i_l}$, где $l>m$. Тогда найдем в графе U^n_m вершину, на которой реализуется функция $x_{i_1}\&\cdots\&x_{i_m}$, выпустим из этой вершины ребро, припишем этому ребру функцию $x_{i_{m+1}}\&\cdots\&x_{i_l}$, объявим конец этого ребра листом и припишем ему запись y.

Проделав эту операцию для каждой записи $y\in V$, мы получим граф U, который, как нетрудно заметить, согласно теореме 1 решает задачу I.

Оценим сложность графа U. Граф U содержит в себе как подграф граф U^n_m. В худшем случае, когда для каждой запи-

си реализуется случай 2, мы добавим к графу U_m^n k ребер, каждое из которых исходит из некоторой вершины, на которой реализуется элементарная конъюнкция длины m. Следовательно, сложность каждого из этих k ребер будет равна 2^{-m}. Таким образом,

$$T(I,\mathcal{F}) \le T(U) \le T(U_m^n) + 2^{-m}k = 1 + \sum_{i=1}^{m} 2^{1-i}\binom{n}{i} + 2^{-m}k.$$

Тем самым теорема доказана.

4.5 Асимптотика функции Шеннона сложности включающего поиска

Если k — натуральное число, то обозначим

$$\mathcal{I}(k, S_{bool}) = \{I = \langle X, V, \rho\rangle \in S_{bool} : |V| = k\}.$$

Рассмотрим функцию Шеннона, характеризующую сложность класса ЗИП $\mathcal{I}(k, S_{bool})$

$$\mathcal{T}(k, S_{bool}, \mathcal{F}) = \sup_{I \in \mathcal{I}(k, S_{bool})} T(I, \mathcal{F}).$$

Теорема 17 позволяет нам показать, что существуют библиотеки, для которых нижняя оценка теоремы 16 асимптотически не улучшаема.

Теорема 18. (Асимптотика функции Шеннона) *Если $m(n) \to \infty$ при $n \to \infty$, $m = \bar{o}(n)$, $k(n)$ такое, что $\binom{n}{m-1} = \bar{o}(k)$ и $k \le \binom{n}{m}$, и базовое множество имеет вид $\mathcal{F} = \langle F, \emptyset\rangle$, где $F \subseteq \mathcal{M}^n$ и $\mathcal{K}^n \subseteq F$, то при $n \to \infty$*

$$\mathcal{T}(k, S_{bool}, \mathcal{F}) \sim 2^{1-m}k.$$

Доказательство. Рассмотрим библиотеку V, являющуюся k-элементным подмножеством m-го слоя куба B^n, то есть $V \subseteq B_m^n$ и $|V| = k$. Рассмотрим связанную с этой библиотекой ЗИП $I = \langle B^n, V, \overset{b}{\succeq}\rangle$. Согласно теореме 16

$$T(I,\mathcal{F}) \ge 2 \cdot \sum_{y \in V} \mathbf{P}(O(y, \overset{b}{\succeq})) = 2^{1-m}k.$$

Согласно теореме 17

$$T(I,\mathcal{F}) \le 1 + \sum_{i=1}^{m-1} 2^{1-i}\binom{n}{i} + 2^{1-m}k = 2^{1-m}k(1+\bar{o}(1)).$$

Осталось заметить, что последние два неравенства доказывают утверждение теоремы 18.

4.6 Асимптотика функции Шеннона включающего поиска в классе древовидных ИГ

Нижняя оценка включающего поиска, приведенная в теореме 16, в два раза лучше мощностной нижней оценки. В случае, когда базовое множество состоит из множества переменных X^n, удается найти такие задачи, для которых в классе древовидных схем была получена нижняя оценка, большая по порядку, чем мощностная нижняя оценка, причем на этих задачах достигается асимптотика фукции Шеннона сложности. Результат данного раздела опубликован в [10].

Пусть $\mathcal{U}_d(I,\mathcal{F})$ — множество всех ИГ над базовым множеством $\mathcal{F}$, решающих ЗИП I и являющихся древовидными. Обозначим

$$T_d(I,\mathcal{F}) = \inf\{T(U) : U \in \mathcal{U}_d(I_k,\mathcal{F})\},$$

$$\mathcal{T}_d(k,S_{bool},\mathcal{F}) = \sup_{I\in\mathcal{I}(k,S_{bool})} T_d(I,\mathcal{F}).$$

Теорема 19. *Если базовое множество имеет вид* $\mathcal{F} = \langle X^n, \emptyset\rangle$ *и* $k(n)$, $m(n)$ *и* $\alpha(n)$ *такие числа, зависящие от* n, *что*

$$k(n) \sim \binom{n}{m-1}\alpha(n),\ m(n) \ge 1,\ m(n) = \bar{o}(\log_2 n/\log_2\log_2 n),$$

$$\alpha(n) \to \infty,\ (\log_2 n - \log_2\alpha(n))/(m(n)\log_2(m(n)+1)) \to \infty$$

при $n \to \infty$, *то при* $n \to \infty$

$$\mathcal{T}_d(k,S_{bool},\mathcal{F}) \sim 2^{2-m}k.$$

Доказательство. **Нижняя оценка.**

Пусть $p(n)$ — такое, что $p^m \geq k > (p-1)^m$. Оценим величину p/n. Для этого рассмотрим два случая: когда $m = const$, и $m \to \infty$ при $n \to \infty$.

В первом случае получим

$$\begin{aligned} \frac{p}{n} &\sim \frac{k^{1/m}}{n} \sim n^{-1}\left(\frac{\alpha n(n-1)\cdots(n-m+2)}{(m-1)!}\right)^{1/m} \sim \\ &\sim \left(\frac{n^{m-1}\alpha}{n^m(m-1)!}\right)^{1/m} = \left(\frac{\alpha}{n(m-1)!}\right)^{1/m} = \bar{o}(1), \quad (4.1) \end{aligned}$$

поскольку, как легко видеть, $\alpha(n) = \bar{o}(n)$.

Во втором случае, используя формулу Стирлинга (см., например, [5, стр. 282]), можем получить аналогичный результат

$$\begin{aligned} \frac{p}{n} &\sim \frac{k^{1/m}}{n} \sim n^{-1}\left(\frac{\alpha}{\sqrt{2\pi(m-1)}}\left(\frac{n\,e}{m-1}\right)^{m-1}\right)^{1/m} = \\ &= \left(\frac{n^{m-1}\alpha}{n^m}\right)^{1/m} \cdot \left(\frac{e}{m-1}\right)^{\frac{m-1}{m}} \cdot \\ &\quad \cdot(2\pi(m-1))^{-1/2m} = \bar{o}(1). \quad (4.2) \end{aligned}$$

Пусть $r(n)$ такое, что

$$p\,m + p\sum_{i=1}^{r} m^i \leq n. \quad (4.3)$$

Договоримся обозначать запись $y \in B^n$ через ее характеристическую функцию, то есть представлять ее в виде конъюнкции переменных, номера которых совпадают с номерами разрядов вектора y, в которых стоят 1.

Разобьем все множество переменных X^n на $m+r+1$ взаимно непересекающиеся части так, что в первых m частях находится по p переменных, в $(m+i)$-й части ($i = \overline{1,r}$) находится $p\,m^i$ переменных, а в $(m+r+1)$-й части все оставшиеся переменные. В соответствии с неравенством (4.3) такое разбиение возможно. Переменные из i-й части X^n ($i = \overline{1, m+r+1}$) будем обозначать через x_q^i, то есть верхний индекс переменной будет говорить какой части принадлежит переменная.

Рассмотрим библиотеку

$$V = \{ \quad x^1_{q_1} x^2_{q_2} \dots x^m_{q_m} x^{m+1}_{f_1} \dots x^{m+r}_{f_r} : q_j \in \{\overline{1,p}\}, j = \overline{1,m};$$
$$f_j(q_1, \dots, q_m) = \sum_{i=1}^{m} q_i \cdot i^{j-1}, j = \overline{1,r}\}.$$

Поскольку каждая запись $x^1_{q_1} x^2_{q_2} \dots x^m_{q_m} x^{m+1}_{f_1} \dots x^{m+r}_{f_r}$ полностью определяется набором $(q_1, \dots, q_m)$, то мощность библиотеки V равна $|V| = p^m \geq k$. Отметим также, что для любого $j \in \{\overline{1,r}\}$

$$f_j = \sum_{i=1}^{m} q_i \cdot i^{j-1} \leq p \cdot m \cdot m^{j-1} = p\, m^j.$$

Покажем, что любая пара различных векторов из V имеет не более чем $m-1$ одинаковую переменную. Возьмем произвольную пару векторов из V:

$$x^1_{q_1} x^2_{q_2} \dots x^m_{q_m} x^{m+1}_{f_1} \dots x^{m+r}_{f_r},$$

$$x^1_{q'_1} x^2_{q'_2} \dots x^m_{q'_m} x^{m+1}_{f'_1} \dots x^{m+r}_{f'_r}.$$

Пусть $q_{i_l} \neq q'_{i_l}$, $i_l \in \{\overline{1,m}\}$, $l = \overline{1,s}$, и $q_j = q'_j$ при условии, что $j \in \{\overline{1,m}\} \setminus \{i_l : \ l = \overline{1,s}\}$. Так как вектора различные, то $s > 0$.

Пусть $f_{j_l} = f'_{j_l}$, $j_l \in \{\overline{1,r}\}$, $l = \overline{1,t}$, и $f_j \neq f'_j$ при условии, что $j \in \{\overline{1,r}\} \setminus \{j_l : \ l = \overline{1,t}\}$. Покажем, что $t < s$.

Предположим, что это не так, то есть $t \geq s$.

Рассмотрим систему уравнений

$$\begin{cases} f_{j_1} = f'_{j_1}, \\ \quad \dots\dots \\ f_{j_t} = f'_{j_t}. \end{cases}$$

Взяв в этой системе первые s уравнений, получим

$$\begin{cases} \sum\limits_{i=1}^{m} q_i \cdot i^{j_1-1} = \sum\limits_{i=1}^{m} q'_i \cdot i^{j_1-1}, \\ \quad \dots\dots\dots\dots\dots\dots\dots\dots \\ \sum\limits_{i=1}^{m} q_i \cdot i^{j_s-1} = \sum\limits_{i=1}^{m} q'_i \cdot i^{j_s-1}. \end{cases}$$

$$\begin{cases} \sum\limits_{l=1}^{s}(q_{i_l}-q'_{i_l})\cdot {i_l}^{j_1-1}=0, \\ \quad\cdots\cdots\cdots\cdots\cdots\cdots \\ \sum\limits_{l=1}^{s}(q_{i_l}-q'_{i_l})\cdot {i_l}^{j_s-1}=0. \end{cases}$$

Определитель этой системы

$$\begin{vmatrix} {i_1}^{j_1-1} & {i_2}^{j_1-1} & \dots & {i_s}^{j_1-1} \\ {i_1}^{j_2-1} & {i_2}^{j_2-1} & \dots & {i_s}^{j_2-1} \\ \vdots & \vdots & \ddots & \vdots \\ {i_1}^{j_s-1} & {i_2}^{j_s-1} & \dots & {i_s}^{j_s-1} \end{vmatrix}$$

есть минор определителя Вандермонда и не равен 0, поскольку $i_1, i_2, \dots, i_s$ — положительные числа. Это довольно простой факт и встречается в качестве упражнения, например, в [24, стр. 54]. Поэтому эта система имеет единственное решение

$$\begin{cases} q_{i_1}=q'_{i_1}, \\ \quad\cdots\cdots \\ q_{i_s}=q'_{i_s}. \end{cases}$$

Но согласно предположению

$$\begin{cases} q_{i_1}\neq q'_{i_1}, \\ \quad\cdots\cdots \\ q_{i_s}\neq q'_{i_s}. \end{cases}$$

Противоречие. Значит, $t < s$ и, следовательно, любая пара записей из V имеет не более чем $m-1$ общую переменную.

Рассмотрим такую библиотеку V', что $V' \subset V$ и $|V'| = k$. Пусть $I = \langle B^n, V', \overset{b}{\succeq}\rangle$ — соответствующая этой библиотеке ЗИП типа S_{bool}. Пусть U' — произвольный граф из $\mathcal{U}_d(I, \mathcal{F})$.

Возьмем произвольную запись $y \in V'$. Согласно следствию 7 из теоремы 14 о существовании главных цепей, в графе U' существует главная цепь C_y записи y. Для каждой записи из V' выберем по главной цепи и далее будем говорить только об этих главных цепях. Длина цепи C_y не меньше чем $m+r$ и каждому ребру этой цепи приписана некоторая переменная из характеристической функции записи y. Пусть $y = x_{i_1}x_{i_2}\dots x_{i_{m+r}}$, где переменные упорядочены в порядке следования вдоль цепи

C_y (если некоторая переменная встречается в цепи дважды и более, то упорядочиваем по первому вхождению). Сопоставим каждой переменной x_{i_j} ($j \in \{\overline{1, m+r}\}$) первое ребро, которому приписана эта переменная. Сложность этого ребра не меньше чем

$$\mathbf{P}(N_{x_{i_1}\&x_{i_2}\&\ldots\&x_{i_{j-1}}}) = 2^{1-j}.$$

Если ребро цепи C_y, соответствующее переменной x_{i_j}, принадлежит также некоторой другой главной цепи $C_{y'}$, то все ребра, соответствующие переменным $x_{i_1}, \ldots, x_{i_{j-1}}$, также принадлежат цепи $C_{y'}$, так как граф U' — древовидный. Отсюда, учитывая, что любая пара записей из V' имеет не более чем $m-1$ общую переменную, получаем, что все ребра, соответствующие переменным $x_{i_m}, \ldots, x_{i_{m+r}}$, принадлежат только одной главной цепи C_y. Таким образом, каждой записи из V' мы можем сопоставить цепочку из $r+1$ ребра, причем цепочки, соответствующие различным записям, не пересекаются. Суммарная сложность каждой такой цепочки не меньше чем

$$\sum_{j=0}^{r} 2^{1-m-j} = 2^{2-m}(1 - 2^{-r-1}).$$

Следовательно,

$$T(U') \geq k \cdot 2^{2-m}(1 - 2^{-r-1}).$$

Возьмем $r(n) = [\log_{m+1}(n/p)]$. При таком выборе r

$$p\,m + p\sum_{i=1}^{r} m^i \leq p(m+1)^r \leq n,$$

то есть выполняется условие (4.3).

Оценим величину $r(n)$. Для этого, как и ранее, рассмотрим два случая: когда $m = const$, и $m \to \infty$ при $n \to \infty$.

В первом случае, используя (4.1), получим

$$\begin{aligned} r &\geq \frac{1}{\log_2(m+1)} \cdot \log_2 \frac{n}{p} - 1 \sim \\ &\sim \frac{\log_2 n - \log_2 \alpha + \log_2((m-1)!)}{m \log_2(m+1)} \to \infty \end{aligned}$$

при $n \to \infty$.

Во втором случае, используя (4.2), получим аналогичный результат

$$\begin{aligned} r &\geq \frac{1}{\log_2(m+1)} \cdot \log_2 \frac{n}{p} - 1 \sim \\ &\sim \frac{1}{\log_2(m+1)}(\log_2 n - \frac{m-1}{m}\log_2 n - \\ &\quad -\frac{m-1}{m}\log_2 e + \frac{m-1}{m}\log_2(m-1) + \\ &\quad +\frac{1}{2m}\log_2(2\pi(m-1)) - \frac{1}{m}\log_2 \alpha) = \\ &= \frac{\log_2 n - \log_2 \alpha + (m-1)\log_2 \frac{m-1}{e} + \log_2 \sqrt{2\pi(m-1)}}{m\log_2(m+1)} \to \infty \end{aligned}$$

при $n \to \infty$. Откуда следует, что для любого графа $U' \in \mathcal{U}_d(I, \mathcal{F})$ при $n \to \infty$ выполнено $T(U') \gtrsim k \cdot 2^{2-m}$. Следовательно, при $n \to \infty$

$$\mathcal{T}_d(k, S_{bool}, \mathcal{F}) \geq T_d(I, \mathcal{F}) \gtrsim k \cdot 2^{2-m}. \tag{4.4}$$

Верхняя оценка. Воэьмем произвольную ЗИП типа S_{bool}. Построим некоторый ПИГ, который решал бы задачу I. Для этого возьмем граф U^n_{m-1}, описанный в теореме 17. Возьмем произвольную запись $y \in V$. Пусть она равна $x_{i_1}x_{i_2}\ldots x_{i_{m+r}}$. Возьмем в графе U^n_{m-1} вершину, на которой реализуется конъюнкция $x_{i_1}x_{i_2}\ldots x_{i_{m-1}}$ и выпустим из нее цепочку ребер, которым приписаны переменные $x_{i_m}, x_{i_{m+1}}, \ldots, x_{i_{m+r}}$. И так проделаем для каждой записи $y \in V$. Понятно, что полученный граф U над базовым множеством $\mathcal{F}$ будет решать задачу I.

Подсчитаем сложность графа U. Поскольку он состоит из графа U^n_{m-1} и k цепочек, каждая из которых содержит $r+1$ ребер, и сложность каждой из которых равна, как мы отмечали выше, $2^{2-m}(1-2^{-r-1})$, то

$$T(U) = 1 + \sum_{i=1}^{m-1} 2^{1-i}\binom{n}{i} + k\cdot 2^{2-m}(1-2^{-r-1}) = k\cdot 2^{2-m}(1+\bar{o}(1)).$$

В силу произвольности ЗИП I имеем

$$\mathcal{T}_d(k, S_{bool}, \mathcal{F}) \lesssim k \cdot 2^{2-m}.$$

Отсюда, учитывая неравенство (4.4), получаем утверждение теоремы.

Глава 5

Одномерный интервальный поиск

Интервальный поиск имеет очевидное приложение к системам баз данных. В базе данных, содержащей записи о служащих некоторой компании, каждая запись имеет несколько атрибутов, таких, как возраст, жалование и т. д., и может рассматриваться как точка в n-мерном пространстве, в котором каждый атрибут соответствует измерению, а число измерений пространства n равно числу атрибутов. Типичная задача интервального поиска для двух измерений заключается в выявлении всех служащих, чьи возраст и жалованье находятся в заданных интервалах. Другой пример можно найти у Д. Кнута [15]. Он рассматривал базу данных городов США с координатами в виде широты и долготы. К такой базе естественен вопрос о перечислении всех городов, попадающих в некоторой прямоугольник-запрос. Это типичная двумерная задача интервального поиска.

Исследованию задачи интервального поиска (в другом переводе с английского — регионального поиска) посвящено большое количество работ [25, 37, 38, 39, 41, 42, 45, 48, 55, 56].

В данном разделе рассматривается следующая задача. Дано конечное множество точек из отрезка $[0, 1]$. Запрос на поиск задает некий отрезок $[a, b] \subseteq [0, 1]$. Надо перечислить все точки из множества, которые попадают в отрезок $[a, b]$. Это известная

геометрическая задача поиска, описанная, например, в [25] и называемая одномерной задачей интервального поиска. В данном разделе исследуется вопрос, какие алгоритмы возникают, если ограничивать набор доступных средств, или, более формально, при различных базовых множествах. Получены также некоторые нижние оценки, с помощью которых показывается, что соответствующие полученные алгоритмы не могут быть существенно улучшены при данных ограничениях на набор доступных средств. Результаты данного раздела были опубликованы в [8].

Итак, в одномерной задаче интервального поиска множество записей Y_{int} есть отрезок $[0,1]$, множество запросов X_{int} есть множество отрезков $[u,v] \subseteq [0,1]$, или множество пар точек (u,v), таких, что $0 \le u \le v \le 1$, то есть $X_{int} = \{x = (u,v) : 0 \le u \le v \le 1\}$.

На X_{int} задано вероятностное пространство $\langle X_{int}, \sigma, \mathbf{P}\rangle$, где σ — алгебра подмножеств множества X_{int}, содержащая все прямоугольники со сторонами параллельными осям координат и прямоугольные равнобедренные треугольники с катетами также параллельными осям координат, $\mathbf{P}$ — вероятностная мера на σ. Будем считать, что мера $\mathbf{P}$ определяется функцией плотности распределения вероятностей $p(u,v)$, то есть для любого $B \in \sigma$

$$\mathbf{P}(B) = \int_B p(u,v)\,du\,dv.$$

Для удобства будем считать, что $p(u,v)$ определена на всем квадрате $[0,1] \times [0,1]$, но $p(u,v) = 0$ при $(u,v) \notin X_{int}$.

Отношение поиска, которое будем обозначать через ρ_{int}, определяется соотношением

$$(u,v)\rho_{int}y \Longleftrightarrow u \le y \le v,$$

где $(u,v) \in X_{int},\ y \in Y_{int}$.

Тип $S_{int} = \langle X_{int}, Y_{int}, \rho_{int}, \sigma, \mathbf{P}\rangle$ будем называть типом одномерного интервального поиска.

В следующих пунктах мы исследуем задачи типа S_{int} при различных базовых множествах.

5.1 Случай базового множества характеристических функций

Пусть

$$F_1 = \{\chi_{a,\rho_{int}} : a \in Y_{int}\}, \tag{5.1}$$
$$\mathcal{F}_1 = \langle F_1, \emptyset \rangle.$$

Отметим, что $N_f \in \sigma$ для любого предиката $f \in F_1$. Кроме того, понятно, что $\mathcal{F}_1$ полно для типа S_{int}, так как для произвольной библиотеки $V = \{y_1, \ldots, y_k\}$ дерево, изображенное на рисунке 1.5 и соответствующее переборному алгоритму, решает ЗИП $I = \langle X_{int}, V, \rho_{int} \rangle$.

Теорема 20. *Для любой ЗИП I типа S_{int} $T(I, \mathcal{F}_1) = k$.*

Доказательство. Нетрудно убедиться, что для любого $a \in Y_{int}$ точка $(a, a) \in N_{\chi_{a,\rho_{int}}}$ и для любого $a' \in Y_{int}$, такого, что $a' \neq a$ выполняется $(a, a) \notin N_{\chi_{a',\rho_{int}}}$, то есть мы находимся в условиях теоремы 5, откуда следует, что $T(I, \mathcal{F}_1) = k$.

Тем самым теорема доказана.

5.2 Случай простого базового множества

Обозначим

$$M_{a,b} = \{x \stackrel{\text{def}}{=} (u, v) \in X_{int} : u \leq b,\ v \geq a\}.$$

Рассмотрим случай, когда базовое множество предикатов равно

$$F_2 = \{f_{a,b} : (a, b) \in X_{int}\}, \tag{5.2}$$

где $N_{f_{a,b}} = M_{a,b}$, а базовое множество равно $\mathcal{F}_2 = \langle F_2, \emptyset \rangle$.

Отметим, что $N_f \in \sigma$ для любого предиката $f \in F_2$, и для любой записи $y \in Y_{int}$ $\chi_{y,\rho_{int}} = f_{y,y}$, то есть $\mathcal{F}_2$ полно для типа S_{int}.

Справедлива следующая теорема.

Теорема 21. *Если функция плотности распределения вероятностей $p(u, v)$, определяющая меру $\mathbf{P}$ вероятностного пространства над множеством запросов X_{int}, ограничена, то для*

произвольной ЗИП $I = \langle X_{int}, V, \rho_{int} \rangle$ *типа* S_{int}

$$\sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) \le T(I, \mathcal{F}_2) \le \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) + \zeta(k),$$

где $k = |V|,\ \zeta(k) = \underline{O}(\sqrt{k})$ *при* $k \to \infty$.

Доказательство. Нижняя оценка следует из теоремы 4.

Докажем верхнюю оценку. Пусть $p(u, v) \le c = const$. Пусть $m = \left[\sqrt{k}\right]$ и $t_i = i/m$ $(i = \overline{0, m})$ — точки на отрезке $[0, 1]$.

Построим для ЗИП I следующее дерево, которое обозначим через D_m. Возьмем вершину и объявим ее корнем. Выпустим из корня m ребер и i-ому ребру $(i = \overline{1, m})$ припишем предикат f_{t_{i-1}, t_i}. Рассмотрим теперь произвольную запись $y \in V$. Пусть $y \in (t_{i-1}, t_i]$ ($y \in [t_0, t_1]$ при $i = 1$). Тогда из вершины, в которую ведет i-ое ребро, выпустим ребро с предикатом $f_{y,y}$, конец этого ребра объявим листом и припишем этому листу запись y. Эту операцию проделаем со всеми k записями из V. Очевидно, что полученное дерево решает ЗИП I.

Пусть (β, α) — ребро, ведущее в лист, которому соответствует запись y. Его сложность равна

$$\begin{aligned}
\mathbf{P}(N_{\varphi_\beta}) &= \int_0^{t_i} \int_{t_{i-1}}^1 p(u, v)\, dv\, du = \int_0^y \int_{t_{i-1}}^y p(u, v)\, dv\, du + \\
&\quad + \int_0^y \int_y^1 p(u, v)\, dv\, du + \int_y^{t_i} \int_y^1 p(u, v)\, dv\, du \le \\
&\le c \cdot y \cdot (y - t_{i-1}) + \mathbf{P}(O(y, \rho_{int})) + c \cdot (t_i - y)(1 - y) \le \\
&\le c \cdot (t_i - t_{i-1}) + \mathbf{P}(O(y, \rho_{int})) = \frac{c}{m} + \mathbf{P}(O(y, \rho_{int})).
\end{aligned}$$

Следовательно, сумма сложностей ребер второго яруса (а их k штук) не больше чем

$$\sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) + \frac{k \cdot c}{m}.$$

Учитывая, что сложность каждого ребра первого яруса равна 1, получаем

$$T(D_m) \le \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) + \frac{k \cdot c}{m} + m.$$

Остается заметить, что $k \cdot c/m + m = \underline{O}(\sqrt{k})$ при $k \to \infty$.
Тем самым теорема доказана.

Прежде чем перейти к формулировке следующей теоремы докажем некоторые вспомогательные утверждения.

Лемма 10. *Пусть* $f_{a,a} = \bigvee\limits_{j=1}^{n} f_j$, *где* $f_j = \bigwedge\limits_{i=1}^{m_j} f_{a_{ij},b_{ij}}$, *тогда существует такое* $j \in \{\overline{1,n}\}$, *что* $f_j = f_{a,a}$.

Доказательство. Обозначим $\min\limits_{1 \le i \le m_j} b_{ij} = b_j$, $\max\limits_{1 \le i \le m_j} a_{ij} = a_j$. Легко видеть, что $N_{f_j} = M_{a_j,b_j}$. Следовательно, $N_{f_{a,a}} = M_{a,a} = \bigcup\limits_{j=1}^{n} M_{a_j,b_j}$. Очевидно, что $b_j \le a$ и $a_j \ge a$ для любого $j \in \{\overline{1,n}\}$, откуда следует, что $b_j \le a_j$. Легко видеть, что если $b < a$, то $M_{a,b}$ не содержит точек вида (u,u). Но $M_{a,a}$ содержит точку (a,a), следовательно, существует $j \in \{\overline{1,n}\}$ такое, что $b_j = a_j = a$, то есть $f_j = f_{a,a}$, что и требовалось доказать.

Лемма 11. *Пусть* $I = \langle X_{int}, V, \rho_{int} \rangle$ — *произвольная ЗИП типа* S_{int}, U — *произвольный ПИГ над базовым множеством* $\mathcal{F}_2$, *решающий ЗИП* I. *Тогда существует ОИГ* U', *решающий ЗИП* I, *такой, что в любой его лист* α *ведет единственное ребро* (β, α), *причем* $\psi_\beta \ge 2$, $\psi_\alpha = 0$, *и* $T(U') \le T(U)$.

Доказательство. Рассмотрим произвольную запись $y \in V$. Так как U решает I и $O(y, \rho_{int}) \ne \emptyset$, то $L_U(y) \ne \emptyset$ и

$$\chi_{y,\rho_{int}} = f_{y,y} = \bigvee_{\alpha \in L_U(y)} \varphi_\alpha.$$

Иными словами, если считать, что $\mathcal{C} = \{C_1, \ldots, C_n\}$ — множество цепей, ведущих из корня графа U в листья из $L_U(y)$, а проводимость цепи C_j ($j = \overline{1,n}$) равна $f_j = \bigwedge\limits_{i=1}^{m_j} f_{a_{ij},b_{ij}}$, то $f_{y,y} = \bigvee\limits_{j=1}^{n} \bigwedge\limits_{i=1}^{m_j} f_{a_{ij},b_{ij}}$. Согласно лемме 10 существует $j \in \{\overline{1,n}\}$ такое, что $f_j = f_{y,y}$, то есть проводимость цепи C_j равна $f_{y,y}$.

Пусть $V = \{y_1, \ldots, y_k\}$. Мы показали, что для каждой записи y_i ($i = \overline{1,k}$) можно подобрать цепь, ведущую из корня в

лист с записью y, проводимость которой равна $\chi_{y_i,\rho_{int}}$. Среди всех таких цепей (если их несколько) выберем минимальную по длине и обозначим ее C'_i. Обозначим через U_0 ПИГ, получающийся из U выбрасыванием всех ребер, не принадлежащих ни одной из цепей C'_i ($i = \overline{1,k}$). По построению U_0 решает ЗИП I, и его сложность не превышает сложности U.

Покажем, что каждая цепь C'_i ($i = \overline{1,k}$) содержит ровно два полюса: корень и лист, в который она ведет (обозначим его через α_i). Отметим, что $\langle\alpha_i\rangle = y_i$.

Предположим, что это не так, то есть существует некоторая цепь C'_i, такая, что она проходит через некоторый лист $\beta \neq \alpha_i$. Рассмотрим сначала случай, когда $\beta \in L_U(y_i)$. Так как проводимость цепи C'_i равна f_{y_i,y_i}, то и проводимость подцепи цепи C'_i от корня до вершины β будет равна f_{y_i,y_i}, а это противоречит минимальности длины C'_i.

Пусть теперь $\beta \in L_U(y_l)$, где $l \neq i$. Проводимость подцепи цепи C'_i от корня до вершины β (обозначим ее f') покрывает проводимость цепи C'_i (скажем, что предикат f покрывает предикат g, если $N_f \supseteq N_g$) и, значит, $f'(y_i, y_i) = 1$, но так как $\chi_{y_l,\rho_{int}}(y_i, y_i) = 0$, то это противоречит тому, что граф U решает ЗИП I.

Таким образом, каждая из цепей C'_i ($i = \overline{1,k}$) содержит ровно 2 полюса, а так как в U_0 ребер, не принадлежащих цепям C'_i ($i = \overline{1,k}$), нет, то это означает, что в U_0 ровно k листьев, и каждый из них имеет полустепень исхода 0, и в каждый из них ведет единственное ребро. Обозначим ребро, ведущее в лист α_i через (β_i, α_i) ($i = \overline{1,k}$). Если $\psi_{\beta_i} \geq 2$, то мы просто переобозначим цепь C'_i на C''_i. Если $\psi_{\beta_i} = 1$, то рассмотрим цепь C'_i, которой принадлежит ребро (β_i, α_i). Пусть γ_i — ближайшая к β_i вершина в цепи C'_i, такая, что $\psi_{\gamma_i} \geq 2$. Очевидно, что такая вершина есть. Отметим также, что часть цепи C'_i между вершинами γ_i и α_i не принадлежит никакой другой цепи. Заменим в цепи C'_i цепь, ведущую из γ_i в α_i, ребром (γ_i, α_i), которому припишем предикат f_{y_i,y_i}. Полученную цепь обозначим через C''_i. Очевидно, что проводимость цепи C''_i также будет равна f_{y_i,y_i}, а ее сложность не больше сложности цепи C'_i. Такую операцию проделаем для каждого листа α_i ($i = \overline{1,k}$). Полученный в результате граф, состоящий из цепей C''_i ($i = \overline{1,k}$), обозна-

чим через U'. Этот граф удовлетворяет требованиям леммы, и, значит, лемма доказана.

Введем следующее обозначение $M_V = \bigcup\limits_{y \in V} M_{y,y}$, где V — некоторая библиотека.

Лемма 12. *Если V — произвольная библиотека, то существует такое разбиение V на непересекающиеся части $V_1, \ldots, V_t$ (то есть $\bigcup\limits_{i=1}^{t} V_i = V$ и $V_i \cap V_j = \emptyset$, если $i \neq j$), причем, возможно, $V_1 = \emptyset$, что сложность ЗИП $I = \langle X_{int}, V, \rho_{int} \rangle$ типа S_{int}*

$$T(I, \mathcal{F}_2) \geq \sum_{i=2}^{t} (|V_i| + 1)\mathbf{P}(M_{V_i}) + |V_1|.$$

Доказательство. Возьмем произвольный ПИГ $U \in \mathcal{U}(I, \mathcal{F}_2)$. Согласно лемме 11, существует ОИГ U_0, такой, что в каждый лист $\alpha \in \mathcal{L}(U_0)$ ведет единственное ребро (β, α), $\psi_\beta \geq 2$ и $T(U) \geq T(U_0)$. Разобьем множество этих ребер графа U_0 на группы так, что в одну группу объединяются ребра, начала которых совпадают. Если среди этих групп есть группа ребер, исходящих из корня, то назовем эту группу первой группой ребер, а если такой группы нет, то будем считать, что первая группа пустая. Остальные группы занумеруем, начиная с 2. Пусть t — число групп. Через V_i обозначим множество записей, соответствующих концам ребер из i-ой группы ($i = \overline{1,t}$) (возможно, $V_1 = \emptyset$). Поскольку сложность ребра, исходящего из корня равна 1, то сумма сложностей ребер из первой группы равна $|V_1|$.

Теперь рассмотрим случай, когда $i \in \{\overline{2,t}\}$. Обозначим через β^i начало ребер из i-ой группы. Очевидно, $N_{\varphi_{\beta^i}} \supseteq M_{V_i}$ ($i = \overline{2,t}$). Откуда следует, что каждое ребро из i-ой группы имеет сложность не меньшую, чем $\mathbf{P}(M_{V_i})$. Добавим к i-ой группе ($i = \overline{2,t}$) множество ребер, входящих в β^i. Ясно, что сумма сложностей этих ребер также не меньше чем $\mathbf{P}(M_{V_i})$.

Так как все β^i ($i = \overline{2,t}$) различны и для любого листа полустепень его исхода равна 0, то разные группы ребер не пересе-

каются. Следовательно,

$$T(U) \geq T(U_0) \geq \sum_{i=2}^{t}(|V_i|+1)\mathbf{P}(M_{V_i}) + |V_1|.$$

Произвольность выбора графа U доказывает утверждение леммы.

Теорема 22. *Существует такая функция плотности распределения вероятностей $p(u,v)$, что если с помощью нее определить меру $\mathbf{P}$ вероятностного пространства над множеством запросов X_{int}, то существует такая ЗИП $I = \langle X_{int}, V, \rho_{int}\rangle$ типа S_{int}, что*

$$T(I,\mathcal{F}_2) = \sum_{y\in V}\mathbf{P}(O(y,\rho_{int})) + \zeta(k),$$

где $k = |V|$, $\zeta(k) = \underline{O}(\sqrt{k})$ при $k \to \infty$.

Доказательство. Нам надо доказать, что существует такая ЗИП $I = \langle X_{int}, V, \rho_{int}\rangle$, что

$$\sum_{y\in V}\mathbf{P}(O(y,\rho_{int})) + \zeta_1(k) \leq T(I,\mathcal{F}_2) \leq \sum_{y\in V}\mathbf{P}(O(y,\rho_{int})) + \zeta_2(k),$$

где $\zeta_1(k) = \underline{O}(\sqrt{k})$ и $\zeta_2(k) = \underline{O}(\sqrt{k})$ при $k \to \infty$.

Верхняя оценка следует из теоремы 21.

Докажем нижнюю оценку.

Пусть A_a — множество точек плоскости, определяемое следующим соотношением:

$$A_a = \{(u,v) \in X_{int} : 0 \leq v-u \leq a\}, \tag{5.3}$$

где $0 < a < 1/2$. Площадь фигуры A_a равна

$$S(A_a) = \int\int_{A_a} du\,dv = \frac{1}{2} - \frac{(1-a)^2}{2} = a - \frac{a^2}{2}.$$

Рассмотрим следующую функцию плотности распределения вероятностей :

$$p_a(u,v) = \begin{cases} 0, & \text{если } (u,v) \notin A_a, \\ 1/S(A_a), & \text{если } (u,v) \in A_a. \end{cases} \tag{5.4}$$

Возьмем a такое, чтобы $a > 1/3$.

Меру $\mathbf{P}$ вероятностного пространства над X_{int} определим с помощью функции $p_a(u,v)$.

Заметим, что для любого $y \in [a, 1-a]$

$$\mathbf{P}(O(y, \rho_{int})) = \int_0^y \int_y^1 p_a(u,v)\, dv\, du = \frac{a^2}{2 \cdot S(A_a)}.$$

Рассмотрим библиотеку $V = \{y_1, \ldots, y_k\}$, где $y_i = a + (i-1)t$, $t = (1-2a)/k$, $i = \overline{1,k}$.

Заметим, что для любого $i \in \{\overline{1,k}\}$ $a \le y \le 1-a$ и $t < a$ при $k \ge 1$.

Пусть $V_1, \ldots, V_r$ — разбиение библиотеки V, удовлетворяющее условию леммы 12, то есть

$$T(I, \mathcal{F}_2) \ge \sum_{i=2}^{r} (|V_i| + 1)\mathbf{P}(M_{V_i}) + |V_1|,$$

где $I = \langle X_{int}, V, \rho_{int} \rangle$.

Заметим что $P(M_{V_i})$ будет минимально, если V_i состоит из подряд идущих записей из V, и тогда M_{V_i} имеет вид, изображенный на рисунке 5.1, и

$$\begin{aligned}
\mathbf{P}(M_{V_i}) &= \frac{1}{2S(A_a)}((((|V_i|-1)t + a)^2 - \\
&\quad -((|V_i|-1)t)^2 - (|V_i|-1)t^2) = \\
&= \frac{1}{2S(A_a)}(a^2 + 2(|V_i|-1) \cdot t \cdot a - (|V_i|-1)t^2).
\end{aligned}$$

Таким образом,

$$\begin{aligned}
T(I, \mathcal{F}_2) &\ge |V_1| + \sum_{i=2}^{r} (|V_i| + 1)\mathbf{P}(M_{V_i}) \ge \\
&\ge |V_1| + \frac{1}{2S(A_a)} \sum_{i=2}^{r} (|V_i| + 1)(a^2 + (|V_i| - 1)(2ta - t^2)) = \\
&= \frac{a^2 k}{2S(A_a)} + |V_1| \left(1 - \frac{a^2}{2S(A_a)}\right) + \frac{(a-t)^2}{2S(A_a)}(r-1) + \\
&\quad + \frac{2ta - t^2}{2S(A_a)} \sum_{i=2}^{r} |V_i|^2 > \\
&> \frac{a^2 k}{2S(A_a)} + |V_1| \left(1 - \frac{a^2}{2S(A_a)}\right) + c_1(r-1) + \\
&\quad + \frac{ta}{2S(A_a)} \sum_{i=2}^{r} |V_i|^2, \qquad (5.5)
\end{aligned}$$

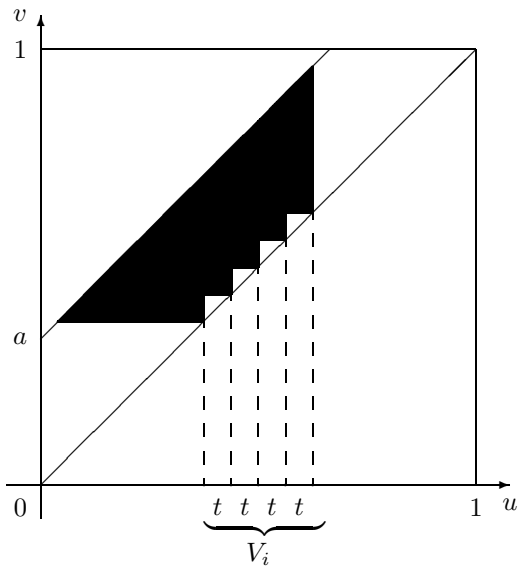


Рис. 5.1: Тени подряд идущих записей

где $0 < c_1 < \dfrac{(a-t)^2}{2S(A_a)}$. Такая константа существует, поскольку $t < a$ и t убывает с ростом k . Из (5.5) также видно, что $|V_1| = \bar{o}(k)$ при $k \to \infty$, поскольку в противном случае нижняя оценка превысит верхнюю, чего быть не может.

Теперь воспользуемся следующим известным фактом. Пусть $m_1, \ldots, m_l$ — целые неотрицательные числа. Тогда минимум функции $\sum_{i=1}^{l} m_i^2$ при условии, что $\sum_{i=1}^{l} m_i = n$, достигается, когда первые $n - [n/l] \cdot l$ слагаемых равны $[n/l] + 1$, а остальные слагаемые равны $[n/l]$, то есть

$$\begin{aligned}\sum_{i=1}^{l} m_i^2 &\geq \left(n - \left[\frac{n}{l}\right] l\right)\left(\left[\frac{n}{l}\right] + 1\right)^2 + \left(l + \left[\frac{n}{l}\right] l - n\right)\left[\frac{n}{l}\right]^2 = \\ &= 2n\left[\frac{n}{l}\right] + n - \left[\frac{n}{l}\right]^2 l - \left[\frac{n}{l}\right] > n\left[\frac{n}{l}\right],\end{aligned}$$

откуда

$$\sum_{i=2}^{r} |V_i|^2 > (k - |V_1|) \left[\frac{k - |V_1|}{r} \right].$$

Подставляя это в 5.5 и учитывая, что $t = (1-2a)/k$, находим, что

$$T(I, \mathcal{F}_2) \geq \frac{a^2 k}{2S(A_a)} + \zeta_2(k),$$

где

$$\zeta_2(k) = c_1(r-1) + \frac{(1-2a)a}{2S(A_a)k}(k - |V_1|) \left[\frac{k - |V_1|}{r} \right].$$

Заметим, что при $k \to \infty$,

$$\zeta_2(k) = c_1 r + c_2 k/r + \bar{o}(r + k/r),$$

где c_1 и c_2 — константы.

Остается заметить, что $\zeta_2(k)$ минимально по порядку при $r \asymp k$ и тогда $\zeta_2(k) \asymp \sqrt{k}$. Последнее доказывает утверждение теоремы, поскольку

$$\frac{a^2 k}{2S(A_a)} = \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})).$$

5.3 Базовое множество логарифмического поиска

В данном пункте мы будем в качестве базового множества предикатов рассматривать множество $F_3 = F_2 \cup \{\bar{f}_{0,a} : a \in [0,1]\}$, где F_2 определяется соотношением (5.1), а черта обозначает логическое отрицание, и положим $\mathcal{F}_3 = \langle F_3, \emptyset \rangle$.

Очевидно, что для $N_f \in \sigma$ для любого $f \in F_3$.

Справедлива следующая теорема.

Теорема 23. *Пусть $I = \langle X_{int}, V, \rho_{int} \rangle$ — ЗИП типа S_{int}, где $V = \{y_1, \ldots, y_k\}$, причем $0 \leq y_1 \leq \ldots \leq y_k \leq 1$. Тогда*

$$T(I, \mathcal{F}_3) \leq \sum_{i=1}^{k-1} \mathbf{P}(O(y_i, \rho_{int})) + 2] \log_2 k[.$$

Доказательство. Возьмем произвольное ориентированное бинарное сбалансированное дерево с корнем и k концевыми вершинами высоты $]\log_2 k[$, у которого все ребра ориентированы от корня к концевым вершинам. Объявим концевые вершины листьями. Из каждой внутренней вершины дерева исходит 2 ребра, назовем одно из них левым (Л), а другое правым (П). Тогда каждому листу можно сопоставить слово из символов Л и П в соответствии с ребрами, встречающимися в цепи от корня к данному листу, причем первый символ слова соответствует ребру, исходящему из корня, а последний ребру, входящему в лист. Упорядочим листья в соответствии с лексикографическим порядком слов, соответствующих листьям, считая, что символ Л меньше символа П. Теперь i-ому листу (обозначим его α_i) сопоставим запись y_i ($i = \overline{1,k}$). Определим теперь нагрузку ребер дерева предикатами из F_3 следующим образом. Рассмотрим произвольную внутреннюю вершину β. Обозначим через V_β множество записей соответствующих листьям ветви, растущей из вершины β. Отметим, что V_β состоит из подряд идущих записей, то есть, если y_i запись с минимальным номером в V_β, а y_j — с максимальным, то $y_l \in V_\beta$ для всех $l \in \{\overline{i,j}\}$. Обозначим через β' и β'' вершины, в которые ведут соответственно левое и правое ребра, исходящие из β. Пусть y_j — запись с максимальным номером в $V_{\beta'}$. Если β' — не лист, то сопоставим левому ребру, исходящему из β, предикат f_{0,y_j}, а если лист, то — f_{y_j,y_j}. Правому ребру сопоставим $\bar{f}_{0,y_j}$, если β'' — не лист, и $f_{\langle\beta''\rangle,\langle\beta''\rangle}$, если β'' — лист. Такую операцию проделаем для всех внутренних вершин, тем самым определим нагрузку всех ребер. Теперь из листа α_i выпустим ребро в лист α_{i+1} и припишем этому ребру предикат $f_{y_{i+1},y_{i+1}}$. Проделаем эту операцию для всех номеров i от 1 до $k-1$. Полученный граф обозначим через U^1.

Отметим одно достаточно очевидное свойство графа U^1. Пусть β — вершина графа U^1, не являющаяся листом. Пусть $V_\beta = \{y_i, y_{i+1}, \ldots, y_j\}$. (Здесь и далее V_β — то множество, которое определялось ранее с помощью дерева.) Тогда

- если $i = 1$ и $j = k$, то $N_{\varphi_\beta} = X_{int}$;
- если $i = 1$ и $j \neq k$, то$N_{\varphi_\beta} = \{(u,v) \in X_{int} : 0 \leq u \leq y_j\}$;
- если $i \neq 1$ и $j = k$, то $N_{\varphi_\beta} = \{(u,v) \in X_{int} : y_{i-1} < u \leq 1\}$;

- если $i \neq 1$ и $j \neq k$, то $N_{\varphi_\beta} = \{(u,v) \in X_{int} : y_{i-1} < u \leq y_j\}$.

Покажем, что U^1 решает ЗИП I. По теореме 1, для этого достаточно показать, что $\varphi_{\alpha_i} = \chi_{y_i,\rho_{int}}$ для любого $i \in \{\overline{1,k}\}$.

Доказывать это будем индукцией по i.

Базис индукции. $i = 1$.

В лист α_1 ведет единственное ребро из вершины, которую обозначим через β'. Очевидно, $\varphi_{\alpha_1} = \varphi_{\beta'} \& f_{y_1,y_1}$. Через y_j обозначим максимальную запись в библиотеке $V_{\beta'}$. Примем $c = y_j$, если $j \neq k$, и $c = 1$, если $j = k$. Очевидно, $c \geq y_1$. Тогда

$$\begin{aligned} N_{\varphi_{\alpha_1}} &= \{(u,v) \in N_{\varphi_{\beta'}} : u \leq y_1,\ v \geq y_1\} = \\ &= \{(u,v) \in X_{int} : 0 \leq u \leq c,\ u \leq y_1,\ v \geq y_1\} = \\ &= \{(u,v) \in X_{int} : 0 \leq u \leq y_1,\ v \geq y_1\} = M_{y_1,y_1} = N_{f_{y_1,y_1}}, \end{aligned}$$

то есть $\varphi_{\alpha_1} = f_{y_1,y_1}$.

Индуктивный переход. Предположим, что для некоторого $i \geq 1$ функция фильтра листа α_i равна $\varphi_{\alpha_i} = f_{y_i,y_i}$.

Рассмотрим лист α_{i+1} $(i+1 \leq k)$. В него ведут два ребра: из листа α_i и некоторой вершины β, не являющейся листом. Пусть y_l и y_j — минимальная и максимальная записи в библиотеке V_β соответственно. Пусть

$$c_1 = \begin{cases} y_{l-1}, & \text{если } l \neq 1, \\ 0, & \text{если } l = 1, \end{cases}$$

$$c_2 = \begin{cases} y_j, & \text{если } j \neq k, \\ 1, & \text{если } j = k, \end{cases}$$

Очевидно, что $c_1 \leq y_i$ и $c_2 \geq y_{i+1}$. Тогда

$$\varphi_{\alpha_{i+1}} = (\varphi_{\alpha_i} \& f_{y_{i+1},y_{i+1}}) \vee (\varphi_\beta \& f_{y_{i+1},y_{i+1}}) = (\varphi_{\alpha_i} \vee \varphi_\beta) \& f_{y_{i+1},y_{i+1}},$$

или

$$\begin{aligned} N_{\varphi_{\alpha_{i+1}}} &= (\{(u,v) \in X_{int} : u \leq y_i,\ v \geq y_i\} \cup \\ &\quad \cup \{(u,v) \in X_{int} : c_1 < u \leq c_2\}) \cap \\ &\quad \cap \{(u,v) \in X_{int} : u \leq y_{i+1},\ v \geq y_{i+1}\} = \\ &= \{(u,v) \in X_{int} : u \leq y_{i+1},\ v \geq y_{i+1}\} = N_{f_{y_{i+1},y_{i+1}}}. \end{aligned}$$

Тем самым мы показали, что U^1 решает I.

Покажем, что

$$T(U^1) \le \sum_{i=1}^{k-1} \mathbf{P}(O(y_i, \rho_{int})) + 2] \log_2 k[.$$

Ярусом графа назовем множество вершин с одинаковой высотой (высота вершины — есть длина наименьшей цепи от корня к данной вершине), а значение высоты — номером яруса.

Пусть β_1 и β_2 — две вершины, не являющиеся листьями, из одного яруса. Очевидно, что $V_{\beta_1} \cap V_{\beta_2} = \emptyset$, откуда $N_{\varphi_{\beta_1}} \cap N_{\varphi_{\beta_2}} = \emptyset$. Поэтому

$$\sum_{\beta} \mathbf{P}(N_{\varphi_\beta}) = \mathbf{P}\left(\bigcup_{\beta} N_{\varphi_\beta}\right) \le \mathbf{P}(X_{int}) = 1,$$

где суммирование берется по всем вершинам одного яруса, не являющимися листьями. В графе U^1 имеется $] \log_2 k[$ ярусов, а из каждой вершины, не являющейся листом, исходит 2 ребра, следовательно, сумма сложностей ребер, исходящих из вершин, не являющихся листьями, не превышает $2] \log_2 k[$. Из каждого листа α_i ($i = \overline{1, k-1}$) исходит одно ребро и его сложность равна $\mathbf{P}(O(y_i, \rho_{int}))$. Следовательно,

$$T(I, \mathcal{F}_2) \le T(U^1) \le \sum_{i=1}^{k-1} \mathbf{P}(O(y_i, \rho_{int})) + 2] \log_2 k[.$$

Тем самым теорема 23 доказана.

Алгоритм поиска, соответствующий графу U^1, построенному в доказательстве теоремы 23, состоит в следующем. В упорядоченной по возрастанию библиотеке с помощью бинарного поиска за $\log_2 k$ шагов находится самая левая запись, находящаяся не левее левого конца запроса. Затем слева направо, начиная с найденной записи, просматриваются записи и сравниваются с правым концом запроса, и если оказывается, что очередная запись не больше правого конца, то эта запись включается в ответ, а если больше, то поиск прекращается, то есть — это традиционный алгоритм решения данной задачи поиска с помощью структуры данных, называемой прошитым двоичным деревом [25]. Там же [25, стр. 93] утверждается, что описанный алгоритм

оптимален как по времени поиска, так и по памяти, но имеется в виду время в худшем случае, а оптимальность понимается по порядку.

Справедлива следующая теорема.

Теорема 24. *Для любого $k \in \mathbf{N}$ существует такая функция плотности распределения вероятностей $p^k(u,v)$, определяющая меру вероятностного пространства над X_{int}, и такая библиотека V_k мощности k, определяющая вместе с $p^k(u,v)$ ЗИП $I_k = \langle X_{int}, V_k, \rho_{int}\rangle$, что для любого измеримого базового множества $\mathcal{F} = \langle F, \emptyset\rangle$, допустимого для I, справедливо неравенство*

$$\begin{aligned} T(I_k,\mathcal{F}) &\geq \sum_{y\in V_k} \mathbf{P}(O(y,\rho_{int})) + \frac{(3\log_3 k - 1)k}{2k+1} \geq \\ &\geq \sum_{y\in V_k} \mathbf{P}(O(y,\rho_{int})) + c\log_2 k, \end{aligned}$$

где c — константа, в качестве которой можно взять, например,

$$c = \frac{2(3\log_3 2 - 1)}{5}.$$

Доказательство. Пусть нам дано число $k \in \mathbf{N}$. В качестве библиотеки V_k возьмем $V_k = \{y_1, \ldots, y_k\}$, где $y_i = i/(k+1)$, $i = \overline{1,k}$. В качестве плотности распределения $p^k(u,v)$ возьмем функцию $p_{1/(k+1)}(u,v)$, определяемую соотношением (5.4).

Нетрудно видеть, что $\mathbf{P}(O(y_i,\rho_{int})) = 1/(2k+1)$ для любого $i \in \{\overline{1,k}\}$ и $\mathbf{P}(O(y_i,\rho_{int}) \cap O(y_j,\rho_{int})) = 0$ для любых $i,j \in \{\overline{1,k}\}$, если $i \neq j$.

Это означает, что мы находимся в условиях теоремы 7, и, значит, для любого измеримого базового множества $\mathcal{F} = \langle F, \emptyset\rangle$, допустимого для I_k, справедливо неравенство

$$T(I_k,\mathcal{F}) \geq \frac{3k\log_3 k}{2k+1}.$$

Учитывая, что

$$\sum_{y\in V_k} \mathbf{P}(O(y,\rho_{int})) = \frac{k}{2k+1},$$

получаем утверждение теоремы.

5.4 Базовое множество сверхлогарифмического поиска

В данном пункте мы рассмотрим случай, когда базовое множество предикатов равно

$$F_4 = F_3 \cup \{f_{-,a} : N_{f_{-,a}} = A_a, a \in [0,1]\} \cup \{\bar{f}_{-,a} : a \in [0,1]\},$$

где A_a — множество, определяемое соотношением (5.3), а черта обозначает логическое отрицание, и положим $\mathcal{F}_4 = \langle F_4, \emptyset \rangle$.

Понятно, что для $N_f \in \sigma$ для любого $f \in F_4$.

Справедлива следующая теорема.

Теорема 25. *Пусть функция плотности распределения вероятностей $p(u,v)$, определяющая меру $\mathbf{P}$ вероятностного пространства над множеством запросов X_{int}, такая, что $p(u,v) \le c = const$. Пусть $I = \langle X_{int}, V, \rho_{int} \rangle$ — произвольная ЗИП типа S_{int}. Тогда*

$$T(I, \mathcal{F}_4) \le \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) + 2 \cdot] \log_2 \log_2 |V| [+6 + 2c.$$

Доказательство. Упорядочим записи в библиотеке $V = \{y_1, \ldots, y_k\}$ так, что $y_1 \le y_2 \le \ldots y_k$.

Пусть $S = \{s_1, \ldots, s_m\}$, где $s_i = i/(m+1),\ i = \overline{1,m}$. Здесь и далее $m = [\log_2 k]$.

Для каждого s_i $(i = \overline{1,m})$ найдем целое число l_i, которое является номером ближайшей к числу s_i записи из V, меньшей, чем s_i, причем если такой записи не существует, то примем $l_i = 0$.

Построим для V граф U^1 так, как это было сделано в доказательстве теоремы 23. Объявим корень графа U^1 обычной внутренней вершиной и обозначим через β_1. Проведем в β_1 ребро, начало его обозначим β_0 и объявим корнем полученного графа. Припишем этому ребру предикат $f_{-,1/(m+1)}$ из F_4. Выпустим из корня еще одно ребро, конец которого обозначим через β_2, и припишем этому ребру предикат $\bar{f}_{-,1/(m+1)}$. Построим ориентированное сбалансированное бинарное дерево с m концевыми вершинами с корнем в вершине β_2, все ребра которого ориентированы от β_2 к концевым вершинам. Из каждой внутренней

вершины дерева исходит 2 ребра, одно из которых назовем левым (Л), а другое правым (П). Это так же, как и в доказательстве теоремы 23, порождает отношение порядка концевых вершин, и в соответствии с ним занумеруем концевые вершины этого дерева в порядке возрастания и обозначим их через $\beta'_1, \ldots, \beta'_m$, то есть вершине β'_1 соответствует слово (Л,Л,. . .,Л), а вершине β'_m слово (П,П,. . .,П). Нагрузку ребер дерева предикатами из F_3 определим следующим образом. Пусть β — произвольная внутренняя вершина дерева. Пусть β'_j — концевая вершина с максимальным номером в ветви, растущей из вершины, в которое ведет левое ребро, исходящее из β. Тогда левому ребру, исходящему из β, припишем предикат f_{0,s_j}, а правому $\bar{f}_{0,s_j}$. Такую операцию проделаем для всех внутренних вершин дерева, и полностью определим нагрузку его ребер. Напомним, что в графе U^1 листья обозначаются $\alpha_1, \ldots, \alpha_k$ и им приписаны соответственно записи $y_1, \ldots, y_k$. Построим еще k вершин, обозначим их $\alpha'_1, \ldots, \alpha'_k$ и также объявим их листьями и припишем каждому листу α'_i ($i = \overline{1,k}$) запись y_i. Теперь для каждого $i \in \{\overline{2,k}\}$ из листа α'_i построим ребро в лист α'_{i-1} и припишем ребру $(\alpha'_i, \alpha'_{i-1})$ предикат $f_{y_{i-1},y_{i-1}}$. Затем для каждого $i \in \{\overline{1,m}\}$, если $l_i \neq 0$, то из вершины β'_i выпустим ребро в лист α'_{l_i} и припишем ему предикат $f_{y_{l_i},y_{l_i}}$, и если при этом $l_i < k$, то из вершины β'_i выпустим еще одно ребро в лист α_{l_i+1} и припишем ему предикат $f_{y_{l_i+1},y_{l_i+1}}$.

Полученный граф обозначим через U^2.

Покажем, что ПИГ U^2 решает ЗИП $I = \langle X_{int}, V, \rho_{int} \rangle$.

Рассмотрим произвольную запись $y_i \in V$. По построению ясно, что $L_{U^2}(y_i) = \{\alpha_i, \alpha'_i\}$. В каждый из листьев α_i и α'_i ведут только ребра, которым приписан предикат f_{y_i,y_i}, следовательно,

$$N_{\varphi_{\alpha_i} \vee \varphi_{\alpha'_i}} \subseteq N_{f_{y_i,y_i}} = O(y_i, \rho_{int}). \tag{5.6}$$

Пусть $x = (u, v)$ — произвольный запрос, такой что $u \leq y_i \leq v$, то есть $x \in O(y_i, \rho_{int})$. Покажем, что $\varphi_{\alpha_i}(x) \vee \varphi_{\alpha'_i}(x) = 1$.

Рассмотрим сначала случай, когда $x \in A_{1/(m+1)}$.

Тогда проводимость ребра (β_0, β_1) равна единице, а ребра (β_0, β_2) — нулю. Поскольку любая цепь, ведущая в лист α'_i проходит через ребро (β_0, β_2), то $\varphi_{\alpha'_i}(x) = 0$.

Обозначим через G'_β подграф графа U^2, состоящий из цепей,

исходящих из вершины β.

Так как проводимость ребра (β_0, β_1) на запросе x равна 1, то мы выходим в вершину β_1. По построению граф G'_{β_1} совпадает с U^1 и, следовательно, $\varphi_{\alpha_i}(x) = 1$, так как U^1 решает ЗИП I.

Теперь рассмотрим случай, когда $x \notin A_{1/(m+1)}$.

Тогда проводимость ребра (β_0, β_1) равна нулю, а ребра (β_0, β_2) единице, то есть мы попадем в вершину β_2, а проводимость всех цепей, проходящих через (β_0, β_1) равна 0. Из вершины β_2 сначала растет дерево, и оно обладает тем свойством, что из каждой его внутренней вершины исходит 2 ребра, и им приписаны противоположные предикаты, то есть всегда проводимость только одного из ребер равна 1. Отсюда следует, что для любого запроса всегда существует единственная цепь, соединяющая β_2 с некоторой вершиной β'_j, проводимость которой равна 1. Таким образом, после прохождения дерева мы попадем в некоторую вершину β'_j, причем по построению этого дерева эта вершина, такая, что s_j является минимальным числом из S, таким что $s_j \geq u$.

Предположим сначала, что $y_i < s_j$. Тогда $i \leq l_j$, и поскольку справедливо $u \leq y_i \leq y_{l_j} < s_j \leq v$, то $f_{y_{l_j}, y_{l_j}}(x) = 1$, то есть проводимость ребра $(\beta'_j, \alpha'_{l_j})$ равна 1. Из вершины α'_{l_j} в вершину α'_i ведет цепочка ребер, которым приписаны функции f_{y_r, y_r}, где $r \in \{\overline{i, l_j}\}$, и поскольку $u \leq y_i \leq y_r \leq y_{l_j} < v$, то $f_{y_r, y_r}(x) = 1$ для любого $r \in \{\overline{i, l_j}\}$, то есть проводимость цепи из α'_{l_j} в α'_i равна 1 на запросе x. Таким образом, мы показали, что существует цепь, ведущая из корня в лист α'_i, проводимость которой равна 1 на запросе x, то есть $\varphi_{\alpha'_i}(x) = 1$. Отметим, что если даже $f_{y_{l_j+1}, y_{l_j+1}}(x) = 1$, то есть из вершины ($\beta'_j$ мы попадем еще и в вершину α_{l_j+1}, то все равно $\varphi_{\alpha_i}(x) = 0$, так как из листа α_{l_j+1} в лист α_i нет цепи.

Осталось рассмотреть случай, когда $y_i \geq s_j$. Тогда $i \geq l_j + 1$, и поскольку справедливо $u \leq s_j \leq y_{l_j+1} \leq y_i \leq v$, то $f_{y_{l_j+1}, y_{l_j+1}}(x) = 1$, и мы попадем в лист α_{l_j+1}. Из листа α_{l_j+1} в лист α_i существует цепь, и ее проводимость по аналогии с предыдущим случаем равна 1 на запросе x, то есть $\varphi_{\alpha_i}(x) = 1$. Также аналогично предыдущему случаю $\varphi_{\alpha'_i}(x) = 0$.

Таким образом, мы показали, что если $x \in O(y_i, \rho_{int})$, то

выполняется $\varphi_{\alpha_i}(x) \vee \varphi_{\alpha'_i}(x) = 1$. Более того мы показали, что при $x \in O(y_i, \rho_{int})$

$$\varphi_{\alpha_i}(x) \wedge \varphi_{\alpha'_i}(x) = 0. \tag{5.7}$$

Учитывая (5.6), получаем, что

$$\varphi_{\alpha_i} \vee \varphi_{\alpha'_i} = \chi_{y_i,\rho_{int}}, \tag{5.8}$$

более того, с учетом (5.7) будем иметь, что

$$N_{\varphi_{\alpha_i}} \cap N_{\varphi_{\alpha'_i}} = \emptyset, \tag{5.9}$$

В силу произвольности записи y_i получаем, что U^2 решает ЗИП I.

Оценим сложность графа U^2.

Обозначим через $\mathcal{N}$ множество всех ребер графа U^2. Пусть $\mathcal{N}_1$ — множество ребер графа U^2, исходящих из листьев, и $\mathcal{N}_2 = \mathcal{N} \setminus \mathcal{N}_1$. Если обозначить через $T(\mathcal{N})$ - сумму сложностей ребер из множества $\mathcal{N}$, то

$$T(U^2) = T(\mathcal{N}) = T(\mathcal{N}_1) + T(\mathcal{N}_2). \tag{5.10}$$

Оценим сначала $T(\mathcal{N}_1)$.

Множество $\mathcal{N}_1$ состоит из $2(k-1)$ ребер, исходящих из листьев α_i ($i = \overline{1, k-1}$) и α_i ($j = \overline{2, k}$), следовательно, учитывая (5.9), а затем (5.8), находим, что

$$\begin{aligned} T(\mathcal{N}_1) &= \sum_{i=1}^{k-1} \mathbf{P}(N_{\varphi_{\alpha_i}}) + \sum_{i=2}^{k} \mathbf{P}(N_{\varphi_{\alpha'_i}}) \leq \\ &\leq \sum_{i=1}^{k} (\mathbf{P}(N_{\varphi_{\alpha_i}}) + \mathbf{P}(N_{\varphi_{\alpha'_i}})) = \\ &= \sum_{i=1}^{k} (\mathbf{P}(N_{\varphi_{\alpha_i} \vee \varphi_{\alpha'_i}})) = \sum_{i=1}^{k} (\mathbf{P}(O(y_i, \rho_{int})). \end{aligned} \tag{5.11}$$

Обозначим через G''_{β_1} (G''_{β_2}) подграф графа G'_{β_1} (G'_{β_2}), состоящий из ребер, не принадлежащих $\mathcal{N}_1$.

Граф G''_{β_1} представляет собой дерево, построенное в доказательстве теоремы 23. Поскольку в вершину $\beta - 1$ проходят только записи из множества $A_{1/(m+1)}$, то по аналогии с тем как

это делалось в теореме 23, легко показать, что сумма сложностей ребер, исходящих из вершин одного яруса дерева G''_{β_1}, не превышает

$$2\mathbf{P}(A_{1/(m+1)}) \le \frac{2c}{m+1} - \frac{c}{(m+1)^2} \le \frac{2c}{\log_2 k}. \qquad (5.12)$$

Аналогично показывается, что сумма сложностей ребер, исходящих из вершин одного яруса дерева G''_{β_2}, не превышает

$$2 \cdot \mathbf{P}(X_{int} \setminus A_{1/(m+1)}).$$

Поскольку количество ярусов в деревьях G''_{β_1} и G''_{β_2} не превышает соответственно $]\log_2 k[$ и $]\log_2[\log_2 k][+1$, то в силу (5.12) с учетом ребер, исходящих из корня, имеем

$$T(\mathcal{N}_2) \le 2\log_2\log_2 k + 2c + 6.$$

С учетом (5.10) и (5.11) получаем утверждение теоремы 25.

Алгоритм поиска, соответствующий графу U^2, представляет собой следующее.

Пусть нам дан некий запрос $x = (u, v) \in X_{int}$. Поиск по этому запросу будем осуществлять следующим образом.

Сначала вычислим длину интервала x.

Если $v - u < 1/([\log_2 k]+1)$, то ответ будем искать по методу, описанному в теореме 23. Если $v - u \ge 1/([\log_2 k]+1)$, то мы за время $\underline{O}(\log_2\log_2 k)$ найдем в множестве S самую левую точку s_i, попадающую в интервал x (такая точка обязательно существует), затем по ссылке l_i идем в ближайшую слева к s_i запись библиотеки V и проверяем попадет ли она в x, если попадет, то справа налево просматриваем следующие записи и проверяем на попадание в x. Затем идем по ссылке $l_i + 1$ и, начиная с записи, в которую ведет эта ссылка, просматриваем слева направо записи с проверкой на попадание в x.

Таким образом, в первом случае помимо перечисления ответа мы тратим время $\underline{O}(\log_2 k)$, а во втором $\underline{O}(\log_2\log_2 k)$. Поскольку в подавляющем большинстве интервал x будет достаточно большим, то мы получим оценку теоремы 25.

5.5 Мгновенное решение

Пусть

$$G_1 = \{g_{\cdot,m}(u,v) = \max(1,]u \cdot m[) : m \in \mathbf{N}\}, \tag{5.13}$$

$$G_2 = \{g_{\leq,a}(u,v) = \begin{cases} 1, \text{ если } u \leq a \\ 2, \text{ если } u > a \end{cases} : a \in [0,1]\}, \tag{5.14}$$

$$G_3 = \{ g_{-,m}(u,v) = \\ = \begin{cases} 1, & \text{если } 0 \leq v-u < 1/m \\ 2 & \text{в противном случае} \end{cases} : m \in \mathbf{N}\}, \tag{5.15}$$

$$\mathcal{F}_5 = \langle F_1, G_1 \cup G_2 \cup G_3 \rangle, \tag{5.16}$$

где F_1 определяется соотношением (5.1).

Справедлива следующая теорема.

Теорема 26. *Пусть ЗИП $I = \langle X_{int}, V, \rho_{int} \rangle$ — одномерная задача интервального поиска, то есть задача типа S_{int}, $\mathcal{F}_5$ — базовое множество, определяемое соотношениями (5.13)–(5.16) и $R(I) \stackrel{\text{def}}{=} \sum_{y \in V} \mathbf{P}(O(y, \rho_{int}))$. Тогда, если функция плотности вероятности $p(x)$, определяющая вероятностную меру $\mathbf{P}$, ограничена константой c, то*

$$R(I) < T(I, \mathcal{F}_5, 4\,|V| - 1 + 6\,c[\log_2 |V|]) \leq R(I) + 5.$$

Доказательство. Пусть $V = \{y_1, \ldots, y_k\}$, причем $y_1 \leq \cdots \leq y_k$, то есть V — множество, упорядоченное в порядке неубывания своих элементов.

Нижняя оценка следует из теоремы 4.

Построим ИГ, на которой достигается верхняя оценка одномерной задачи интервального поиска.

Возьмем точку, объявим ее корнем графа и обозначим β_0. Выпустим из корня два ребра, одно из которых будем считать левым, а другое правым. Конец левого ребра обозначим β_1, а конец второго — β_2.

Пусть m — некоторый параметр, значение которого определим позже.

Припишем корню переключатель $g_{-,m}(u,v)$ из множества G_3. Левому ребру припишем 1, а правому 2.

Выпустим из вершины β_1 первое дерево бинарного поиска для библиотеки V, описанное в разделе 3.1, но только нагрузку переключательных вершин мы будем брать из множества G_2, а вместо предикатов $f_{=,a}$ использовать предикаты $\chi_{a,\rho_{int}} \in F_1$. Обозначим это дерево через D.

Для удобства дальнейшего изложения назовем множество переключательных ребер дерева D переключательной частью дерева D, а множество предикатных ребер — предикатной частью дерева D.

Обозначим лист, которому соответствует запись y_i, через α_i ($i = \overline{1,k}$). Из каждого листа α_i ($i = \overline{1,k-1}$) выпустим ребро, ведущее в лист α_{i+1}, и припишем ему предикат $\chi_{y_{i+1},\rho_{int}} \in F_1$.

Это множество из $(k-1)$-го ребра назовем правосторонней концевой цепью.

Теперь из вершины β_2 (напомним, что это вершина, в которую ведет правое ребро, исходящее из корня) выпустим $m-1$ ребро, припишем им числа от 1 до $m-1$, объявим β_2 точкой переключения и припишем ей переключатель $g_{\cdot,m} \in G_1$ (напомним, что m — введенный ранее параметр). Отметим, что из β_2 мы выпустили именно $m-1$ ребро, хотя переключатель $g_{\cdot,m}$ может принимать m значений.

Конец ребра, исходящего из вершины β_2 и имеющего номер i, обозначим β'_i.

Введем множество номеров $S = \{s_1, \ldots, s_{m-1}\}$ такое, что s_i — номер записи из V такой, что y_{s_i} — ближайшая слева запись к точке i/m ($i = \overline{1,m-1}$), а если такой записи не существует, то $s_i = 0$.

Возьмем k новых точек, объявим их листьями и обозначим $\alpha'_1, \ldots, \alpha'_k$. Каждому листу α_i ($i = \overline{1,k}$) припишем запись y_i (тем самым каждая запись y_i будет приписана двум листьям — α_i и α'_i). Из каждого листа α'_i ($i = \overline{2,k}$) выпустим ребро, ведущее в лист α'_{i-1}, и припишем ему предикат $\chi_{y_{i-1},\rho_{int}} \in F_1$.

Это множество из $(k-1)$-го ребра назовем левосторонней концевой цепью.

Теперь для каждой вершины β'_i проделаем следующие действия. Если $s_i \neq 0$, то из β'_i выпустим ребро, ведущее в лист α'_{s_i}, и припишем ему предикат $\chi_{y_{s_i},\rho_{int}} \in F_1$. Если $s_i < k$, то из β'_i выпустим ребро, ведущее в лист α_{s_i+1}, и припишем ему

предикат $\chi_{y_{s_i+1},\rho_{int}} \in F_1$.

Это множество ребер, исходящих из вершин β_i' ($i = \overline{1, m-1}$), назовем правой предикатной частью.

Полученный таким образом ИГ обозначим через U_0.

Покажем, что граф U_0 решает одномерную задачу интервального поиска $I = \langle X_{int}, V, \rho_{int} \rangle$.

Каждая запись $y_i \in V$ соответствует ровно двум листьям графа U_0, то есть $L_{U_0}(y_i) = \{\alpha_i, \alpha_i'\}$. Так как $O(y, \rho_{int}) = N_{\chi_{y,\rho_{int}}}$ и так как в каждый лист α_i и α_i' ведут только ребра, которым приписаны предикаты $\chi_{y_i,\rho_{int}}$, то

$$N_{\varphi_{\alpha_i} \vee \varphi_{\alpha_i'}} \subseteq O(y_i, \rho_{int}).$$

Тогда согласно теореме 1 достаточно показать, что для любого $y_i \in V$

$$N_{\varphi_{\alpha_i} \vee \varphi_{\alpha_i'}} \supseteq O(y_i, \rho_{int}),$$

или, что то же самое, показать, что для $\forall x \in O(y_i, \rho_{int})$ либо $\varphi_{\alpha_i}(x) = 1$, либо $\varphi_{\alpha_i'}(x) = 1$, то есть из корня либо в лист α_i, либо в лист α_i' существует проводящая цепь.

Пусть как и ранее $A_a = \{x = (u, v) : 0 \leq v - u \leq a\}$.

Возьмем произвольную запись $y_i \in V$.

Рассмотрим сначала случай, когда $x = (u, v) \in A_{1/m} \cap O(y_i, \rho_{int})$. Это означает, что $v - u < 1/m$ и $u \leq y_i \leq v$.

Покажем, что из корня в лист α_i существует проводящая цепь.

В рассматриваемом случае $g_{-,m}(x) = 1$, и проводимость ребра (β_0, β_1) будет равна 1.

Отметим, что проводимость ребра (β_0, β_2) будет равна 0.

По аналогии с теоремой 9 нетрудно заметить, что в дереве D (растущем из вершины β_1) существует проводящая цепь, ведущая из β_1 в такой лист α_j, что запись y_j, ближайшая справа к u точка из V, принадлежащая отрезку $[u, v]$ (такая точка существует, так как $y_i \in [u, v]$). Таким образом, справедливо $u \leq y_j \leq y_i \leq v$. Отсюда легко видеть, что часть правосторонней концевой цепи, ведущая из y_j в y_i, является проводящей.

Таким образом, мы показали наличие проводящей на запросе x цепи, ведущей из корня в лист α_i.

Отметим также, что $\varphi_{\alpha'_i}(x) = 0$, так как в лист α'_i можно попасть только через ребро (β_0, β_2), проводимость которого на запросе x, как мы уже отмечали, равна 0.

Рассмотрим теперь случай, когда

$$x = (u, v) \in (X \backslash A_{1/m}) \cap O(y_i, \rho_{int}),$$

то есть $v - u \geq 1/m$ и $u \leq y_i \leq v$.

В этом случае $g_{-,m}(x) = 2$ и проводимость ребра (β_0, β_1) будет равна 0, а ребра (β_0, β_2) равна 1.

Пусть $j \in \overline{\{1, m-1\}}$ — такой номер, что j/m — ближайшая справа к u точка. Легко видеть, что $g_{\cdot,m}(x) = j$.

Так как $v - u \geq 1/m$, то точка j/m принадлежит отрезку $[u, v]$.

Рассмотрим два случая.

1) $y_i \leq j/m$.

Тогда $u \leq y_i \leq y_{s_j} \leq v$, так как y_{s_j} — ближайшая слева к j/m запись из библиотеки V. Отсюда следует, что проводимость ребра, ведущего из β'_j в лист α'_{s_j} равна 1. Осталось заметить, что проводимость части левосторонней концевой цепи, ведущей из α'_{s_j} в α'_i, также будет равна 1, так как $0 \leq y_i \leq y_{s_j} \leq v$.

Отметим также, что в этой ситуации $\varphi_{\alpha_i}(x) = 0$, так как в лучшем случае мы попадем в вершины правосторонней концевой цепи в точке α_{s_j+1}, которая в этой цепи находится после точки α_i.

2) $y_i > j/m$.

Тогда $u \leq y_{s_j+1} \leq y_i \leq v$. Отсюда следует, что проводимость ребра, ведущего из β'_j в лист α_{s_j+1}, равна 1, и проводимость части правосторонней концевой цепи, ведущей из α_{s_j+1} в α_i, также равна 1 на запросе x.

Аналогично предыдущему случаю $\varphi_{\alpha'_i}(x) = 0$.

Тем самым мы показали, что для $\forall y_i \in V$ и $\forall x \in X : x \rho_{int} y_i$ в графе U_0 существует проводящая на запросе x цепь, ведущая из корня в какой-либо (но ровно в один) из листьев α_i и α'_i.

Что и доказывает, что граф U_0 решает задачу I.

Подсчитаем сложность графа U_0.

Рассмотрим сначала произвольный запрос $x \in A_{1/m}$.

В этом случае

$$T(U_0, x) \leq 1 + (] \log_2 k[-1) + 2 + |\mathcal{J}_{U_0}(x)|.$$

Здесь первое слагаемое соответствует вычислению переключателя $g_{-,m}$ в вершине β_0. Второе слагаемое дают переключатели, входящие в единственную проводящую цепь, пролегающую через переключательную часть дерева D. Третье слагаемое соответствует вычислению одного или двух предикатов, соответствующих предикатным ребрам из предикатной части дерева D, растущим из вершины, в которую ведет проводящая цепь. Четвертое слагаемое соответствует вычислению предикатов, соответствующих ребрам, исходящим из листьев, записи которых входят в ответ (по одному на каждую запись).

Рассмотрим случай, когда $x \in X \backslash A_{1/m}$.

Тогда

$$T(U_0, x) \leq 1 + 1 + 2 + |\mathcal{J}_{U_0}(x)|.$$

Здесь первое слагаемое соответствует вычислению переключателя $g_{-,m}$, второе — переключателя $g_{\cdot,m}$ из вершины β_2, третье — вычислению одного или двух предикатов, приписанных ребрам, исходящим из той вершины β'_j, в которую ведет проводящее ребро из β_2. И, наконец, четвертое слагаемое, как и ранее, соответствует вычислению предикатов, соответствующих ребрам, исходящим из листьев, записи которых входят в ответ. Как мы показали ранее, для любой записи y_i, вошедшей в ответ, ровно у одного из листьев α_i и α'_i функция фильтра будет равна 1, и, следовательно, каждой записи соответствует ровно одно ребро и соответственно ровно один вычисленный предикат.

Подсчитаем сложность графа U_0.

$$\begin{aligned}
T(U_0) &= \mathbf{M}_x\, T(U_0, x) = \mathbf{P}(A_{1/m}) \cdot (2+] \log_2 k[) + \\
&\quad + \mathbf{P}(X \backslash A_{1/m}) \cdot 4 + \mathbf{M}_x\, |\mathcal{J}_{U_0}(x)| \leq \\
&\leq \mathbf{P}(A_{1/m}) \cdot ([\log_2 k] - 1) + 4 + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) \leq \\
&\leq c \cdot ([\log_2 k] - 1) \left(\frac{2}{m} - \frac{1}{m^2} \right) + 4 + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})) \leq \\
&\leq \frac{2c \cdot ([\log_2 k] - 1)}{m} + 4 + \sum_{y \in V} \mathbf{P}(O(y, \rho_{int})).
\end{aligned}$$

Поскольку граф U_0 решает задачу I, то

$$\mathcal{J}_{U_0}(x) = \{y \in V : x \rho_{int} y\},$$

и следовательно, используемое выше равенство

$$\mathbf{M}_x\,|\mathcal{J}_{U_0}(x)| = \sum_{y\in V}\mathbf{P}(O(y,\rho_{int}))$$

доказывается простой сменой порядка суммирования.

Подсчитаем объем графа U_0:

$$Q(U_0) \le 2 + (2k-1) + (k-1) + (k-1) + m + 2m.$$

Здесь первое слагаемое соответствует ребрам, исходящим из β_0. Второе слагаемое есть количество ребер в дереве D. Третье и четвертое слагаемые соответствуют ребрам из правосторонней и левосторонней концевых цепочек. Пятое слагаемое — это ребра, исходящие из вершины β_2. И, наконец, шестое слагаемое не меньше чем число ребер, исходящих из вершин β_i' $(i = \overline{1,m})$.

Возьмем в качестве параметра $m = 2\,c\,[\log_2 k]$ и получим

$$T(U_0) \le 5 + \sum_{y\in V}\mathbf{P}(O(y,\rho_{int})),$$

$$Q(U_0) \le 4\,k - 1 + 6\,c[\log_2 k],$$

что и доказывает утверждение теоремы 26.

Дадим неформальное описание алгоритма, приведенного выше.

Пусть нам дано множество $V = \{y_1,\dots,y_k\}$, в котором мы должны производить поиск. Сначала упорядочим его в порядке возрастания. Если известна оценка сверху c функции плотности вероятности появления запросов, то в качестве параметра m возьмем $m = 2\,c\,[\log_2 k]$, если же c неизвестна, то вместо нее можно взять любое число, например, $c = 2$. Затем для V построим множество номеров $S = \{s_1,\dots,s_{m-1}\}$, описанное выше. Отметим, что оно строится только однажды. Теперь поиск по произвольно взятому интервалу-запросу $x = (u,v)$ производится следующим образом.

Сначала вычисляется длина запроса x.

Если она меньше чем $1/m$, то в множестве V бинарным поиском находится ближайшая справа к точке u запись. Далее, начиная с этой записи, просматриваются слева направо все записи из V и сравниваются с правым концом запроса — точкой v

— до тех пор, пока очередная запись не станет больше v. Тем самым в этом случае, помимо перечисления ответа, производится порядка $\log_2 k$ действий.

Если $v - u \geq 1/m$, то с помощью функции $g_{\cdot,m}$ получаем номер j точки j/m, попадающей в интервал $[u, v]$. Затем, начиная с записи с номером s_j, просматриваем справа налево записи из V и сравниваем с левым концом запроса — точкой u. Как только очередная запись окажется меньше u, мы, начиная с записи с номером $s_j + 1$, просматриваем слева направо записи из V и сравниваем с правым концом запроса — точкой v до тех пор, пока очередная запись не станет больше v. Тем самым в этом случае мы, помимо перечисления ответа, производим 4 лишних действия (сравниваем $v - u$ с $1/m$, вычисляем функцию $g_{\cdot,m}$, делаем 1 лишнее действие, идя справа налево, и 1 лишнее действие, идя слева направо).

Осталось заметить, что параметр m подобран так, что средняя сложность первого случая не превышает 1, если известна оценка сверху функции плотности вероятности, и не превышает некоторой константы, если эта оценка точно не известна.

И, наконец, заметим, что данный алгоритм требует дополнительную память порядка $\log_2 k$, чтобы хранить множество S.

Литература

[1] Адельсон-Вельский Г. М., Ландис Е. М. Алгоритм организации информации. *ДАН СССР* (1962) **146**, 263–266.

[2] Альсведе Р., Вегенер И. *Задачи поиска.* Мир, Москва, 1982.

[3] Андреев А. Е. Метод бесповторной редукции синтеза самокорректирующихся схем. *ДАН СССР* (1985) **283**, №2, 265–269.

[4] Ахо А., Хопкрофт Дж., Ульман Дж. *Построение и анализ вычислительных алгоритмов.* Мир, Москва, 1979.

[5] Гаврилов Г. П., Сапоженко А. А. *Задачи и упражнения по курсу дискретной математики.* Наука, Москва, 1992.

[6] Гасанов Э. Э. Об одной математической модели информационного поиска. *Дискретная математика* (1991) **3**, № 2, 69–76.

[7] Гасанов Э. Э. Нижняя оценка сложности информационных сетей для одного класса задач информационного поиска. *Дискретная математика* (1992) **4**, № 3, 118–127.

[8] Гасанов Э. Э. Об одномерной задаче интервального поиска. *Дискретная математика* (1995) **7**, № 2, 40–60.

[9] Гасанов Э. Э. Нижняя оценка сложности информационных сетей для одного отношения частичного порядка. *Дискретная математика* (1996) **8**, № 4, 108–122.

[10] Гасанов Э. Э. Нижняя оценка сложности включающего поиска в классе древовидных схем. *Дискретная математика* (1998) **10**, № 1, 63–72.

[11] Гасанов Э. Э., Луговская Ю. П. Константный в худшем случае алгоритм поиска идентичных объектов. *Дискретная математика* (1999) **11**, № 4, 139–144.

[12] Ершов А. П. О программировании арифметических операторов. *ДАН СССР* (1958) **118**, 427–430.

[13] Ершов А. П. Операторные алгоритмы. III (Об операторных схемах Янова). *Проблемы кибернетики.* (1968), вып. 20, 181–200.

[14] Ким Д. П. *Методы поиска и преследования подвижных объектов.* Наука, Москва, 1989.

[15] Кнут Д. *Искусство программирования для ЭВМ. Сортировка и поиск.* **3**, Мир, Москва, 1978.

[16] Котов В. Е., Сабельфельд В. К. *Теория схем программ.* Наука, Москва, 1991.

[17] Кудрявцев В. Б., Алешин С. А., Подколзин А. С. *Введение в теорию автоматов.* Наука, Москва, 1985.

[18] Ли Д., Препарата Ф. Вычислительная геометрия. Обзор. *Кибернетический сб.* (1987) **24**, 5–96.

[19] Лупанов О. Б. О синтезе некоторых классов управляющих систем. *Проблемы кибернетики* (1963) **10**, 63–97.

[20] Мальцев А. И. *Алгоритмы и рекурсивные функции.* Наука, Москва, 1986.

[21] Мартин Дж. *Организация баз данных в вычислительных системах.* Мир, Москва, 1980.

[22] Минский М., Пейперт С. *Персептроны.* Мир, Москва, 1971.

[23] Мошков М. Ю. *Деревья решений. Теория и приложения.* Изд-во Нижнегородского ун-та, Нижний Новгород, 1994, 175 с.

[24] Полиа Г., Сеге Г. *Задачи и теоремы из анализа.* Наука, Москва, 1978.

[25] Препарата Ф., Шеймос М. *Вычислительная геометрия: Введение.* Мир, Москва, 1989.

[26] Решетников В. Н. Алгебраическая теория информационного поиска. *Программирование* (1979), № 3, 68–74.

[27] Селтон Г. *Автоматическая обработка, хранение и поиск информации.* Советское радио, Москва, 1973.

[28] Солтон Дж. *Динамические библиотечно-информационные системы.* Мир, Москва, 1979.

[29] Страуструп Б. *Язык программирования Си++.* Радио и связь, Москва, 1991.

[30] Хеллман О. *Введение в теорию оптимального поиска.* Наука, Москва, 1985.

[31] Черный А. И. *Введение в теорию информационного поиска.* Наука, Москва, 1975.

[32] *Энциклопедия кибернетики.* Киев, 1975, т. 1. Статьи: "Информационно-поисковая система"(с.359), "Информационно-поисковая система документальная"(с.359–400).

[33] Яблонский С. В. *Введение в дискретную математику.* Наука, Москва, 1979.

[34] Янов Ю. И. О логических схемах алгоритмов. *Проблемы кибернетики.* (1958), вып. 1, 75–127.

[35] Bayer R., McCreight E. M. Organization and Maintenance of Large Ordered Indexes. *Acta Informatica* (1972) **1**, № 3, 173–189.

[36] Ben-Or M. Lower bounds for algebraic computation trees. *Proc. 15th ACM Annu. Symp. Theory Comput.* (Apr. 1983) 80–86.

[37] Bentley J. L. Multidimensional binary search trees used for associative searching. *Commun. Ass. Comput. Mach.* (Sept. 1975), **18** 509–517.

[38] Bentley J. L., Friedman J. H. Data structures for range searching. *Comput. Surveys* (1979), **11** 397–409.

[39] Bentley J. L., Maurer H. A. Efficient worst-case data structures for range searching. *Acta Inform.* (1980), **13** 155–168.

[40] Bentley J. L., Saxe J. B. Decomposable searching problems. I. Static-to-dynamic transformation. *J. Algorithms* (1980), v. 1, 301–358.

[41] Bentley J. L., Stanat D. F. Analysis of range searching in quad trees. *Inform. Processing Lett.* (1975), **3** 170–173.

[42] Bolour A. Optimal retrieval algorithms for small region queries. *SIAM J. Comput.* (1981) **10**, 721–741.

[43] Borodin A. B. Computational complexity — Theory and practice. *Currents in Theory of Computings.* Englewood Cliffs, NJ: Printice-Hall, 1973, 35–89.

[44] Bottenbruch H. Structure and use of ALGOL 60. *JACM* (1962) **9**, 161-221.

[45] Chazelle B. M. Filtering search: a new approach to query-answering. *Proc. 24th IEEE Annu. Symp. Found. Comput. Sci.* (Nov. 1983), 122–132.

[46] Dobkin D. P., Lipton R. J. On the complexsity of computations under varying sets of primitives. *J. Comput. Syst. Sci.* (1979) **18**, 86–91.

[47] Dumey A. Indexing for Rapid Random Access Memory Systems. *Computers and Automation.* (1956) **4**, № 12, 6–9.

[48] Fredman M. L. A lower bound of the complexity of ortogonal range queries. *J. ACM.* (1981) **28**, 696–705.

[49] Harper L. H. Optimal numbering and isoperimetric problem on graphs. *J. Combin. Theory* (1966) **1**, № 3, 385–393.

[50] Hibbard T. N. Some Combinatorial Properties of Certain Trees with Applications to Searching and Sorting. *J. ACM* (1962) **9**, № 1, 13–28.

[51] Katona G. O. H. A Theorem on finite sets. *Theory of Graphs, Proc. Colloq. held at Tihany.* Hungary, (1966), 187–207.

[52] Katona G. O. H. The Hamming-sphere has minimum boundary *Studia Scient. Math. Hungarica* (1975) **10**, 131–140.

[53] Knuth D. E. Optimum Binary Search Trees. *Acta Informatica* (1971) **1**, 14–25.

[54] Kruskal F. B. The number of simplices in compex. *Mathematical Optimization Techniques* (1963), 251–278.

[55] Lee D. T., Wong C. K. Worst case analysis for region and partial region searches in multidimensional binary search trees and balansed quad trees. *Acta Informatica* (1977) **9** 23–29.

[56] Lueker G. S., Willard D. E. A data structure for dynamic range queries. *Inform. Processing Lett.* (Dec. 1982) **15**, № 5, 209–213. *Commun. ACM* (1971) **14**, № 4, 228–239.

[57] Maurer W. D., Lewis T. G. Hash table methods. *Commputing Surveys* (1975) **7**, 5–20.

[58] Peterson W. W. Addressing for Random Access Storage. *IBM J. Research & Development* (1957) **1**, 130–146.

[59] Preparata F. P., Shamos M. J. *Computational Geometry.* Springer-Verlag, New York, 1985.

[60] Rabin M. O. Proving simultaneous positivity o linear forms. *J. Comput. Syst. Sci.* (1972) **6**, 639–650.

[61] Reingold E. M. On the optimality of some set algorithms. *J. ACM* (Oct. 1972) **19**, № 4, 649–659.

[62] Schay G.,Jr, Spruth W. G. B. Analysis of a File Addressing Method. *Comm. ACM* (1962) **5**, 459–462.

[63] Steele J. M., Yao A. C. Lower bounds for algebraic decision trees. *J. Algorith.* (1982) **3**, 1–8.

[64] Turing A. M. On Computable Numbers, whis an Application to the Entsheidungsproblem. *Proc. London Math. Soc.* (1937) **42**, Ser 2, 230–235.

[65] Ullman J. D. A Note on the Efficiency of Hashing Functions. *JACM* (1972) **19**, 569–575.

[66] Yao A. C., Rivest R. L. On the polyhedral decision problem. *SIAM J. Comput.* (1980) **9**, 343–347.

Э.Э.Гасанов

Теория сложности информационного поиска. Учебное пособие
М., Издательство Центра прикладных исследований при механико-математическом факультете МГУ, 144 стр.

Оригинал макет изготовлен издательской группой механико-математического факультета МГУ

Подписано в печать 05.05.2005 г.
Формат 60×90 1/16. Объем 9,0 п.л.
Заказ 8 Тираж 250 экз.

Издательство ЦПИ при механико–математическом факультете МГУ
г. Москва, Воробьевы горы.
Лицензия на издательскую деятельность ИД № 04059 от 20.02.2001 г.

Отпечатано на типографском оборудовании механико-математического факультета