ФЕДЕРАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЦЕНТР «ИНФОРМАТИКА И УПРАВЛЕНИЕ» РОССИЙСКОЙ АКАДЕМИИ НАУК

На правах рукописи

Драгунов Никита Аркадьевич

Поиск частых (нечастых) элементов декартова произведения конечных частичных порядков и приложения

Специальность 1.2.3
«Теоретическая информатика, кибернетика»

Диссертация на соискание учёной степени
кандидата физико-математических наук

Научный руководитель:
д.ф.-м.н., доцент
Дюкова Елена Всеволодовна

Москва
2026

Оглавление

Введение

Актуальность темы исследования. Задачи поиска частых и нечастых элементов в данных являются одними из центральных задач анализа информации и имеют широкий спектр приложений, в частности, в дискретной математике, машинном обучении и при нахождении ассоциативных правил. Во многих прикладных областях данные естественным образом упорядочены, однако традиционные алгоритмы решения рассматриваемых задач порядок не учитывают. В связи с этим возникает необходимость разработки эффективных методов поиска частых и нечастых элементов для случая, когда данные представлены в виде декартова произведения конечных частично упорядоченных множеств.

Конечное множество P называется *частично упорядоченным*, если на элементах этого множества задано бинарное отношение $\preccurlyeq$, обладающее свойствами:

- рефлексивности: $x \preccurlyeq x$, $\forall x \in P$;
- антисимметричности: если $x \preccurlyeq y$ и $y \preccurlyeq x$, то $x = y$, $\forall x \in P$, $\forall y \in P$;
- транзитивности: если $x \preccurlyeq y$ и $y \preccurlyeq z$, то $x \preccurlyeq z$, $\forall x \in P$, $\forall y \in P$, $\forall z \in P$.

Если $x \preccurlyeq y$, $x \in P$, $y \in P$, то говорят, что элемент y *следует* за элементом x или что x *предшествует* y. Элементы x и y называются *сравнимыми*, если либо y следует за x, либо x следует за y. Иначе x и y *несравнимы*. Множество P – *цепь* (*антицепь*), если любые два элемента в P сравнимы (несравнимы). Элемент x из P называется *максимальным* (*минимальным*), если не существует другого элемента из P, следующего за x (предшествующего x).

Задачи поиска частых и нечастых элементов в частично упорядоченных данных рассматриваются в следующей постановке. Дана база данных $D(P)$, представляющая собой совокупность некоторых не обязательно различных элементов из P. Элементы базы данных называются *транзакциями*. *Частотой* элемента $x \in P$ называется величина $\nu(x)$, равная числу транзакций в $D(P)$, следующих за x. Задан некоторый порог $s \in \mathbb{N}$, $\mathbb{N}$ – множество натуральных чисел.

Если $v(x) \geq s$, то элемент x называется s *-частым*. Иначе x называется s*-нечастым* элементом. Требуется по заданным $D(P)$ и s перечислить частые (нечастые) элементы.

На практике вместо перечисления всех частых (нечастых) элементов, как правило, строят более компактное множество максимальных частых (минимальных нечастых) элементов [8, 51, 55, 74, 75]. Если $x \in P$ является s -частым элементом и в P не существует другого s-частого элемента, следующего за x, то x называется *максимальным* s *-частым* элементом. Если $x \in P$ является s -нечастым элементом и в P не существует другого s -нечастого элемента, предшествующего x, то x называется *минимальным* s *-нечастым* элементом. Далее в работе s -частые и s -нечастые элементы для краткости называются просто частыми и нечастыми элементами соответственно.

Степень разработанности темы. Рассматриваемые задачи впервые возникли в контексте анализа потребительской корзины при поиске ассоциативных правил [47, 49] и наиболее изучены в случае бинарной информации. Одними из наиболее известных алгоритмов в этой области являются алгоритмы Apriori, DepthProject и FP-Growth [48, 50, 51, 67].

В [63] поставлена имеющая большое число приложений и важная с теоретической точки зрения задача совместного перечисления максимальных частых и минимальных нечастых элементов множества P при условии, что P представлено в виде декартова произведения конечных частично упорядоченных множеств $P_1, \ldots, P_n, n \geq 2$ (запись $P = P_1 \times \ldots \times P_n$). Считается, что элемент $y = (y_1, \ldots, y_n)$ из P следует за элементом $x = (x_1, \ldots, x_n)$ этого множества, если y_i следует за x_i при $i = \overline{1,n}$. Показано, что при решении указанной задачи появляется необходимость рассматривать одну из центральных труднорешаемых перечислительных проблем дискретной математики – дуализацию над произведением частичных порядков. Поясним сказанное.

Пусть R – некоторое непустое подмножество частично упорядоченного множества P . Через R^+ обозначается множество всех элементов из P следующих

хотя бы за одним элементом из R , а через R^- – множество всех элементов из P предшествующих хотя бы одному элементу из R . Множество $I(R^+)$, состоящее из всех максимальных элементов множества $P\backslash R^+$, называется *максимальным независимым* от R . Множество $I(R^-)$, состоящее из всех минимальных элементов множества $P\backslash R^-$, называется *минимальным независимым* от R . Каждая из задач построения $I(R^+)$ и $I(R^-)$ называется дуализацией над произведением частичных порядков [62].

Особый интерес представляет случай, когда $P_i = \{0, 1, \dots, k-1\}$, $k \geq 2$, при $i = \overline{1,n}$, и в каждом P_i задан порядок $0 \preccurlyeq 1 \preccurlyeq \cdots \preccurlyeq k-1$. Тогда множество P называется k *-значным* n *-мерным кубом* со стандартным линейным порядком и обозначается далее через E_k^n . Множество E_2^n называется *булевым кубом*.

Если $P = E_2^n$, а R – множество всех элементов из P , на которых монотонная булева функция F обращается в ноль, то задача нахождения $I(R^-)$ – это труднорешаемая проблема построения сокращенной дизъюнктивной нормальной формы функции F , заданной конъюнктивной нормальной формой, называемая монотонной дуализацией. Проблема поставлена в середине 1950-х годов чл.-корр. РАН С. В. Яблонским в связи с построением минимальных тестов для контактных схем [26, 29 – 30, 34, 46, 63]. Монотонная дуализация допускает другие постановки, и часто формулируется как задача перечисления неприводимых покрытий булевой матрицы или перечисления минимальных вершинных покрытий гиперграфа [71]. Её важность обусловлена большим числом приложений среди которых следует особо выделить машинное обучение (построение логических процедур классификации) [20 – 22, 25, 33, 39].

В теории алгоритмической сложности дискретных задач эффективность алгоритмов для перечислительных задач принято оценивать временем выполнения одного шага, то есть временем нахождения очередного решения. Алгоритм с полиномиальной временной оценкой в «худшем случае» (для самой сложной индивидуальной задачи) считается наиболее эффективным и называется алгоритмом с полиномиальной задержкой [69]. Для монотонной дуализации

построить такие алгоритмы удалось лишь для небольшого числа частных случаев. В связи с этим выделились два основных подхода, применимых в целом к широкому классу дискретных перечислительных задач. Первый подход основан на построении инкрементальных алгоритмов, сложность шага которых зависит от размера входных данных и числа уже найденных решений [45, 54, 62, 64]. В рамках инкрементального подхода Л. Г. Хачияну с соавторами удалось построить алгоритм монотонной дуализации с квазиполиномиальной задержкой в «худшем случае», то есть с временем нахождения очередного решения, превышающим полиномиальное, но существенно меньшим экспоненциального [45]. Второй подход предполагает построение асимптотически оптимальных алгоритмов [20 – 22, 25 – 31, 33 – 35, 57]. Этим алгоритмам разрешено делать лишние полиномиальные шаги, не приводящие к новым решениям при условии, что «почти всегда» (для почти всех индивидуальных задач) число таких шагов должно быть «мало» (должно иметь более низкий порядок роста) по сравнению с числом всех решений задачи. При этом проверка того, является ли шаг лишним, должна осуществляться за полиномиальное от размера входных данных время. Подход предложен Е. В. Дюковой в 1977 г. [20]. В отличие от инкрементальных алгоритмов монотонной дуализации асимптотически оптимальные алгоритмы имеют практическое значение и на сегодняшний день являются лидерами по скорости счёта [27, 29, 30, 34].

Обозначим через X_{max} и Y_{min} соответственно множества всех максимальных частых и минимальных нечастых элементов частичных порядков. Существует достаточно очевидный последовательный способ перечисления этих множеств, основанный на свойстве двойственности: $I(X_{max}^{-}) = Y_{min}$ и $I(Y_{min}^{+}) = X_{max}$. То есть сначала находится одно из множеств X_{max} или Y_{min} , а затем второе множество строится путем дуализации первого. Экспериментальное исследование показывает, что подход является эффективным только в случае, когда одно из множеств частых или нечастых элементов существенно меньше другого [13].

В [63] рассмотрен важный с теоретической точки зрения инкрементальный подход к совместному перечислению максимальных частых и минимальных нечастых элементов частичных порядков. Однако, в [63] не приведена реализация предложенного алгоритма и не представлено экспериментальное сравнение с альтернативными методами. Данный алгоритм реализован и экспериментально исследован в рамках настоящей диссертационной работы. Результаты исследования показывают практическую неприменимость метода, так как сложность каждого шага алгоритма возрастает с увеличением числа уже найденных решений.

Таким образом, актуальной является разработка эффективных в «типичном случае» (для почти всех вариантов задачи) алгоритмов совместного перечисления максимальных частых и минимальных нечастых элементов декартова произведения частично упорядоченных множеств.

Как уже было отмечено ранее, задача совместного перечисления максимальных частых и минимальных нечастых элементов частичных порядков имеет большое число практических приложений. В частности, она связана с важной в дискретной математике задачей расшифровки двузначной монотонной функции, которая рассматривается в следующей постановке.

Исследуется функция f, определенная на элементах декартова произведения частично упорядоченных множеств $P = P_1 \times \ldots \times P_n$, $n \geq 1$, и принимающая два значения 0 и 1. Функция f называется *монотонной*, если для любых двух элементов x и y из P таких, что $x \preccurlyeq y$, выполнено $f(x) \leq f(y)$. Монотонная функция f задается при помощи некоторого оператора B (оракула), который для любого $x \in P$ возвращает значение функции $f(x)$. Если $f(x) = 0$, то элемент x называется *нулем* функции f, если же $f(x) = 1$, то элемент x называется *единицей* функции f. Требуется путем «минимального» числа обращений к оператору B найти все нули и единицы функции f.

Традиционный подход к решению задачи расшифровки нацелен на случай функции, заданной на E_k^n, и основан на построении оптимального по Шеннону алгоритма [1, 2, 38]. Согласно данному подходу, сложность алгоритма расшифровки

следует оценивать числом обращений к оракулу B в «худшем случае» (то есть для самого сложного варианта задачи). Пусть V – множество всех двузначных монотонных функций, определенных на P ; A – некоторый алгоритм, выполняющий расшифровку функций из V . Обозначим через $t_A(f)$ общее число обращений к оператору B алгоритма A при расшифровке функции $f \in V$. Под сложностью алгоритма A по Шеннону (сложностью в худшем случае) понимается величина $max_{f \in V} t_A(f)$, где максимум берется по всем функциям из V . Алгоритм называется *оптимальным по Шеннону* на V, если его сложность минимальна среди всех алгоритмов, выполняющих расшифровку функций из V. Задача оптимальной по Шеннону расшифровки для монотонной функции, заданной на E_2^n (монотонная булева функция), поставлена В. К. Коробковым в 1965 году [38] и решена Ж. Анселем в 1968 году [2]. В 1976 г. В. Б. Алексеевым исследован случай функции, заданной на E_k^n , и предложен алгоритм, который является оптимальным по Шеннону в случае $k = 2$ и «близок» по сложности к оптимальному по Шеннону в случае $k > 2$ [1].

Методы, ориентированные на «худший случай», имеют существенные ограничения для практического применения. Это делает актуальной разработку альтернативных подходов к расшифровке, эффективных для большинства встречающихся на практике вариантов задачи.

Еще одним важным приложением методов поиска частых и нечастых элементов в данных является классификация по прецедентам. Задача рассматривается в стандартной для логического подхода постановке. Исследуется множество объектов M , которое представимо в виде объединения непересекающихся подмножеств $K_1, \dots, K_l$, называемых *классами*. Объекты из M описываются в системе числовых *признаков* $x_1, \dots, x_n$. Предполагается, что каждый признак x_i , $i = \overline{1,n}$, имеет ограниченное множество допустимых значений N_i , элементы которого кодируются целыми числами. На множестве N_i задается частичный порядок. Множество M фактически определяется как $M = N_1 \times \dots \times N_n$. Даны примеры объектов из каждого класса, называемые *прецедентами*. Требуется

на основе анализа множества прецедентов построить алгоритм, умеющий по признаковому описанию объекта из M определять его класс.

При конструировании логических классификаторов большое внимание уделяется вопросам синтеза *корректных алгоритмов*, то есть алгоритмов, не ошибающихся на обучающей выборке. Обучение классификатора сводится к поиску в исходных данных информативных фрагментов описаний прецедентов. Такие фрагменты, называемые корректными элементарными классификаторами, позволяют различать объекты из разных классов и, как правило, имеют содержательное описание в терминах той прикладной области, в которой решается задача.

Пусть $H = \{x_{j_1}, \dots, x_{j_r}\} \subseteq \{x_1, \dots, x_n\}$, $j_1 < \dots < j_r$, – набор из r различных признаков; $\sigma = (\sigma_1, \dots, \sigma_r)$, $\sigma_i \in N_i$, $i = \overline{1,r}$, – набор допустимых значений признаков из H . Пара (σ, H) называется *элементарным классификатором (ЭК) ранга* r . Объект $S = (a_1, \dots, a_n) \in M$ *содержит* ЭК (σ, H) , если $a_{j_i} \leqslant \sigma_i$, $i = \overline{1,r}$.

Пусть $K \in \{K_1, \dots, K_l\}$, $\overline{K} = \{K_1, \dots, K_l\} \setminus \{K\}$, $R(K)$ и $R(\overline{K})$ – множество всех прецедентов из K и из $\overline{K}$ соответственно. ЭК (σ, H) называется *корректным* для класса K , если не существует пары прецедентов $S' \in R(K)$, $S'' \in R(\overline{K})$ таких, что S' и S'' содержат (σ, H) . Корректный ЭК (σ, H) называется *представительным* для класса K, если хотя бы один прецедент из $R(K)$ содержит (σ, H).

Будем считать, что множество N_i , $i = \overline{1,n}$, содержит *наибольший* элемент h_i, который следует за любым другим элементом этого множества. Если такого элемента нет, то множество N_i можно дополнить таким элементом. Пусть $S = (a_1, \dots, a_n)$ и $S^* = (b_1, \dots, b_n)$ – объекты из M. Будем считать, что элемент S^* следует за элементом S , если $a_j \leqslant b_j$ при $j = 1, \dots, n$. В этом случае любому ЭК (σ, H) можно поставить в соответствие элемент $\tilde{\sigma} = (\tilde{\sigma}_1, \dots, \tilde{\sigma}_n)$ из M , такой что: 1) $\tilde{\sigma}_{j_i} = \sigma_i$, $i = \overline{1,r}$; 2) $\tilde{\sigma}_i = h_i$ при $i = \overline{1,n}$, $i \notin \{j_1, \dots, j_r\}$. Тогда считается, что ЭК (σ, H) следует за ЭК (σ', H') , если $\tilde{\sigma}'$ следует за $\tilde{\sigma}$ согласно порядку, заданному на элементах из M .

ЭК (σ, H) называется s *-частым* в K, если не менее s прецедентов из $R(K)$ содержат (σ, H) . ЭК (σ, H) называется *максимальным* s *-частым* в K, если он s -частый в K и не существует другого s -частого ЭК в K , следующего за данным. ЭК (σ, H) называется s *-нечастым* в $\overline{K}$, если менее s прецедентов из $R(\overline{K})$ содержат (σ, H). ЭК (σ, H) называется *минимальным* s *-нечастым* в $\overline{K}$, если он s -нечастый в $\overline{K}$ и не существует другого s -нечастого в $\overline{K}$ ЭК, предшествующего данному. Фактически, представительный ЭК для класса K – это 1 -частый в K и 1 -нечастый в $\overline{K}$ элементарный классификатор.

Отечественное направление логической классификации в основном представлено методами, именуемыми процедурами корректного голосования (Correct Voting Procedures или CVP) [3, 7, 20 – 22, 25, 33, 39, 46]. Фундаментальную роль в создании этого направления сыграли научные школы чл.-корр. РАН С. В. Яблонского и акад. РАН Ю. И. Журавлева. Направление развивается с 70-х годов прошлого века. В ряде публикаций отечественных и зарубежных авторов предложены и развиты два других направления логической классификации, а именно логический анализ данных (Logical Analysis of Data или LAD) [36, 41, 53, 54, 70] и анализ формальных понятий (Formal Concept Analysis или FCA) [40, 43, 44, 65, 68, 73].

В CVP наиболее информативным считается такой представительный ЭК для класса K , который является минимальным 1 -нечастым в $\overline{K}$ (это тупиковый представительный ЭК класса K). В LAD ищутся так называемые максимальные логические закономерности класса K . Это представительные ЭК для K , которые содержатся в наибольшем числе прецедентов из $R(K)$. Направление FCA в основном представлено классификаторами, вдохновленными идеями ДСМ-метода В. К. Финна [40, 43, 44]. Базовый FCA-классификатор нацелен на построение множества всех таких представительных ЭК для класса K , каждый из которых является максимальным s -частым для некоторого $s \geq 1$. В [28] показано, что во всех трех направлениях на этапе обучения логического классификатора для каждого класса K фактически задается некоторый частичный порядок на

множестве представительных ЭК и строится множество максимальных относительно заданного порядка ЭК.

В каждом из этих направлений логической классификации на этапе обучения возникают сложные в вычислительном плане задачи. Для CVP – это задача монотонной дуализации, для LAD – это, в основном, оптимизационные задачи линейного программирования, для FCA – перечислительные задачи дискретной математики, которые являются менее сложными, чем задачи, возникающие в CVP и LAD. Решение этих задач требует значительных вычислительных ресурсов, что делает традиционные методы логической классификации слабо применимыми для задач большой размерности. В связи с этим актуальной является разработка более эффективных алгоритмов логической классификации, не уступающих по качеству распознавания традиционным моделям.

В CVP и LAD решающее правило основано на процедуре «голосования». Пусть $T(K)$ – множество всех искомых ЭК класса K ; $|T(K)|$ – мощность множества $T(K)$; S – распознаваемый объект; $W_{(\sigma,H)}^{K}$ – число прецедентов из класса K , содержащих ЭК (σ, H) ; $\Omega(\sigma, H, S)$ – величина, равная 1, если S содержит (σ, H), и равная 0 иначе. Для каждого класса K вычисляется оценка $\Gamma(S, K)$ принадлежности S рассматриваемому классу согласно формуле

$$\Gamma(S,K) = \frac{1}{|T(K)|} \sum_{(\sigma,H)\in T(K)} W_{(\sigma,H)}^{K}\, \Omega(\sigma, H, S)\,.$$

Объект S относится к классу с наибольшей оценкой. В случае, если таких классов несколько, то происходит отказ от распознавания. Базовая версия классификатора из направления FCA использует более строгое решающее правило, что приводит к большому числу отказов от классификации. Замена в FCA-классификаторе решающего правила на процедуру голосования приводит к повышению качества распознавания [32].

Традиционные схемы логической классификации ориентированы исключительно на случай, когда множество значений каждого признака представляет собой конечную антицепь и для сравнения целочисленных

признаковых описаний объектов используется отношение равенства. Вопросы модификации процедур CVP, LAD и FCA для корректного решения задачи классификации частично упорядоченных целочисленных данных общего вида рассматривались в ряде работ [29 – 32, 52]. Особое внимание уделялось случаю, когда на множествах значений признаков заданы конечные линейные порядки, т.е. указанные множества – конечные цепи.

Целью настоящей работы является разработка вычислительно эффективных методов логического анализа информации, представленной в виде декартова произведения частично упорядоченных конечных множеств (произведения частичных порядков), и применения этих методов в области дискретной математики и машинного обучения. Поставлены следующие **задачи**.

1. Разработать эффективный алгоритм совместного перечисления максимальных частых и минимальных нечастых элементов декартова произведения частично упорядоченных множеств.

2. Предложить ориентированный на «типичный случай» алгоритм расшифровки двузначной монотонной функции, определенной на декартовом произведении конечных частично упорядоченных множеств.

3. Построить быстрые алгоритмы логической классификации, базирующиеся на поиске часто встречающихся ЭК в описаниях прецедентов класса и ориентированные на обработку данных, представленных в виде декартова произведения конечных частично упорядоченных множеств. Получить теоретические оценки типичного числа и ранга искомых ЭК.

Методология и методы исследования. В работе использованы методы дискретной математики, комбинаторики, машинного обучения, теории вычислительной сложности и теории вероятности. Экспериментальное исследование проведено с использованием программно-алгоритмического комплекса, разработанного автором.

Опишем полученные в работе результаты. Предложен и исследован новый последовательно-совместный подход к перечислению максимальных частых X_{max}

и минимальных нечастых Y_{min} элементов частичных порядков, который является синтезом последовательного и совместного методов [10, 13].

Для случая, когда данные представлены в виде произведения конечных цепей, приведены результаты экспериментального сравнения последовательно-совместного подхода с ранее известными последовательным и совместным [63] подходами, а также с независимым способом построения множеств X_{max} и Y_{min} , не требующим решения задачи дуализации. Задача дуализации над произведением частичных порядков решается с помощью асимптотически оптимального алгоритма дуализации цепей RUNC-M+ [29, 30]. Результаты исследования свидетельствуют о том, что последовательно-совместный метод требует меньших временных затрат по сравнению с другими рассмотренными методами в случае, когда мощность множества частых элементов мало отличается от мощности множества нечастых элементов. Иначе выигрывает последовательный поиск. Наихудшие показатели у независимого перечисления множеств X_{max} и Y_{min} с использованием в качестве базового алгоритма Apriori.

В диссертационной работе поставлена задача «асимптотически оптимальной» расшифровки двузначной монотонной функции f , определенной на декартовом произведении конечных частично упорядоченных множеств. В рамках поставленной задачи исследуется вопрос о построении алгоритма расшифровки, эффективного в «типичном случае». Показана связь между поставленной задачей и поиском максимальных частых и минимальных нечастых элементов частичных порядков. На базе последовательно-совместного подхода к поиску максимальных частых и минимальных нечастых элементов частичных порядков построен требуемый алгоритм расшифровки функции f . В ходе экспериментального исследования рассмотрен случай функции, заданной на E_k^n , и доказана эффективность предлагаемого алгоритма.

Установим связь между задачей совместного перечисления X_{max} и Y_{min} и задачей расшифровки функции f. Введем ряд необходимых определений. Ноль двузначной монотонной функции f называется *верхним*, если он не предшествует

никакому другому нулю этой функции. Единица функции f называется *нижней*, если она не следует ни за какой другой единицей этой функции. Для расшифровки функции f достаточно построить множества всех ее верхних нулей и нижних единиц.

Пусть дана база данных $D(P)$ и порог $s \leq |D(P)|$, $s \in \mathbb{N}$ (здесь и далее $|D(P)|$ – это число транзакций в базе данных). На множестве P определим двузначную монотонную функцию $f_{D,s}$, которая принимает значение 1 на частых элементах и 0 – на нечастых. Фактически, функция задаётся при помощи оператора B, который для произвольного $x \in P$ возвращает значение функции $f_{D,s}$ путем вычисления частоты встречаемости элемента x.

Нетрудно видеть, что нижние единицы (верхние нули) функции $f_{D,s}$ взаимно однозначно соответствуют минимальным нечастым (максимальным частым) элементам множества P. Таким образом, для решения задачи расшифровки произвольной двузначной монотонной логической функции может быть применен предложенный в работе последовательно-совместный алгоритм перечисления множеств X_{max} и Y_{min}.

На базе последовательно-совместного алгоритма поиска максимальных частых и минимальных нечастых элементов декартова произведения цепей построен новый алгоритм расшифровки функции f, нацеленный на «типичный случай». Предложенный алгоритм использует последовательно-совместный метод для перечисления верхних нулей и нижних единиц функции f. Экспериментально продемонстрирована эффективность предлагаемого алгоритма «асимптотически оптимальной» расшифровки при задании функции на E_k^n: в случае больших значений k и n предложенный метод работает существенно быстрее ориентированного на «худший случай» алгоритма Алексеева [1] и требует меньшего числа обращений к B. В экспериментальном исследовании задача дуализации решается с помощью асимптотически оптимального алгоритма дуализации цепей RUNC-M+ [29, 30].

В диссертационной работе рассмотрена также возможность повышения качества и скорости работы логических классификаторов на основе применения методов поиска в описаниях прецедентов каждого класса K часто встречающихся фрагментов специального вида и с последующим отбором среди них тех ЭК, которые являются представительными для K . Исследованы случаи, когда на множествах значений признаков частичные порядки не заданы, то есть описания изучаемых объектов – элементы декартова произведения антицепей, и, когда признаковые описания объектов – элементы декартова произведения цепей.

Для случая декартова произведения антицепей построен алгоритм REC, перечисляющий такие представительные ЭК для K, каждый из которых имеет ранг r ($r \geq 1$) и принимает значение 1 на не менее, чем r прецедентах класса K . Такие ЭК называются *правильными представительными элементарными классификаторами*. Фактически, правильный представительный ЭК – это r -частый элемент в $R(K)$ и 1 -нечастый элемент в $R(\overline{K})$.

Для анализа сложности алгоритмов логической классификации в направлении CVP традиционно исследуют метрические свойства искомых множеств представительных ЭК и получают асимптотические оценки типичного числа таких ЭК. Впервые подобные результаты получены в работе [42], где рассматривались так называемые тестовые алгоритмы классификации. Техника получения оценок была существенно развита Е. В. Дюковой и ее учениками [19 – 25, 27, 30, 33, 35, 58] а также А. Е. Андреевым [39], А. А. Кибкало [37].

Введем ряд необходимых понятий. Обозначим через L_K матрицу, строками которой являются признаковые описания прецедентов класса K . Квадратная подматрица матрицы L_K называется *правильной*, если она состоит из одинаковых строк. Правильные ЭК порождаются правильными подматрицами матрицы L_K .

Пусть m_1 – число прецедентов в $R(K)$; $M_{m_1 n}^k$ – множество всех матриц размера $m_1 \times n$ с элементами из $E_k = \{0, 1, \dots, k-1\}$; $S(L_K)$, $L_K \in M_{m_1 n}^k$, – множество правильных подматриц в матрице L_K ; ϕ_1 – интервал $(0\ ,\ r_1)$, где $r_1 =$

$0.5log_k m_1 n - 0.5log_k log_k m_1 n + log_k log_k log_k n$; $b_n \sim c_n$, $n \to \infty$, означает, что $lim_{n\to\infty} b_n/c_n = 1$.

В приведенной ниже Теореме 1 получены оценки типичного числа и типичного порядка правильных подматриц целочисленной матрицы. Выявление типичной ситуации связано с высказыванием вида «для почти всех матриц L_K из $M_{m_1 n}^k$ при $n \to \infty$ выполнено $F_1(L_K) \sim F_2(L_K)$» (здесь $F_1(L_K)$ и $F_2(L_K)$ – два функционала, заданные на матрицах из $M_{m_1 n}^k$). Данное высказывание означает, что существуют две положительные бесконечно убывающие функции $\alpha(n)$ и $\beta(n)$, такие, что для всех достаточно больших n имеет место

$$1 - |M|/|M_{mn}^k| \leq \alpha(n) ,$$

где M – множество таких матриц L_K в M_{mn}^k , для которых

$$1 - \beta(n) < F_1(L_K)/F_2(L_K) < 1 + \beta(n) .$$

Теорема 1. Если $n^a \leq m_1 \leq k^n$, $a > 1$, то для почти всех матриц L_K из $M_{m_1 n}^k$ справедливо

$$|S(L_K)| \sim \sum_{r\in\phi_1} C_n^r C_{m_1}^r k^{r-r^2} \text{ , при } n \to \infty ,$$

и порядки почти всех правильных подматриц из $S(L_K)$ принадлежат интервалу ϕ_1 .

Так как правильные представительные ЭК класса K порождаются правильными подматрицами из $S(L_K)$, то оценка типичного числа правильных подматриц является верхней асимптотической оценкой числа правильных представительных ЭК. Оценка типичного порядка правильной подматрицы свидетельствует о том, что в случае, когда число прецедентов существенно больше числа признаков, типичный ранг правильного представительного ЭК не превосходит r_1 .

В [19] получены асимптотические оценки типичного числа и оценки типичного ранга правильных ЭК при иных ограничениях: 1) ${m_1}^a \leq n \leq k^{{m_1}^\beta}$, $a > 1$, $\beta < 1$; 2) $n \leq m_1 \leq k^{n^\beta}$, $\beta < 1/2$.

Экспериментально показано, что осуществляемый на этапе обучения поиск правильных представительных ЭК требует меньших временных затрат по

сравнению с решением задач, возникающих при реализации классических моделей логических классификаторов из направления CVP. В экспериментах ранг искомых правильных ЭК в алгоритме REC выбирается с использованием полученной оценки типичного порядка правильной подматрицы.

Следует подчеркнуть, что алгоритм REC осуществляет поиск правильных представительных ЭК класса K путем анализа, в первую очередь, прецедентов из $R(K)$, тогда как алгоритмы CVP, основанные на голосовании по тупиковым представительным ЭК, работают в основном с прецедентами из $R(\overline{K})$. В задачах с числом классов более двух мощность $R(K)$ обычно существенно меньше мощности $R(\overline{K})$, что делает поиск правильных ЭК в этом случае менее трудоёмким по сравнению с поиском тупиковых ЭК. Более того, сравнение полученных оценок типичного числа правильных представительных ЭК с известными оценками числа тупиковых представительных ЭК [27] показывает, что даже при равных мощностях $R(K)$ и $R(\overline{K})$ число правильных ЭК существенно меньше числа тупиковых. Наконец, в [23] доказано, что алгоритм REC является асимптотически оптимальным. Таким образом, согласуются теоретические и экспериментальные выводы об эффективности REC.

В диссертационной работе разработана модификация алгоритма REC для работы с данными, представленными в виде совокупности элементов декартова произведения конечных цепей, названная REC+. Классификатор REC+ работает по схеме алгоритма REC. Эффективность предлагаемого алгоритма обоснована теоретически (Теоремы 2, 3) и экспериментально на реальных данных. Приводимые ниже в Теоремах 2 и 3 оценки получены в предположении, что признак x_j принимает значения из E_k , $k \geq 2$, $j = \overline{1,n}$, и элементы в E_k линейно упорядочены в порядке возрастания.

Пусть $\mathfrak{U}(L_K)$, где $L_K \in M_{m_1 n}^k$, – это множество всех правильных ЭК в $R(K)$; $d = k/(k-1)$; $\sigma \in E_{k-1}^r$, $\sigma = (\sigma_1, \dots, \sigma_r)$; $Q_r(\sigma) = (\sigma_1 + 1)^r \times \dots \times (\sigma_r + 1)^r$; $r_2 =]\log_d m_1 + \log_d \log_d m_1[$, здесь $]q[$ – наименьшее целое, превосходящее q ;

$r_3 =]0.5\log_d m_1 n - \ 0.5\log_d \log_d m_1 n + \log_d \log_d log_d n[$; φ_2 – интервал $[1, r_2]$, φ_3 – интервал $[1, r_3]$; $b_n \lesssim c_n$, $n \to \infty$, означает, что $lim_{n\to\infty} b_n/c_n \leq 1$.

Теорема 2. Если $n \leq m_1$, то для почти всех матриц L_K из $M^k_{m_1 n}$ верно

$$|\mathfrak{A}(L_K)| \lesssim \sum_{r\in\varphi_2} \sum_{\sigma\in E^r_{k-1}} Q_r(\sigma) C^r_n C^r_{m_1} k^{-r^2} \text{ , при } n \to \infty ,$$

и ранги почти всех правильных ЭК из $\mathfrak{A}(L_K)$ принадлежат интервалу φ_2 .

Теорема 3. Если $m_1^\alpha \leq n \leq d^{m_1}$, $\alpha > 1$, то для почти всех матриц L_K из $M^k_{m_1 n}$ верно

$$|\mathfrak{A}(L_K)| \sim \sum_{r\in\varphi_3} \sum_{\sigma\in E^r_{k-1}} Q_r(\sigma) C^r_n C^r_{m_1} k^{-r^2} \text{ , при } n \to \infty ,$$

и ранги почти всех правильных ЭК из $\mathfrak{A}(L_K)$ принадлежат интервалу φ_3 .

В Теоремах 2 и 3 получены оценки типичного числа и ранга правильных ЭК соответственно в случаях, когда число прецедентов превосходит число признаков, и, когда число признаков существенно больше числа прецедентов. В экспериментах ранг искомых правильных ЭК в алгоритме REC+ выбирается с использованием приведенных в Теоремах 2 и 3 оценок типичного ранга правильного ЭК.

Таким образом, **научная новизна** работы заключается в следующем.

1. Предложен новый последовательно-совместный подход к перечислению множеств X_{max} максимальных частых и Y_{min} минимальных нечастых элементов декартова произведения частичных порядков.

2. На базе последовательно-совместного подхода к перечислению X_{max} и Y_{min} построен алгоритм асимптотически оптимальной расшифровки двузначной монотонной функции, заданной на декартовом произведении частично упорядоченных множеств.

3. На основе поиска частых элементов в данных разработаны новые быстрые алгоритмы логической классификации, базирующиеся на перечислении корректных ЭК специального вида, названных правильными представительными. Для случая декартова произведения антицепей построен алгоритм REC перечисления правильных представительных ЭК. Для случая декартова произведения цепей разработана его модификация REC+.

4. Развита техника получения оценок метрических (количественных) характеристик логических классификаторов. А именно, получены асимптотические оценки типичного числа и типичного ранга правильных ЭК для случаев антицепей (Теорема 1, раздел 3.2.1) и цепей (Теоремы 2 и 3, раздел 3.2.2). Тем самым дано теоретическое обоснование эффективности предложенных в работе классификаторов.

Теоретическая и практическая значимость работы. Предложенные в диссертации методы работают с данными, представленными в виде декартова произведения конечных частичных порядков, тогда как классические алгоритмы ориентированы в основном на случай декартова произведения антицепей. Таким образом, класс задач, к которым применимы предложенные алгоритмы, шире по сравнению с ранее известными методами.

Предложенные в работе алгоритмы программно реализованы и проверены на синтетических и реальных данных. Алгоритм последовательно-совместного перечисления X_{max} и Y_{min} выигрывает у ранее известных последовательного и совместного методов в случае, когда мощности множеств частых и нечастых элементов не сильно отличаются. Алгоритм асимптотически оптимальной расшифровки двузначной монотонной функции работает в среднем быстрее ориентированного на худший случай алгоритма из [1] и требует меньшего числа обращений к оракулу. Классификаторы REC и REC+ требуют меньших вычислительных затрат, чем классические модели направления CVP, не уступая, а в ряде случаев превосходя их в качестве распознавания.

Полученные асимптотические оценки типичного числа и ранга правильных представительных ЭК (Теоремы 1–3) показывают, что число правильных представительных ЭК существенно меньше, чем число тупиковых ЭК (Теорема 4), поиск которых ведётся в направлении CVP. Тем самым эффективность REC и REC+ подтверждена не только экспериментально, но и теоретически. Кроме того, полученные оценки позволяют обоснованно выбирать параметры алгоритмов (ранги искомых ЭК) на практике, не прибегая к перебору.

Результаты диссертационной работы включены в отчеты по проекту РФФИ 19-01-00430 «Логический анализ частично упорядоченных целочисленных данных в задачах классификации и поиска ассоциативных правил» и проекту РНФ 24-21-00301 «Разработка новых методов корректной классификации и анализа данных на основе логического подхода».

Положения, выносимые на защиту.

1. Последовательно-совместный подход к перечислению множеств максимальных частых и минимальных нечастых элементов декартова произведения частичных порядков. Доказательство корректности предлагаемого метода (Утверждения 1–3, раздел 1.2). Экспериментальное исследование подхода, подтверждающее его эффективность.

2. Алгоритм асимптотически оптимальной расшифровки двузначной монотонной функции, заданной на декартовом произведении частично упорядоченных множеств. Доказательство корректности предлагаемого метода (Утверждения 4–6, раздел 2.2). Экспериментальное исследование алгоритма, подтверждающее его эффективность.

3. Быстрые логические классификаторы REC (для случая декартова произведения антицепей) и REC+ (для случая декартова произведения цепей), основанные на поиске правильных представительных элементарных классификаторов. Экспериментальное исследование качества и скорости работы классификаторов.

4. Асимптотические оценки типичного числа и типичного ранга правильных ЭК для случаев антицепей (Теорема 1, раздел 3.2.1) и цепей (Теоремы 2 и 3, раздел 3.2.2), позволяющие теоретически обоснованно выбирать параметры алгоритмов REC и REC+.

Степень достоверности полученных результатов обеспечивается доказательством сформулированных утверждений и теорем, а также результатами экспериментов, проведенных автором.

Апробация работы. Основные результаты работы доложены на конференциях «Математические методы распознавания образов (ММРО-19)»

(Москва, 2019), «Информационные технологии и нанотехнологии (ИТНТ-2021)» (Самара, 2021), «Математические методы распознавания образов (ММРО-20)» (Москва, 2021), «Информационные технологии и нанотехнологии (ИТНТ-2022)» (Самара, 2022), «Интеллектуализация обработки информации (ИОИ-14)» (Москва, 2022), «Информационные технологии и нанотехнологии (ИТНТ-2025)» (Самарканд, 2025).

Публикации. По тематике работы опубликовано 11 научных работ [9 – 18, 60], при этом статьи [14, 18] имеют англоязычные версии [59, 61]. В журналах, рекомендованных ВАК, опубликованы 5 статей [9, 13, 14, 16, 18]. В журналах, индексируемых в Web of Science Core Collection, опубликованы 2 статьи [59, 61]. В изданиях, индексируемых в Scopus, опубликованы 5 работ [13, 16, 18, 59, 60].

Работа состоит из введения, трех глав, заключения и списка литературы.

В первой главе предложен и исследован последовательно-совместный подход к перечислению максимальных частых и минимальных нечастых элементов декартова произведения частичных порядков, основанный на применении асимптотически оптимального алгоритма дуализации цепей RUNC-M+. Доказана корректность предложенного подхода. Экспериментально показано, что последовательно-совместный метод наиболее эффективен в случае, когда мощность множества частых элементов мало отличается от мощности множества нечастых элементов.

Во второй главе показана связь между задачей совместного перечисления максимальных частых и минимальных нечастых элементов декартова произведения частичных порядков и задачей оптимальной расшифровки двузначной монотонной функции. На базе последовательно-совместного подхода к перечислению X_{max} и Y_{min} построен новый алгоритм расшифровки, нацеленный на «типичный случай». Экспериментально показано, что при задании функции на E_k^n в случае больших значений k и n, предложенный алгоритм работает существенно быстрее классического алгоритма, ориентированного на «худший случай», и требует меньшего числа обращений к оператору B.

В третьей главе предложены и реализованы алгоритмы логической классификации, основанные на поиске частых элементов в частично упорядоченных данных. Получены асимптотические оценки типичного числа и типичного ранга правильных представительных элементарных классификаторов, которые позволяют теоретически обоснованно выбирать параметры алгоритмов. Экспериментально показано, что предложенные алгоритмы работают быстрее традиционных логических классификаторов и не уступают им в качестве распознавания.

В заключении формулируются основные результаты диссертационной работы, выносимые на защиту, и задаются направления дальнейших исследований.

Глава 1. Поиск частых (нечастых) элементов частично упорядоченных данных

Задачи поиска частых и нечастых элементов в данных являются одними из центральных задач логического анализа информации и имеют широкий спектр приложений, в частности, в анализе потребительского поведения, биоинформатике, задачах классификации по прецедентам и в задачах дискретной математики.

Вопросы поиска частых и нечастых элементов данных впервые сформулированы в контексте анализа потребительской корзины при поиске ассоциативных правил [47, 49]. Наиболее подробно указанные задачи исследовались для случая бинарных данных, где были предложены такие классические алгоритмы, как Apriori [48], DepthProject [50] и FP-Growth [67]. Однако в ряде прикладных областей, в частности в задачах из области дискретной математики и машинного обучения, возникает необходимость работы с данными, представленными в виде декартова произведения конечных частично упорядоченных множеств.

В настоящей главе приводится обзор основных методов поиска частых и нечастых элементов в данных. Подробно рассматриваются алгоритмы Apriori, DepthProject и FP-Growth, обсуждаются их достоинства и недостатки. Предлагается новый последовательно-совместный подход к перечислению максимальных частых и минимальных нечастых элементов декартова произведения частичных порядков. Показывается связь рассматриваемой задачи с задачей дуализации над произведением частичных порядков. Приводится теоретическое обоснование корректности предложенного метода и результаты экспериментального исследования его эффективности для случая декартова произведения конечных цепей.

1.1. Задачи поиска частых и нечастых элементов частично упорядоченных данных.

Рассматривается множество P представленное в виде декартова произведения n конечных непустых частично упорядоченных множеств $P_1, \dots, P_n$. На множестве P вводится частичный порядок таким образом, что элемент $y = (y_1, \dots, y_n)$ множества P следует за элементом $x = (x_1, \dots, x_n)$ этого множества тогда и только тогда, когда y_i следует за x_i в P_i, $i = \overline{1,n}$. Пусть дана база данных $D(P)$, представляющая собой совокупность некоторых не обязательно различных элементов из P. Элементы базы данных называются транзакциями. *Частотой* элемента $x \in P$ называется величина $\nu(x)$, равная числу транзакций в $D(P)$, следующих за x. Задан некоторый порог $s \in \mathbb{N}$, где $\mathbb{N}$ – множество натуральных чисел. Если $\nu(x) \geq s$, то элемент x называется s *-частым*. Иначе x называется s*-нечастым* элементом. Требуется по заданным $D(P)$ и s перечислить частые (нечастые) элементы.

Поставленная задача является вычислительно сложной. Число всех частых (нечастых) элементов множества $P = P_1 \times \dots \times P_n$ растет экспоненциально с ростом n. Поэтому на практике вместо перечисления всех частых (нечастых) элементов, как правило, строят более компактное множество максимальных частых (минимальных нечастых) элементов.

Если $x \in P$ является s -частым элементом и в P не существует другого s-частого элемента, следующего за x, то x называется *максимальным* s *-частым* элементом. Если $x \in P$ является s -нечастым элементом и в P не существует другого s -нечастого элемента, предшествующего x, то x называется *минимальным* s *-нечастым* элементом. Обозначим через X_{max} и Y_{min} соответственно множества всех максимальных частых и минимальных нечастых элементов.

Поиск только максимальных частых и минимальных нечастых элементов позволяет компактно представлять информацию о всех частых и нечастых элементах соответственно. Действительно, из определения частичного порядка следует, что любой частый элемент предшествует хотя бы одному максимальному

частому элементу, а любой нечастый элемент следует за хотя бы одним минимальным нечастым элементом. Таким образом, множества максимальных частых и минимальных нечастых элементов полностью характеризуют разбиение P на частые и нечастые элементы: элемент x является частым тогда и только тогда, когда он предшествует хотя бы одному элементу из множества максимальных частых элементов.

Ключевым свойством, лежащим в основе всех алгоритмов поиска частых элементов, является свойство *антимонотонности* частоты: если элемент x является частым, то все предшествующие ему элементы также являются частыми. Если элемент x является нечастым, то все следующие за ним элементы также являются нечастыми. Это свойство означает, что множества частых и нечастых элементов разделены границей, которая фактически определяется множествами X_{max} и Y_{min} .

Множества X_{max} и Y_{min} связаны свойством двойственности. Это означает, что каждое из множеств X_{max} или Y_{min} однозначно определяет другое. Задача построения одного из этих множеств при наличии другого называется задачей дуализации над произведением частичных порядков – это труднорешаемая перечислительная задача дискретной математики.

Свойство двойственности открывает два способа построения множеств X_{max} и Y_{min} . Первый способ – последовательный: сначала одно из множеств строится непосредственно по базе данных $D(P)$, затем второе множество находится путём дуализации первого. Второй способ – совместное перечисление обоих множеств, при котором на каждом шаге одновременно пополняются как множество максимальных частых, так и множество минимальных нечастых элементов. В настоящей главе предлагается новый последовательно-совместный подход, синтезирующий идеи обоих методов. Прежде чем перейти к его описанию, требуется рассмотреть основные алгоритмы поиска частых (нечастых) элементов в данных.

Традиционные алгоритмы поиска частых элементов в данных разработаны для случая бинарных данных, когда каждое частично упорядоченное множество содержит ровно два элемента (0 и 1) с порядком $0 \preccurlyeq 1$. В этом случае каждый элемент множества P может быть представлен в виде бинарного вектора. Длиной такого вектора называется число его ненулевых компонент (координат).

Алгоритмы поиска частых элементов в данных работают итеративно по общей схеме. На каждом шаге генерируется некоторое множество кандидатов. Среди элементов этого множества есть как частые, так и нечастые элементы. Для каждого кандидата определяется его частота и далее рассматриваются только те элементы, которые являются частыми [49]. Алгоритмы различаются способом генерации кандидатов, порядком обхода множества частых элементов и способом подсчёта частот.

Работу такого алгоритма удобно описывать в терминах обхода дерева решений. Дерево решений определяется следующим образом. Вершины дерева соответствуют частым элементам множества P ; корнем дерева является минимальный (пустой) элемент. Вершина, соответствующая элементу y, является потомком вершины, соответствующей элементу x , если x предшествует y и не существует такого частого элемента z , что x предшествует z и z предшествует y . Максимальные частые элементы соответствуют листовым вершинам дерева; однако не все листовые вершины соответствуют максимальным частым элементам. Обход дерева решений может осуществляться в ширину или в глубину. Выбор порядка обхода существенно влияет на эффективность алгоритма. Упомянутое выше свойство антимонотонности частоты позволяет существенно сократить обход дерева решений: если на некотором шаге обнаружен нечастый элемент, то все следующие за ним элементы также являются нечастыми, и соответствующие ветви дерева могут быть отсечены.

Наиболее известными и широко используемыми алгоритмами являются Apriori [48], DepthProject [50] и FP-Growth [67]. Алгоритм Apriori обходит дерево решений в ширину. Алгоритм DepthProject обходит дерево решений в глубину и

использует так называемые проекции базы данных для ускорения времени счета. Алгоритм FP-Growth строит специальное сжатое представление базы данных, FP-дерево, и осуществляет рекурсивный поиск частых элементов по FP-дереву без обращения к исходной базе данных.

Алгоритм Apriori, предложенный в 1994 году [48], является исторически первым алгоритмом поиска частых элементов и наиболее простым в описании. Алгоритм реализует обход дерева решений в ширину и использует свойство антимонотонности для отсечения заведомо нечастых кандидатов.

Apriori работает итеративно, на каждом шаге k находятся все частые элементы длины k (содержащие ровно k координат, отличных от нуля в случае бинарных данных). Процесс состоит из двух основных фаз на каждом шаге: генерация кандидатов и подсчет их частот.

Шаг 1. На первом шаге ($k = 1$) множество кандидатов C_1 состоит из всех элементов бинарного множества P длины 1. Для каждого кандидата $x \in C_1$ вычисляется его частота $\nu(x)$ путем просмотра всей базы данных $D(P)$ и формируется множество L_1 элементов длины 1 с частотой встречаемости большей или равной s .

Шаг k , $k > 1$. Пусть на предыдущем шаге сформировано множество L_{k-1}. Тогда множество кандидатов C_k строится следующим образом. Пусть $x = (x_1, \dots, x_n) \in L_{k-1}$ и пусть j - позиция самой правой ненулевой компоненты элемента x . Для каждого элемента $y = (y_1, \dots, y_n) \in L_1$ такого, что $y_i \neq 0$ для некоторого $i > j$, формируется кандидат $z = (z_1, \dots, z_n)$, где $z_l = \max(x_l, y_l)$ для $l = \overline{1, n}$. Полученный кандидат z добавляется в множество C_k . Для каждого кандидата $z \in C_k$ вычисляется его частота $\nu(z)$ путем просмотра всей базы данных $D(P)$ и формируется множество L_k элементов длины k с частотой встречаемости не менее s . Если множество $L_k = \emptyset$ или $k = n$, то алгоритм завершает свою работу и возвращает в качестве ответа множество $L = \bigcup_{i=1}^{k} L_i$.

Работа алгоритма Apriori соответствует обходу дерева решений в ширину: на каждом шаге k рассматриваются все частые элементы длины k и только они, что соответствует обходу всех вершин k-го уровня дерева.

Основным недостатком алгоритма Apriori является необходимость полного просмотра базы данных на каждом уровне дерева решений. Если максимальная длина частого элемента равна k, то алгоритм выполняет не менее k полных проходов по базе данных. При большом числе транзакций и низком пороге частоты это приводит к значительным затратам времени. Кроме того, число генерируемых кандидатов растёт экспоненциально, что создаёт существенную нагрузку на память. Ещё одним ограничением является то, что на каждом уровне k алгоритм строит все частые элементы длины k, не имея возможности перейти к элементам большей длины до завершения обхода текущего уровня. Это делает Apriori неэффективным для задачи поиска максимальных частых элементов, особенно при большой длине максимальных частых элементов.

В отличие от алгоритма Apriori, алгоритм DepthProject обходит дерево решений в глубину, что позволяет находить максимальные частые элементы более эффективно. Сначала DepthProject строит множество L_1 всех частых элементов длины 1. Затем для каждого частого элемента $x \in L_1$ рекурсивно строится поддерево, корнем которого является x. Пусть на очередном шаге рассматривается частый элемент $x = (x_1, \dots, x_n)$. При генерации кандидатов рассматриваются все элементы $y = (y_1, \dots, y_n)$ такие, что позиция ненулевой компоненты элемента y больше позиции самой правой ненулевой компоненты элемента x. В результате формируется кандидат $z = (z_1, \dots, z_n)$, где $z_l = \max(x_l, y_l)$ для $l = \overline{1,n}$.

При переходе к дочерней вершине алгоритм формирует *проекцию* базы данных – такое подмножество транзакций исходной базы, которые следуют за текущим элементом. Причем рассматриваются только такие координаты векторов, которые в текущем элементе равны нулю. С каждым шагом вглубь дерева решений проекция базы данных сокращается как по числу транзакций, так и по числу координат, что позволяет повторно использовать результаты подсчёта частот и

избежать многократного обращения к полной базе данных [50, 51]. Обход в глубину также позволяет эффективно отсекать лишние ветви. Перед обработкой дочерних вершин осуществляется проверка, является ли частым элемент, полученный из текущего установкой в единицу всех таких нулевых координат, каждая из которых находится правее самого правого единичного элемента рассматриваемого вектора. Если такой элемент уже содержится среди ранее найденных частых элементов, то все вершины текущего поддерева заведомо соответствуют частым элементам и поддерево может далее не рассматриваться. Поскольку при обходе в глубину длинные частые элементы обнаруживаются рано, множество найденных элементов быстро пополняется, что делает процедуру отсечения эффективной [50, 51].

При таком обходе дерева решений справедливо следующее утверждение. Если на очередном шаге построен частый элемент, соответствующий висячей вершине, то либо этот элемент предшествует ранее найденному максимальному частому элементу, либо сам является максимальным частым. Таким образом, при построении новой висячей вершины алгоритм DepthProject проверяет предшествует ли соответствующий ей элемент ранее найденному максимальному частому элементу или нет.

Основным недостатком алгоритма DepthProject является необходимость проверки максимальности найденного частого элемента путём сравнения с множеством ранее найденных максимальных частых элементов. При большом числе максимальных частых элементов эта проверка требует значительных вычислительных ресурсов.

В настоящей работе исследована модификация алгоритма DepthProject, названная алгоритмом AD, которая проверяет максимальность частого элемента непосредственно по базе данных, не обращаясь к множеству ранее найденных максимальных частых элементов. Такой подход позволяет существенно ускорить поиск максимальных частых элементов.

Процедура проверки максимальности в алгоритме AD основана на следующем свойстве: частый элемент x является максимальным тогда и только тогда, когда для каждой позиции i , где x_i не является наибольшем в P_i (в случае

бинарных данных это значит, что $x_i = 0$) любой элемент x' , полученный увеличением x_i , является нечастым. Фактически, для проверки частого элемента x на максимальность достаточно вычислить частоту встречаемости $O(n)$ элементов, следующих за x , что в случае больших данных существенно быстрее сравнения элемента x с экспоненциально большим числом ранее найденных максимальных элементов.

Проведено экспериментальное сравнение скорости счета алгоритмов поиска максимальных частых элементов в бинарных данных DepthProject и AD. Алгоритмы реализованы на языке C++. Эксперименты проведены на случайных бинарных базах данных из равномерного распределения с числом транзакций m . Минимальная частота бралась равной $s = 0.1 * m$. Результаты счета усреднены по двадцати независимым запускам. В таблице 1 приведено время работы алгоритмов в миллисекундах, DepthProject и AD соответственно.

Таблица 1 – Время счета алгоритмов DepthProject и AD, мс

	$n = 10$	$n = 20$	$n = 30$	$n = 35$
$m = 10$	0.254 / 0.149	25.52 / 10.59	2282 / 736.2	10130 / 3498
$m = 100$	1.292 / 0.124	145.5 / 1.214	2972 / 5.886	9439 / 10.93
$m = 1000$	2.405 / 0.125	228.4 / 1.197	3010 / 4.898	7894 / 8.875

Как видно из приведенных в таблице 1 данных счета, предложенная модификация алгоритма DepthProject, названная алгоритмом AD, работает существенно быстрее и время ее работы растет медленнее с ростом n .

Алгоритм FP-Growth также реализует стратегию обхода дерева решений в глубину, но использует принципиально иное представление исходных данных. Вместо многократного сканирования базы данных $D(P)$, как это делают алгоритмы Apriori и DepthProject, алгоритм FP-Growth строит компактное представление базы данных в виде специальной древовидной структуры, называемой FP-деревом

(Frequent Pattern tree). FP-дерево позволяет эффективно подсчитывать частоты элементов без повторного обращения к исходной базе данных.

Процесс построения FP-дерева состоит из двух этапов. Сначала находится множество L_1 всех частых элементов длины 1, элементы которого упорядочиваются по убыванию частоты. Элементы с частотой меньше заданного порога s исключаются из рассмотрения. Затем для каждой транзакции из $D(P)$ создается упорядоченный список частых элементов (в порядке убывания частоты), который «добавляется» в FP-дерево.

FP-дерево является ориентированным деревом. Корневая вершина соответствует минимальному элементу из P . Каждая транзакция $t = (t_1, \dots, t_n) \in D(P)$ представляется в виде пути от корня к некоторой вершине дерева. Каждая некорневая вершина содержит пару - частый элемент длины 1 из множества L_1 и счетчик, показывающий количество транзакций из базы данных, которые содержат путь от корня до данной вершины.

FP-дерево строится по правилам 1–4:

1. Для транзакции $t = (t_1, \dots, t_n)$ определяются все позиции i, где $t_i \neq 0$. Соответствующие этим позициям элементы из L_1 упорядочиваются по убыванию частоты, формируя в результате список $T = [t_{i_1}, \dots, t_{i_k}]$.
2. Начиная с корневой вершины, алгоритм ищет путь в дереве, соответствующий списку T . Последовательно для каждого элемента t_{i_j} проверяется, существует ли среди потомков текущей вершины вершина с меткой t_{i_j} , $j = \overline{1,k}$.
3. Если такой потомок существует, алгоритм переходит к этой вершине и ее счетчик увеличивается на 1.
4. Если потомка с меткой t_{i_j} не существует, то создается новая вершина с меткой t_{i_j} и счетчиком равным 1. Эта вершина становится потомком текущей вершины. Затем аналогичным образом обрабатывается следующий элемент $t_{i_{j+1}}$.

Для эффективного поиска частых элементов по дереву поддерживается так называемая таблица заголовков (header table), которая содержит ссылки на все

вершины дерева, соответствующие элементам из L_1 . После построения FP-дерева поиск частых элементов осуществляется рекурсивно. Основная идея заключается в том, что для поиска всех частых элементов, следующих за элементом $x \in L_1$, достаточно рассмотреть только ту совокупность транзакций базы данных $D(P)$, которые содержат x , и построить для этой совокупности отдельное «условное» FP-дерево.

Алгоритм FP-Growth обрабатывает частые элементы из множества L_1 в порядке возрастания частоты (от наименее частого к наиболее частому). Для каждого $x \in L_1$ выполняется следующая процедура.

1. На первом шаге условная база данных $D_x(P) = \emptyset$.
2. С помощью таблицы заголовков находятся все вершины в FP-дереве, помеченные элементом x . Для каждой такой вершины v находится путь от корня дерева до родителя этой вершины.
3. Каждый найденный путь соответствует некоторой транзакции, которая добавляется в условную базу данных $D_x(P)$ столько раз, какова величина счетчика вершины v .
4. Для условной базы данных $D_x(P)$ строится новое FP-дерево по тому же алгоритму, что и для исходных данных.
5. К условному FP-дереву рекурсивно применяются пункты 1–4. Все найденные частые элементы объединяются с x, образуя таким образом частые элементы большей длины.

Преимущество FP-Growth заключается в том, что каждый элемент обрабатывается только один раз, и не требуется генерации кандидатов, как в алгоритмах Apriori и DepthProject. Однако алгоритм может потребовать значительного объема памяти для хранения FP-дерева при работе с плотными базами данных.

Таким образом, среди рассмотренных подходов алгоритм Apriori наиболее прост в реализации, но требует многократного просмотра базы данных и не всегда эффективен для поиска максимальных частых элементов. Алгоритм DepthProject,

обходящий дерево решений в глубину, нацелен на поиск максимальных частых элементов благодаря раннему анализу листовых вершин и процедуре отсечения ветвей. Алгоритм FP-Growth использует специальную структуру данных для сжатия исходной базы данных и требует всего двух проходов по базе данных, однако может быть неэффективным в случае большого числа частых элементов в данных.

Описанные выше алгоритмы могут быть модифицированы для поиска нечастых элементов в данных. Модификация основана на принципе антимонотонности нечастых элементов: если элемент $x \in P$ является нечастым, то все элементы, следующие за x , также являются нечастыми.

При поиске всех нечастых элементов алгоритмы обходят инвертированное дерево решений, корнем которого является наибольший элемент из P. Ребра в инвертированном дереве направлены от элементов к их предшественникам, то есть вершина, соответствующая элементу x , является потомком вершины, соответствующей элементу y , если x предшествует y и не существует элемента z такого, что x предшествует z и z предшествует y .

Алгоритмы также могут быть модифицированы для построения искомых элементов в случае декартова произведения антицепей и декартова произведения цепей путем бинаризации исходных данных. Пусть $x = (x_1, \dots, x_n) \in P$. В случае если P представляет собой декартово произведение антицепей с элементами из $\{0, 1, \dots, k-1\}$, то к вектору $x = (x_1, \dots, x_n)$ применяется хорошо известная процедура one-hot кодирования, которая преобразует его в бинарный вектор $\hat{x} = (\alpha(x_1, 0), \alpha(x_1, 1), \dots, \alpha(x_1, k-1), \dots, \alpha(x_n, k-1))$ размера $k \times n$, где $\alpha(i, j)$ равняется 1 в случае $i = j$ и равняется 0 иначе, $i \in \{0, 1, \dots, k-1\}$, $j \in \{0, 1, \dots, k-1\}$. В случае если $P -$ k -значный n -мерный куб (случай произведения цепей), то к вектору $x = (x_1, \dots, x_n)$ применяется иная процедура бинаризации, которая преобразует его в вектор $\tilde{x} = (\beta(x_1, 0), \beta(x_1, 1), \dots, \beta(x_1, k-1), \dots, \beta(x_n, k-1))$ размера $k \times n$, где $\beta(i, j)$ равняется 1 в случае $i \geq j$ и равняется

0 иначе, $i \in \{0, 1, \dots, k-1\}$, $j \in \{0, 1, \dots, k-1\}$. Затем алгоритмы применяются к векторам $\hat{x}$ и к $\tilde{x}$.

В [63] показано, что при поиске максимальных частых и минимальных нечастых элементов частичных порядков естественным образом возникает необходимость решения задачи дуализации над произведением частичных порядков. Дуализация над произведением частичных порядков является труднорешаемой перечислительной задачей дискретной математики и занимает важную роль в теории алгоритмической сложности.

Пусть P – частично упорядоченное множество, представимое в виде декартова произведения конечных частично упорядоченных множеств $P = P_1 \times \dots \times P_n$. Пусть R – непустое подмножество множества P. Через R^+ обозначается множество всех элементов из P следующих хотя бы за одним элементом из R : $R^+ = \{y \in P \mid \exists x \in R : x \leqslant y\}$. Через R^- – множество всех элементов из P предшествующих хотя бы одному элементу из R : $R^- = \{y \in P \mid \exists x \in R : y \leqslant x\}$. Множество $I(R^+)$, состоящее из всех максимальных элементов множества $P \backslash R^+$, называется *максимальным независимым* от R. Множество $I(R^-)$, состоящее из всех минимальных элементов множества $P \backslash R^-$, называется *минимальным независимым* от R. Каждая из задач построения $I(R^+)$ и $I(R^-)$ называется задачей дуализации над произведением частичных порядков.

Связь между задачей дуализации и перечислением максимальных частых и минимальных нечастых элементов основана на свойстве двойственности. Множества X_{max} и Y_{min} являются двойственными в том смысле, что $I(X_{max}^-) = Y_{min}$ и $I(Y_{min}^+) = X_{max}$. Действительно, нетрудно видеть, что X_{max}^- – это множество всех частых элементов, а Y_{min}^+ – множество всех нечастых элементов. Тогда $P \backslash X_{max}^-$ и $P \backslash Y_{min}^+$ – множества всех нечастых и всех частых элементов соответственно. Следовательно, по определению, $I(X_{max}^-) = Y_{min}$ и $I(Y_{min}^+) = X_{max}$. Указанные свойства означают, что одно из множеств X_{max} или Y_{min} однозначно определяет другое через решение задачи дуализации.

Важным частным случаем дуализации над произведением частичных порядков является монотонная дуализация, которая формулируется как задача построения сокращенной дизъюнктивной нормальной формы монотонной булевой функции по ее конъюнктивной нормальной форме. Монотонная дуализация может быть сформулирована также как задача поиска всех минимальных вершинных покрытий гиперграфа или как задача построения всех неприводимых покрытий булевой матрицы. В случае, если P – булев куб, а R – множество всех единиц двузначной монотонной функции f , заданной на булевом кубе P , то задача нахождения множества $I(R^+)$ эквивалентна задаче монотонной дуализации.

Число решений задачи дуализации растёт экспоненциально с ростом размера входных данных. В теории алгоритмической сложности дискретных задач эффективность алгоритмов для перечислительных задач принято оценивать не общей временной сложностью, а временем выполнения одного шага, то есть временем нахождения очередного решения. Алгоритм с полиномиальной временной оценкой одного шага в худшем случае (для самой сложной индивидуальной задачи) называется алгоритмом с полиномиальной задержкой [69]. Для задачи дуализации в общем случае алгоритм с полиномиальной задержкой не построен, и вопрос о его существовании остаётся открытым. Для монотонной дуализации построить такие алгоритмы удалось лишь для небольшого числа частных случаев. В связи с этим выделились два основных подхода, применимых к широкому классу дискретных перечислительных задач.

Первый подход основан на построении инкрементальных алгоритмов, сложность шага которых зависит от размера входных данных и числа уже найденных решений [54, 62, 64]. Инкрементальному алгоритму разрешено на каждом шаге просматривать множество ранее построенных решений, затрачивая на это время, полиномиальное от размера задачи и числа уже найденных решений. Сложность инкрементальных алгоритмов оценивается в худшем случае. Наиболее значимым результатом в рамках данного подхода является алгоритм монотонной дуализации Фредмана–Хачияна [64], имеющий квазиполиномиальную задержку в

худшем случае. Квазиполиномиальная сложность означает, что время нахождения очередного решения ограничено величиной $p(m,n)^{O(\log p(m,n))}$, где $p(m,n)$ – полином от размера входа задачи и числа уже найденных решений. То есть превышает полиномиальное, но существенно меньше экспоненциального. Основным недостатком инкрементального подхода является снижение эффективности при большом числе решений вследствие необходимости сравнения каждого нового кандидата с множеством найденных решений.

Второй подход предполагает построение асимптотически оптимальных алгоритмов [20]. Этим алгоритмам разрешено делать «лишние» шаги, не приводящие к новым решениям исходной задачи, при условии, что для почти всех индивидуальных задач число таких шагов имеет более низкий порядок роста по сравнению с числом всех решений задачи. Формально, если N обозначает число решений, а L – число лишних шагов, то требуется, чтобы $L = o(N)$ при $N \to \infty$ для почти всех вариантов задачи. Лишний шаг – это построение такого кандидата, который либо был найден ранее, либо не является решением; при этом проверка того, является ли шаг лишним, должна осуществляться за полиномиальное от размера входных данных время. Подход предложен Е. В. Дюковой [20], построившей в 1977 г. первый асимптотически оптимальный алгоритм монотонной дуализации. В отличие от инкрементальных алгоритмов, асимптотически оптимальные алгоритмы ориентированы на типичный случай (эффективны для почти всех вариантов задачи) и на сегодняшний день являются лидерами по скорости счёта.

Для случая декартова произведения конечных цепей разработан асимптотически оптимальный алгоритм дуализации RUNC-M+. Алгоритм является модификацией алгоритма RUNC-M, первоначально предложенного для решения задачи монотонной дуализации. Временная сложность шага алгоритма RUNC-M+ составляет $O(mnq)$, где m – это мощность исследуемого множества R , $q = \min(m,n)$. В настоящее время алгоритмы RUNC-M и RUNC-M+ являются лидерами по скорости счета и используются в диссертационной работе для

реализации последовательно-совместного подхода к перечислению максимальных частых и минимальных нечастых элементов частичных порядков.

1.2. Последовательно-совместный подход к перечислению максимальных частых и минимальных нечастых элементов частично упорядоченных данных

Идея совместного инкрементального перечисления максимальных частых и минимальных нечастых элементов частичных порядков предложена в [63]. Идея совместного перечисления, описанная в [63], имеет итеративную природу. На первом шаге рассматривается некоторый случайный элемент $x \in P$. Если x – частый элемент, то ищется максимальный частый элемент, следующий за x , который пополняет множество $X \subseteq X_{max}$. Если x – нечастый элемент, то ищется минимальный нечастый элемент, предшествующий x , который пополняет множество $Y \subseteq Y_{min}$. Если на шаге i $(i \geq 1)$ $X \neq \emptyset$, $Y = \emptyset$, то ищется элемент x такой, что $x \not\preccurlyeq z$, $\forall z \in P$. Если $X = \emptyset$, $Y \neq \emptyset$, то ищется элемент x такой, что $x \not\succcurlyeq z$, $\forall z \in P$. Если же $X \neq \emptyset$, $Y \neq \emptyset$, то ищется элемент x такой, что $x \not\preccurlyeq z$, $\forall z \in P$ и $x \not\succcurlyeq z$, $\forall z \in P$. Затем, аналогично первому шагу, находится максимальный частый или минимальный нечастый элемент. Однако в [63] идея совместного перечисления искомых множеств экспериментально не исследована и не предложены конкретные указания по возможной ее реализации.

Алгоритм, основанный на идее совместного перечисления X_{max} и Y_{min} , реализован и исследован в настоящей работе для случая декартова произведения цепей. Для доказательства корректности алгоритмов совместного и последовательно-совместного перечисления необходимо доказать приведенные ниже утверждения 1–3.

Утверждение 1. *Если* $X \subset X_{max}$ *, то* $I(X^-)$ *содержит хотя бы один частый элемент. Аналогично, если* $Y \subset Y_{min}$ *, то* $I(Y^+)$ *содержит хотя бы один нечастый элемент.*

Доказательство. Пусть $X \subset X_{max}$, $x \in X_{max} \setminus X$. Из того, что x не сравним ни с одним другим элементом множества X_{max}, следует, что $x \in P \setminus X^-$. Таким образом, в $I(X^-)$ существует элемент q такой, что $q \preccurlyeq x$ и q – частый. Утверждение для множества $I(Y^+)$ доказывается аналогично.

Утверждение 2. *Если $X \subset X_{max}$ и $y \in I(X^-)$ является нечастым элементом, то y – минимальный нечастый элемент. Аналогично, если $Y \subset Y_{min}$ и $x \in I(Y^+)$ является частым элементом, то x – максимальный частый элемент.*

Доказательство. Пусть $X \subset X_{max}$ и $y \in I(X^-)$ – нечастый элемент. Предположим, что y не является минимальным нечастым элементом, то есть $y \notin Y_{min} = I(X^-_{max})$. Тогда, так как y – нечастый элемент, то в $P \setminus X^-_{max}$ найдется минимальный нечастый элемент z такой, что $z \neq y$, $z \preccurlyeq y$. Из $(P \setminus X^-_{max}) \subseteq (P \setminus X^-)$ следует, что $z \in P \setminus X^-$, что противоречит условию $y \in I(X^-)$. Полученное противоречие доказывает первую часть утверждения. Вторая часть утверждения доказывается аналогично.

Утверждение 3. *Пусть $X \subseteq X_{max}$, $Y \subseteq Y_{min}$. Тогда $I(X^-) = Y$ (или $I(Y^+) = X$) тогда и только тогда, когда $X = X_{max}$, $Y = Y_{min}$.*

Доказательство. Пусть $X \subset X_{max}$. Из утверждения 1 следует, что $I(X^-)$ содержит хотя бы один частый элемент. Однако множество Y не содержит частых элементов, следовательно $I(X^-) \neq Y_{min}$. Если же $X = X_{max}$, то $I(X^-) = Y_{min}$. Таким образом, $I(X^-) = Y$ тогда и только тогда, когда $X = X_{max}$, $Y = Y_{min}$.

Алгоритм, основанный на идее совместного перечисления X_{max} и Y_{min}, строит две последовательности вложенных множеств: $X_1 \subset X_2 \subset \cdots \subset X_{max}$ и $Y_1 \subset Y_2 \subset \cdots \subset Y_{min}$. На первом шаге $X_1 = \{x\}$, $Y_1 = \{y\}$, где x и y ищутся по схеме алгоритма Apriori. Если на шаге $i + 1$ $(i \geq 1)$ мощность X_i больше мощности Y_i, то строится множество $I(Y_i^+)$, иначе строится множество $I(X_i^-)$. Опишем шаг алгоритма при условии, что на шаге $i + 1$ построено множество $I(X_i^-)$. Описание алгоритма для случая построения множества $I(Y_i^+)$ будет аналогичным. Согласно утверждениям 1 и 2, множество $I(X_i^-)$ либо не содержит частых элементов и совпадает с множеством Y_{min} (в этом случае $X_i = X_{max}$ и алгоритм заканчивает

работу), либо $I(X_i^-)$ содержит как частые, так и нечастые элементы. Каждый нечастый элемент из $I(X_i^-)$ является минимальным нечастым и пополняет множество Y_i , формируя в результате множество Y_{i+1} . Для каждого частого элемента находится один содержащий его максимальный частый элемент обходом дерева решений в глубину, который пополняет множество X_i , формируя в результате множество X_{i+1}.

Очевидно, что время работы совместного алгоритма в основном зависит от числа минимальных нечастых и максимальных частых наборов. На каждой новой итерации происходит дуализация все больших по мощности множеств, X_i или Y_i . Если число итераций становится достаточно большим, то скорость работы совместного перечисления существенно снижается, что делает его практически неприменимым для задач большой размерности.

Достаточно очевиден поиск X_{max} и Y_{min} при заданной $D(P)$ путем последовательного построения множеств X_{max} и Y_{min} . Данный поиск осуществляется в два этапа. На первом этапе находятся все максимальные частые элементы X_{max} или все нечастые элементы Y_{min} с помощью алгоритма поиска соответствующих элементов (могут применяться алгоритмы Apriori, DepthProject, FPGrowth и другие). На втором этапе используется свойство двойственности $I(X_{max}^-) = Y_{min}$ и $I(Y_{min}^+) = X_{max}$. Если на первом этапе найдено множество X_{max} , то множество Y_{min} находится путем дуализации X_{max} . Если на первом этапе найдено множество Y_{min} , то множество X_{max} находится путем дуализации Y_{min}. Очевидно, что данный подход будет проявлять себя наилучшим образом в случаях, когда алгоритм, примененный на первом этапе, может найти одно из искомых множеств существенно быстрее, чем другое множество. В частности, это происходит, когда мощность X_{max} существенно отличается от мощности Y_{min} .

Предлагается новый итеративный алгоритм, который синтезирует идеи последовательного и совместного методов, описанных выше. Положим $X_0 = \emptyset$. Строится последовательность $X_1 \subset X_2 \subset \cdots \subset X_{max}$. На первом шаге $X_1 = \{x\}$, где x ищется простым обходом дерева решений в глубину. На шаге $i + 1$ $(i \geq 1)$

решается задача дуализации множества $X_i \backslash X_{i-1}$. Пусть множество D есть результат дуализации $X_i \backslash X_{i-1}$. Согласно утверждению 1, множество D содержит частые элементы. Для каждого частого элемента из D находится один содержащий его максимальный частый элемент путем обхода дерева решений в глубину. Все найденные максимальные частые элементы, которых нет в множестве X_i , добавляются к X_i , и таким образом формируется X_{i+1}. Если же все найденные частые наборы уже содержатся в X_i , то решается задача дуализации множества X_i . Если в $I(X_i^-)$ нет частых элементов, то $I(X_i^-) = Y_{min}$, $X_i = X_{max}$ и алгоритм завершает работу. Иначе для каждого частого элемента из $I(X_i^-)$ находится один содержащий его максимальный частый элемент, который пополняет множество X_i , формируя в результате множество X_{i+1}.

1.3. Экспериментальное исследование

В ходе экспериментального исследования рассматривался случай данных, представленных в виде произведения цепей мощности 5. Для таких данных проводился поиск максимальных частых и минимальных нечастых наборов следующими методами: алгоритмом Apriori, модифицированным для случая цепей; последовательным методом; совместным методом; последовательно-совместным методом.

Все методы реализованы на языке Python 3. Задача дуализации решалась алгоритмом дуализации цепей RUNC-M+ [29, 30]. Эксперименты проведены на случайных базах данных различной размерности. Можно выделить два следующих случая соотношения мощностей множеств всех частых и нечастых наборов.

Случай 1: мощность множества частых элементов примерно равна мощности множества нечастых элементов.

Случай 2: мощность множества частых элементов существенно меньше (больше) мощности множества нечастых элементов.

Описанные случаи схематично изображены на рисунке 1. Графики зависимости времени работы тестируемых методов от мощности множеств X_{max} и Y_{min} приведены на рисунке 2.

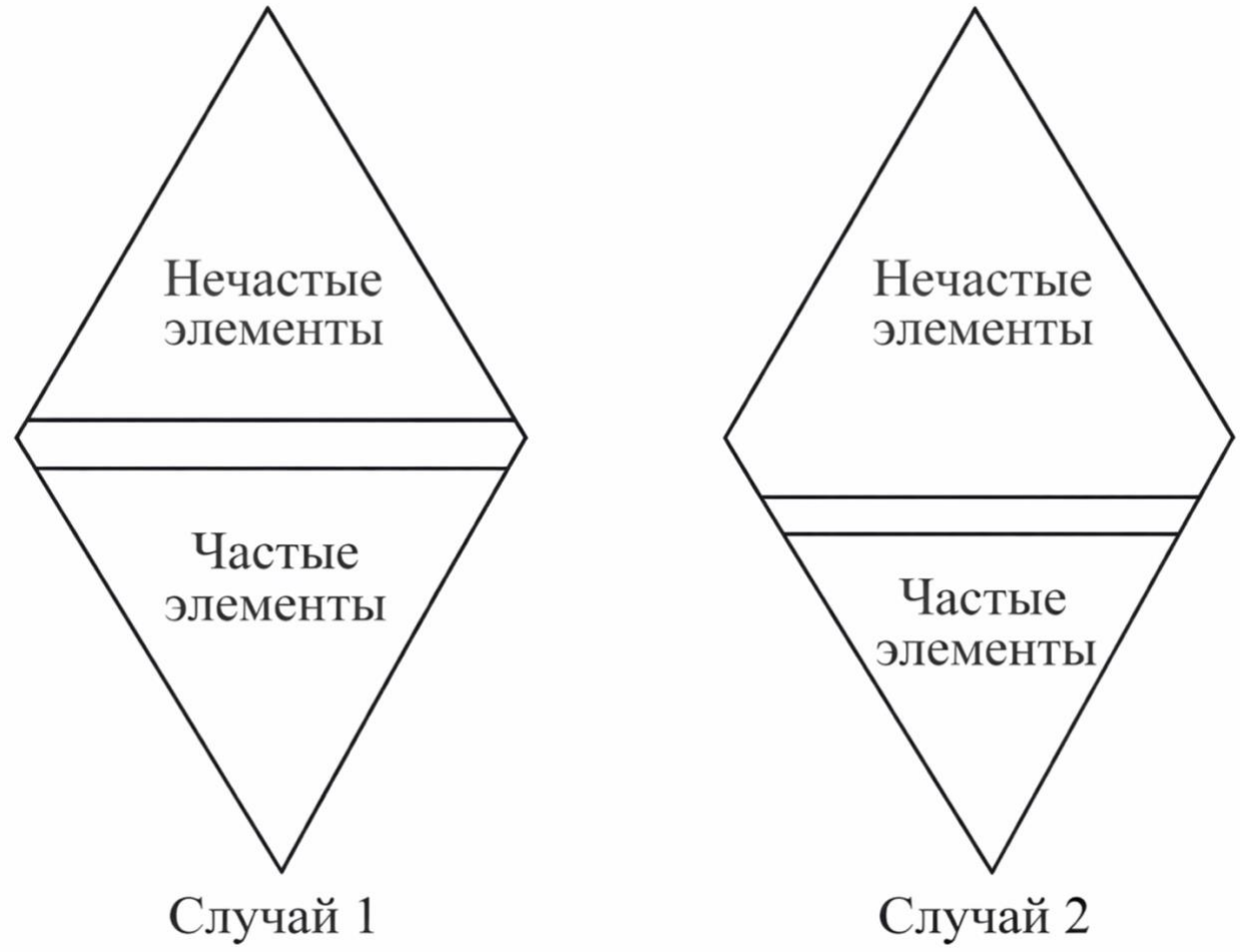

Рисунок 1 – Два случая соотношения мощностей множеств частых и нечастых элементов

Нетрудно видеть, что в случае 1 лучше работает последовательно-совместный алгоритм: множества частых и нечастых элементов имеют примерно одинаковую мощность. В случае 2 быстрее работает последовательный алгоритм: быстрее найти множество максимальных частых элементов, обработав небольшое множество частых элементов, и дуализировать результат. Время поиска множеств X_{max} и Y_{min} совместным методом и модифицированным алгоритмом Apriori растет

существенно быстрее времени поиска последовательно-совместным методом в обоих случаях.

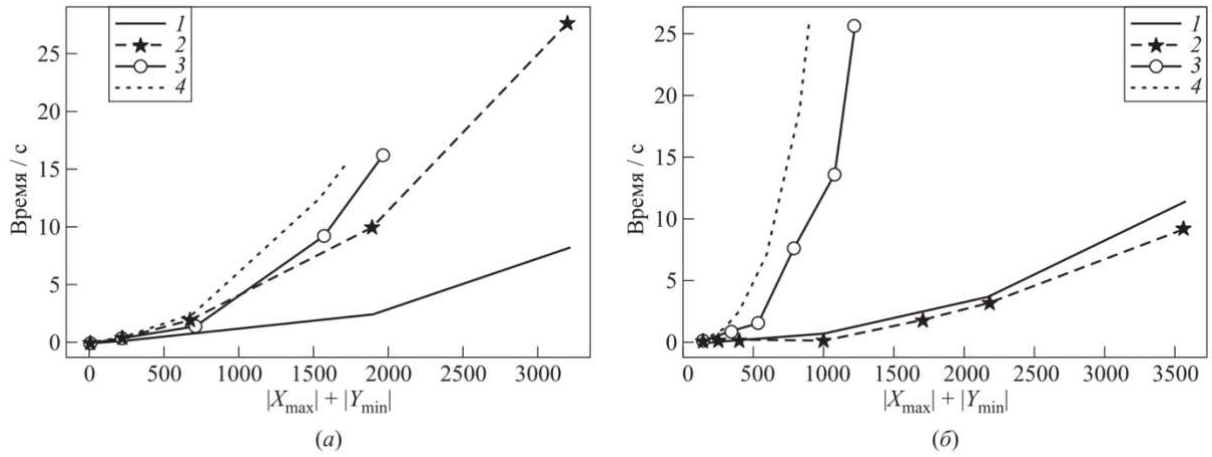

Рисунок 2 – Зависимость времени работы алгоритмов от суммы мощностей множеств X_{max} *и* Y_{min} для случая 1 (а) и 2 (б): 1 — последовательно-совместный; 2 — последовательный; 3 — совместный; 4 — Apriori

Таким образом, в настоящей главе рассмотрена задача поиска максимальных частых и минимальных нечастых элементов в данных, представленных в виде декартова произведения частичных порядков. Проведён обзор основных алгоритмов поиска частых элементов, исследованы их достоинства и ограничения. Разработан новый последовательно-совместный подход к построению множеств максимальных частых и минимальных нечастых элементов, представляющий собой синтез последовательного и совместного методов. Сложность последовательного, совместного и предлагаемого последовательно-совместного поиска обусловлена, в том числе, необходимостью рассматривать в процессе поиска труднорешаемую перечислительную задачу дискретной математики, называемую дуализацией над произведением частичных порядков.

Для случая, когда данные представлены в виде произведения конечных цепей, приведены результаты экспериментального сравнения названных подходов, а также независимого способа построения множеств X_{max} и Y_{min} , не требующего решения задачи дуализации. Эксперименты проводились на модельных задачах с применением асимптотически оптимального алгоритма дуализации над

произведением конечных цепей RUNC-M+. Результаты исследования свидетельствуют о том, что последовательно-совместный метод наиболее эффективен (требует меньших временных затрат по сравнению с другими рассмотренными методами) в случае, когда мощность множества частых элементов примерно равна мощности множества нечастых элементов. Иначе выигрывает последовательный поиск. Наихудшие показатели у независимого перечисления множеств X_{max} и Y_{min} с использованием в качестве базового алгоритма Apriori , точнее, его модификации на тестируемый случай. Таким образом, показана целесообразность применения алгоритмов дуализации для построения множеств X_{max} и Y_{min} .

1.4. Основные результаты

В настоящей главе рассмотрена задача поиска максимальных частых и минимальных нечастых элементов в данных, представленных в виде декартова произведения конечных частично упорядоченных множеств. Проведён обзор основных алгоритмов поиска частых элементов, включая алгоритмы Apriori, DepthProject и FP-Growth. Исследована модификация алгоритма DepthProject, названная алгоритмом AD, в которой проверка максимальности частого элемента осуществляется непосредственно по базе данных, а не путём сравнения с множеством ранее найденных максимальных частых элементов. Экспериментально показано, что алгоритм AD работает существенно быстрее алгоритма DepthProject, причём выигрыш во времени возрастает с увеличением числа транзакций в базе данных.

Показана связь задачи перечисления максимальных частых и минимальных нечастых элементов с задачей дуализации над произведением частичных порядков. Разработан новый последовательно-совместный алгоритм перечисления множеств максимальных частых и минимальных нечастых элементов, синтезирующий идеи последовательного и совместного методов. Доказана корректность предложенного алгоритма (утверждения 1–3). Алгоритм строит последовательность вложенных

множеств, на каждом шаге используя дуализацию для поиска новых максимальных частых элементов и одновременного накопления минимальных нечастых элементов.

Для случая данных, представленных в виде произведения конечных цепей, проведено экспериментальное сравнение последовательного, совместного и последовательно-совместного методов, а также независимого построения искомых множеств алгоритмом Apriori. Результаты показывают, что последовательно-совместный метод наиболее эффективен в случае, когда мощности множеств частых и нечастых элементов сопоставимы. При существенном различии мощностей этих множеств выигрывает последовательный метод. Наихудшие результаты продемонстрировал независимый подход на основе алгоритма Apriori, что подтверждает целесообразность применения алгоритмов дуализации.

Глава 2. Задача оптимальной расшифровки двузначной монотонной функции

Задача расшифровки двузначной монотонной функции f, определенной на декартовом произведении конечных частично упорядоченных множеств, является одной из важных задач дискретной математики. Традиционные подходы к решению данной задачи основаны на построении алгоритмов, оптимальных по Шеннону, которые имеют минимальную сложность в «худшем случае». Однако методы, ориентированные на «худший случай», имеют существенные ограничения для практического применения. Это делает актуальной разработку альтернативных подходов к расшифровке, эффективных для большинства встречающихся на практике вариантов задачи.

В настоящей главе показана связь между задачей расшифровки монотонной функции и задачей совместного перечисления максимальных частых и минимальных нечастых элементов декартова произведения частичных порядков. Показанная связь позволяет применить при решении задачи расшифровки последовательно-совместный подход к перечислению X_{max} и Y_{min} , описанный в главе 1. Данный подход к расшифровке основан на использовании асимптотически оптимального алгоритма дуализации декартова произведения цепей RUNC-M+ и ориентирован на достижение высокой эффективности в «типичном случае».

В настоящей главе рассматриваются классические и новые подходы к задаче расшифровки двузначной монотонной функции. Приводится формальная постановка задачи, описываются известные оптимальные по Шеннону алгоритмы. Ставится задача «асимптотически оптимальной» расшифровки, в рамках которой исследуется вопрос о построении алгоритма расшифровки, эффективного в «типичном случае». Предлагается новый последовательно-совместный алгоритм расшифровки рассматриваемой функции f. Приводится теоретическое обоснование корректности предложенного метода и результаты экспериментального исследования его эффективности в сравнении с традиционными алгоритмами.

2.1. Оптимальная по Шеннону расшифровка двузначной монотонной функции

Исследуется функция f, определенная на элементах декартова произведения частично упорядоченных множеств $P = P_1 \times \ldots \times P_n$, $n \geq 1$, и принимающая два значения 0 и 1. Функция f называется *монотонной*, если для любых двух элементов x и y из P таких, что $x \preccurlyeq y$, выполнено $f(x) \leq f(y)$. Функция f задаётся при помощи некоторого оператора B (оракула), который для любого $x \in P$ возвращает значение функции $f(x)$. Если $f(x) = 0$, то элемент x называется *нулём* функции f, если же $f(x) = 1$, то элемент x называется *единицей* функции f. Требуется путём «минимального» числа обращений к оператору B найти все нули и единицы функции f.

Особый интерес представляет случай, когда $P_i = \{0,1, \ldots, k-1\}$, $k \geq 2$, при $i = \overline{1,n}$, и в каждом P_i задан порядок $0 \preccurlyeq 1 \preccurlyeq \cdots \preccurlyeq k-1$. Множество P называется *k-значным n-мерным кубом* и обозначается через E_k^n. Множество E_2^n называется *булевым кубом*.

Введем понятия верхнего нуля и нижней единицы функции f, которые являются центральными для рассматриваемой задачи. Ноль функции f называется *верхним*, если он не предшествует никакому другому нулю этой функции. Единица функции f называется *нижней*, если она не следует ни за какой другой единицей этой функции. Несложно видеть, что для расшифровки функции f достаточно построить множества всех ее верхних нулей и нижних единиц, поскольку по свойству монотонности функции эти множества полностью определяют значение функции f на элементах из P. Обозначим через X_{up} множество всех верхних нулей функции f, а через Y_{low} – множество всех нижних единиц функции f. Тогда задача расшифровки может быть решена путем построения множеств X_{up} и Y_{low}.

Заметим, что понятие «минимальности» числа обращений к оператору B в постановке задачи расшифровки допускает различные формализации, от выбора которых зависит подход к решению задачи. В настоящем разделе рассматривается традиционный подход, основанный на минимизации числа обращений к оператору B в «худшем случае» (оптимальность по Шеннону). В разделе 2.2 ставится задача «асимптотически оптимальной» расшифровки, в которой рассматривается минимизация числа обращений к оракулу B в «типичном случае».

Традиционный подход к решению задачи расшифровки нацелен на случай функции, заданной на E_k^n , и основан на построении оптимального по Шеннону алгоритма. Данный подход был предложен В.К. Коробковым в 1965 году [38] и нацелен на минимизацию числа обращений к оператору B в «худшем случае».

Пусть V - множество всех двузначных монотонных функций, определенных на P . Пусть A - некоторый алгоритм, выполняющий расшифровку функций из V . Обозначим через $t_A(f)$ общее число обращений к оператору B алгоритма A при расшифровке функции $f \in V$. Под сложностью алгоритма A по Шеннону (сложностью в худшем случае) понимается величина $max_{f \in V} t_A(f)$, где максимум берется по всем функциям из V .

Алгоритм A^* называется *оптимальным по Шеннону* на множестве V, если его сложность минимальна среди всех алгоритмов, выполняющих расшифровку функций из V. Такой подход гарантирует эффективность алгоритма для наиболее трудного варианта задачи, однако может приводить к избыточным вычислительным затратам для типичных случаев.

Для булева случая задача оптимальной по Шеннону расшифровки решена Ж. Анселем в 1968 году [2]. Алгоритм из [2] основан на следующей идее. Булев куб E_2^n разбивается на непересекающиеся цепи, то есть на последовательности наборов, в которых каждый следующий набор непосредственно следует за предыдущим. На каждой цепи монотонная функция принимает сначала значение 0, а затем значение 1, поэтому граница между нулями и единицами на цепи может быть найдена с помощью двоичного поиска. Число обращений к оператору B при расшифровке

функции на одной цепи длины l не превосходит $\lfloor \log_2 l \rfloor + 1$. В [2] показано, что построенное разбиение обладает рядом специальных свойств, позволяющих при расшифровке функции на очередной цепи использовать информацию, полученную при расшифровке на ранее рассмотренных цепях. За счёт этого достигается оптимальность алгоритма по Шеннону.

Более формально, разбиение булева куба строится по индукции. Базис индукции ($n = 2$): куб E_2^n разбивается на две цепи: $C_1 = \{(0,0) \preccurlyeq (0,1) \preccurlyeq (1,1)\}$ и $C_2 = \{(1,0)\}$. Индуктивный переход: рассматривается куб E_2^{n+1}, который можно представить в виде объединения двух экземпляров куба E_2^n . По индуктивному предположению разбиение на цепи в кубах E_2^n уже построено. Куб E_2^{n+1} разбивается на цепи по следующему правилу: для двух одинаковых цепей $C = \{\alpha_1 \preccurlyeq \alpha_2 \preccurlyeq \cdots \preccurlyeq \alpha_l\}$ в кубах E_2^n строится цепь $C_1 = \{(0, \alpha_{11}, \dots, \alpha_{1n}) \preccurlyeq (0, \alpha_{21}, \dots, \alpha_{2n}) \preccurlyeq \cdots \preccurlyeq (0, \alpha_{l1}, \dots, \alpha_{ln})\}$ и цепь $C_2 = \{(1, \alpha_{11}, \dots, \alpha_{1n}) \preccurlyeq (1, \alpha_{21}, \dots, \alpha_{2n}) \preccurlyeq \cdots \preccurlyeq (1, \alpha_{l1}, \dots, \alpha_{ln})\}$, $C_1 \subset E_2^{n+1}$, $C_2 \subset E_2^{n+1}$. После построения разбиения на каждой полученной цепи применяется алгоритм двоичного поиска для определения границы между нулями и единицами монотонной функции.

В 1976 г. В. Б. Алексеевым в [1] метод из [2] обобщён на случай монотонных функций, определённых на k-значном n-мерном кубе E_k^n . Алгоритм Алексеева A_{opt} является оптимальным по Шеннону в случае $k = 2$ и близок по сложности к оптимальному в случае $k > 2$. Алгоритм A_{opt} работает в два основных этапа. На первом этапе множество P разбивается на непересекающиеся подмножества, каждое из которых является цепью. На втором этапе выполняется расшифровка функции f на каждой построенной цепи с помощью алгоритма двоичного поиска.

Пусть $i \in \{2, \dots, n\}$, $r \in \{0, \dots, k-1\}$. Положим

$$S_r^i = \{(\alpha_1, \dots, \alpha_{i-1}, r) | (\alpha_1, \dots, \alpha_{i-1}) \in E_k^{i-1}\} .$$

Процесс разбиения куба E_k^n на непересекающиеся цепи происходит путем последовательного построения разбиений на непересекающиеся цепи кубов меньшей размерности.

На первом шаге рассматривается куб E_k^1 , представляющий собой цепь согласно установленному частичному порядку. Пусть на шаге $i-1$, $2 \le i \le n$ куб E_k^{i-1} разбит на непересекающиеся цепи. Тем самым на непересекающиеся цепи разбито каждое из множеств S_r^i , $r \in \{0, \dots, k-1\}$. Далее построенные разбиения множеств $S_0^i, \dots S_{k-1}^i$ изменяются следующим образом. Сначала в S_0^i добавляются все максимальные элементы цепей из построенных разбиений для множеств $S_1^i, \dots S_{k-1}^i$ при этом все добавленные к S_0^i элементы удаляются из множеств $S_1^i, \dots S_{k-1}^i$. Затем аналогичная процедура проводится для измененной последовательности $S_1^i, \dots S_{k-1}^i$. В результате, учитывая, что $E_k^i = \bigcup_{r=0}^{k-1} S_r^i$, получается требуемое разбиение для куба E_k^i .

Построенные на первом этапе работы алгоритма цепи просматриваются в порядке неубывания их мощности. Пусть C_i - очередная цепь. Если для некоторого элемента $x \in C_i$ известно, что $x \preccurlyeq y$, где y - ноль, принадлежащий ранее просмотренной цепи C_j $(j < i)$, то x – ноль цепи C_i . Аналогично, если для некоторого элемента $x \in C_i$ известно, что $y \preccurlyeq x$, где y - единица, принадлежащая ранее просмотренной цепи C_j $(j < i)$, то x - единица цепи C_i .

Таким образом, цепь C_i делится на три отрезка: сначала следуют найденные нули, затем следуют элементы, на которых значение функции f неизвестно, после чего следуют найденные единицы. Для расшифровки цепи C_i на втором отрезке запускается алгоритм двоичного поиска. При этом для определения значения функции f происходит обращение к оператору B .

Пусть A_0 - любой алгоритм расшифровки функций из V с наименьшей сложностью по Шеннону. Для алгоритма Алексеева A_{opt} справедливо соотношение

$$\frac{t_{A_{opt}}(f)}{t_{A_0}} \le 0.5(\log_2 k + 1) .$$

Это означает, что алгоритм Алексеева является оптимальным по Шеннону при $k = 2$ и имеет сложность, отличающуюся от оптимальной не более чем в $0.5(\log_2 k + 1)$ раз при $k > 2$.

В [1] приведён пример практического приложения задачи расшифровки из области геологоразведки: проектное учреждение, получающее информацию о запасах полезных ископаемых в некотором районе, должно принять решение об освоении этого района. Данные о запасах каждого полезного ископаемого округляются до значений из конечной стандартной шкалы, а решение является монотонной функцией от этих данных: если запасы каждого полезного ископаемого не убывают, то не может убывать целесообразность освоения района. Составление таблицы решений сводится к расшифровке монотонной функции.

2.2. Асимптотически оптимальная расшифровка двузначной монотонной функции

Оптимальный по Шеннону подход к расшифровке, описанный в разделе 2.1, оценивает сложность алгоритма числом обращений к оператору B в «худшем случае», то есть для самого сложного варианта задачи. Однако методы, ориентированные на «худший случай», имеют существенные ограничения для практического применения. В связи с этим актуальным является построение алгоритмов расшифровки, эффективных в «типичном случае».

В диссертационной работе ставится задача «асимптотически оптимальной» расшифровки двузначной монотонной функции f. Пусть V — множество всех двузначных монотонных функций, определённых на P; A — некоторый алгоритм, выполняющий расшифровку функций из V; $t_A(f)$ — число обращений к оператору B алгоритма A при расшифровке функции $f \in V$. Под сложностью алгоритма A в «типичном случае» понимается величина

$$T_{avg}(A) = \frac{1}{|V|}\sum_{f \in V} t_A(f) .$$

Алгоритм A^* называется *оптимальным в «типичном случае»* на множестве V, если его сложность $T_{avg}(A^*)$ минимальна среди всех алгоритмов, выполняющих расшифровку функций из V. Таким образом, если оптимальный по Шеннону

алгоритм минимизирует $\max_{f \in V} t_A(f)$, то оптимальный в «типичном случае» алгоритм минимизирует $T_{avg}(A)$.

Подход к «асимптотически оптимальной» расшифровке, предлагаемый в настоящей работе, основан на связи задачи расшифровки с задачей совместного перечисления максимальных частых и минимальных нечастых элементов декартова произведения частичных порядков. Установим эту связь.

Пусть задана база данных $D(P)$ и порог $s \leq |D(P)|$, $s \in \mathbb{N}$. Напомним (см. главу 1), что частотой элемента $x \in P$ называется величина $\nu(x)$, равная числу транзакций в $D(P)$, предшествующих x. Если $\nu(x) \geq s$, то элемент x называется s-частым, иначе x – s-нечастый элемент. На множестве P определим двузначную монотонную функцию $f_{D,s}$ которая принимает значения 0 и 1 соответственно на s-частых элементах и s-нечастых элементах этого множества. Фактически $f_{D,s}$ задается при помощи оператора B_D , который для произвольного $x \in P$ выдает значение $f_{D,s}(x)$ путем вычисления частоты встречаемости x .

Нетрудно видеть, что нижние единицы Y_{low} функции $f_{D,s}$ взаимно однозначно соответствуют минимальным s-нечастым элементам множества P, а верхние нули X_{up} – максимальным s-частым элементам. Действительно, если элемент x является верхним нулём функции $f_{D,s}$, то $f_{D,s}(x) = 0$ и, следовательно, x является s-частым, и при этом любой следующий за x элемент является s-нечастым (единицей функции $f_{D,s}$), то есть x – максимальный s-частый элемент по определению. Аналогично устанавливается соответствие между нижними единицами и минимальными s-нечастыми элементами.

Связь между значениями функции $f_{D,s}$ и частотой элементов множества P схематично изображена на рисунке 3.

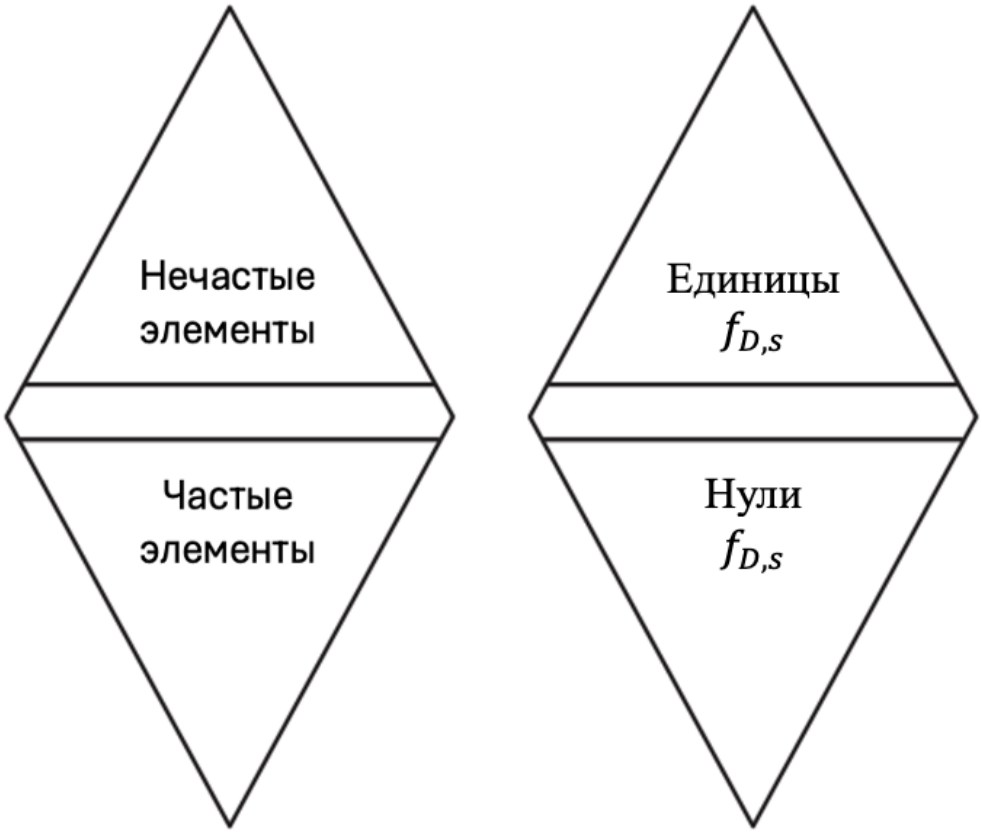

Рисунок 3 – Связь между значениями функции $f_{D,\,s}$ и частотой элементов множества P .

Таким образом, для решения задачи расшифровки двузначной монотонной функции, определенной на декартовом произведении конечных частично упорядоченных множеств, может быть применён последовательно-совместный алгоритм перечисления множеств X_{max} и Y_{min}, описанный в главе 1. В терминах задачи расшифровки множество X_{max} соответствует множеству верхних нулей X_{up}, а множество Y_{min} – множеству нижних единиц Y_{low}. Для решения задачи дуализации, возникающей на каждом шаге последовательно-совместного алгоритма, применяется асимптотически оптимальный алгоритм дуализации декартова произведения цепей RUNC-M+ [29, 30]. Подход к построению асимптотически оптимальных алгоритмов для дискретных перечислительных задач предложен Е. В. Дюковой в [20].

Корректность последовательно-совместного алгоритма расшифровки основана на следующих утверждениях 1–3. Утверждения 1-3 являются аналогами

соответствующих утверждений из главы 1, переформулированными в терминах верхних нулей и нижних единиц функции $f_{D,s}$.

Утверждение 4. *Если* $X \subset X_{up}$ *, то* $I(X^-)$ *содержит хотя бы один ноль функции* $f_{D,s}$ *. Аналогично, если* $Y \subset Y_{low}$ *, то* $I(Y^+)$ *содержит хотя бы одну единицу функции* $f_{D,s}$ *.*

Доказательство. Пусть $X \subset X_{up}$, $x \in X_{up} \setminus X$. Из того, что x не сравним ни с одним другим элементом множества X_{up} , следует, что $x \in P \setminus X^-$. Таким образом, в $I(X^-)$ существует элемент q такой, что $q \preccurlyeq x$ и q – ноль функции $f_{D,s}$. Утверждение для множества $I(Y^+)$ доказывается аналогично.

Утверждение 5. *Если* $X \subset X_{up}$ *и* $y \in I(X^-)$ *является единицей* функции $f_{D,s}$ *, то* y *– нижняя единица функции* $f_{D,s}$ *. Аналогично, если* $Y \subset Y_{low}$ *и* $x \in I(Y^+)$ *является нулем функции* $f_{D,s}$ *, то* x *– верхний ноль функции* $f_{D,s}$ *.*

Доказательство. Пусть $X \subset X_{up}$ и $y \in I(X^-)$ – единица функции $f_{D,s}$. Предположим, что y не является нижней единицей функции $f_{D,s}$, то есть $y \notin Y_{low} = I(X_{up}^-)$. Тогда, так как y – единица функции $f_{D,s}$, то в $P \setminus X_{up}^-$ найдется нижняя единица функции $f_{D,s}$ z такая, что $z \neq y$, $z \preccurlyeq y$. Из $(P \setminus X_{up}^-) \subseteq (P \setminus X^-)$, следует, что $z \in P \setminus X^-$, что противоречит условию $y \in I(X^-)$. Полученное противоречие доказывает первую часть утверждения. Вторая часть утверждения доказывается аналогично.

Утверждение 6. *Пусть* $X \subseteq X_{up}$, $Y \subseteq Y_{low}$ *. Тогда* $I(X^-) = Y$ *(или* $I(Y^+) = X$ *) тогда и только тогда, когда* $X = X_{up}$, $Y = Y_{low}$.

Доказательство. Пусть $X \subset X_{up}$. Из утверждения 4 следует, что $I(X^-)$ содержит хотя бы один ноль функции $f_{D,s}$. Однако множество Y не содержит нулей функции $f_{D,s}$, следовательно $I(X^-) \neq Y_{low}$. Если же $X = X_{up}$, то $I(X^-) = Y_{low}$. Таким образом, $I(X^-) = Y$ тогда и только тогда, когда $X = X_{up}$, $Y = Y_{low}$.

Алгоритм последовательно-совместной расшифровки двузначной монотонной функции, определенной на P , работает итеративно и строит последовательность вложенных множеств $X_1 \subset X_2 \subset \cdots \subset X_{up}$. Положим $X_0 = \emptyset$.

Шаг 1. Рассматривается множество $X_1 = \{x\}$, где x – произвольный верхний ноль f.

Шаг $i + 1$ ($i \geq 1$). Решается задача дуализации множества $X_i \setminus X_{i-1}$. Пусть множество Z есть результат дуализации $X_i \setminus X_{i-1}$. Согласно утверждению 4, множество Z содержит хотя бы один ноль функции f. Для каждого нуля из Z находится один содержащий его верхний ноль. Все найденные верхние нули, которые не содержатся в множестве X_i, добавляются к X_i, формируя в результате множество X_{i+1}. Если же все найденные верхние нули уже содержатся в X_i, то происходит дуализация множества X_i, в результате чего формируется множество $I(X_i^-)$. Если в $I(X_i^-)$ нет нулей, то, согласно утверждению 6, следует, что $I(X_i^-) = Y_{\text{low}}$, $X_i = X_{\text{up}}$, и алгоритм завершает свою работу. Иначе для каждого нуля из $I(X_i^-)$ находится один содержащий его верхний ноль, который пополняет множество X_i, формируя в результате множество X_{i+1}.

2.3. Экспериментальное исследование

В ходе экспериментального исследования рассмотрен случай функции, заданной на E_k^n. Для проведения экспериментов необходимо задать конкретную реализацию оператора B. В настоящей работе оператор B реализован через задание базы данных $D(P)$ и порога s: для произвольного $x \in P$ оператор B_D вычисляет частоту встречаемости элемента x в $D(P)$ и возвращает значение функции $f_{D,s}(x)$. Таким образом, в экспериментальном исследовании решается задача расшифровки функции $f_{D,s}$, которая описана в разделе 2.2.

Проведено сравнение двух алгоритмов расшифровки функции $f_{D,s}$, описанных в разделах 2.1 и 2.2. Алгоритмы реализованы на языке C++. При реализации последовательно-совместного алгоритма поиска верхних нулей и нижних единиц функции $f_{D,s}$ использовался асимптотически оптимальный алгоритм дуализации над произведением k-значных цепей RUNC-M+ [29, 30].

Эксперименты проведены для случайных баз данных D с элементами из E_k^n и числом транзакций $m = 100$. Данные выбирались из равномерного распределения. Результаты усреднялись по 20 независимым запускам.

На рисунках 4–7 представлены графики зависимости среднего времени расшифровки от числа переменных n при различных значениях k и s .

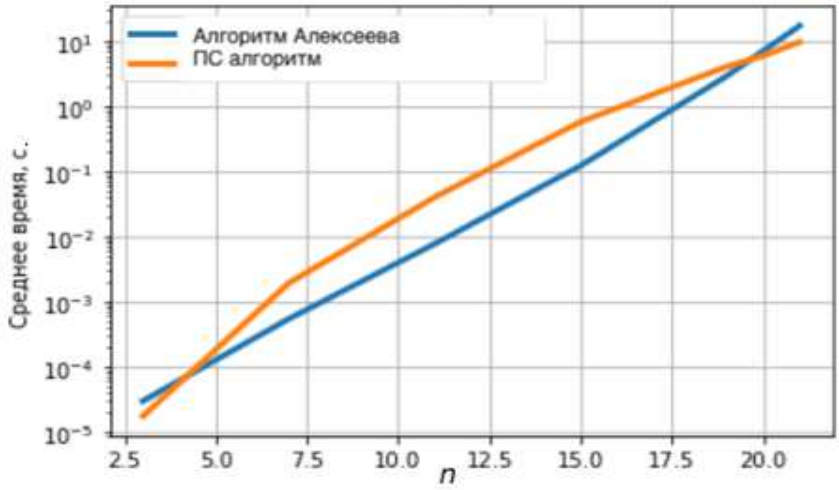

Рисунок 4 – Время расшифровки функции $f_{D,s}$ при $k = 2$, $s = 10$.

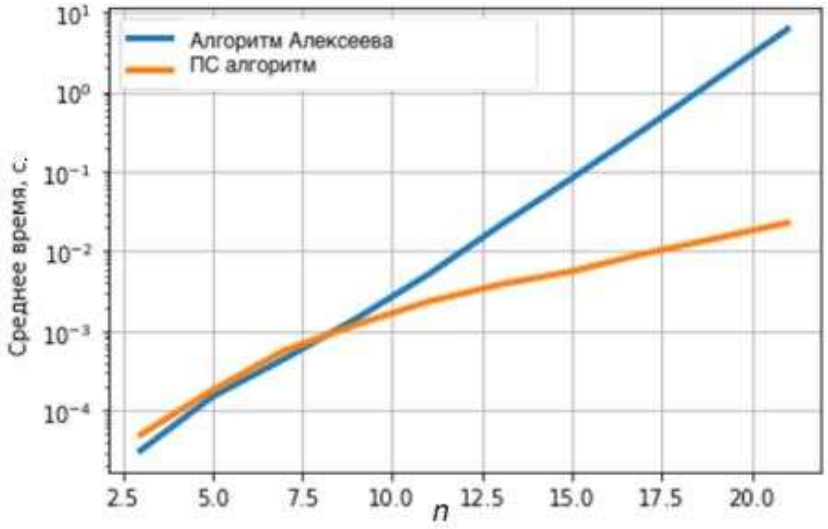

Рисунок 5 – Время расшифровки функции $f_{D,s}$ при $k = 2$, $s = 30$.

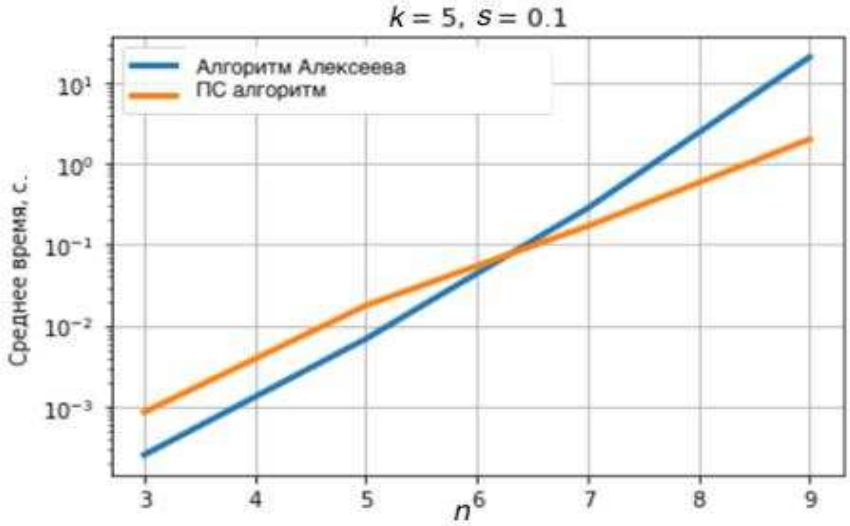

Рисунок 6 – Время расшифровки функции $f_{D,s}$ при $k = 5$, $s = 10$.

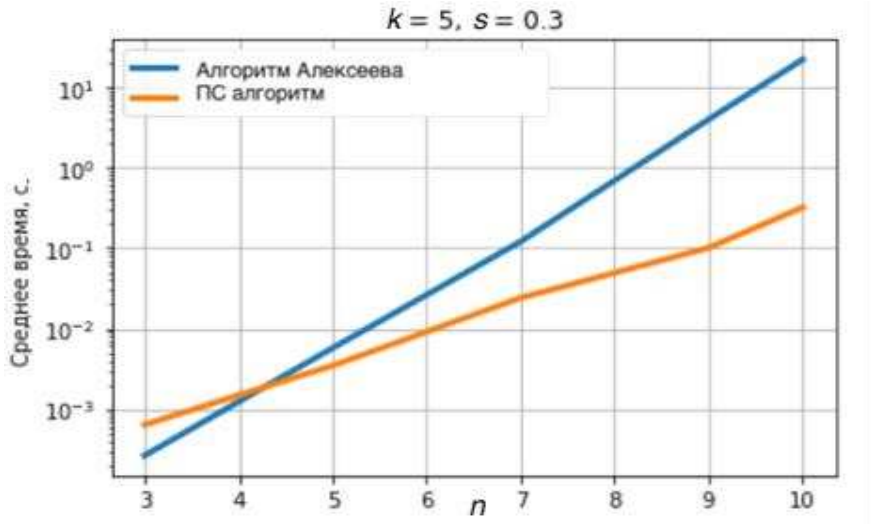

Рисунок 7 – Время расшифровки функции $f_{D,s}$ при $k = 5$, $s = 30$.

Из представленных графиков следует, что при $s = 0{,}3$ и $n > 5$ независимо от значения k последовательно-совместный алгоритм (ПС алгоритм) работает быстрее алгоритма Алексеева. Алгоритм Алексеева более эффективен при $s = 0{,}1$, $k = 2$, но при этом время его работы растёт быстрее с ростом n. В случае $k = 5$, $n > 7$ последовательно-совместный алгоритм на порядки быстрее алгоритма Алексеева.

Различие в поведении алгоритмов обусловлено следующим. Алгоритм Алексеева основан на разбиении куба E_k^n на цепи и применении двоичного поиска на каждой цепи. При небольших значениях k число цепей в разбиении невелико,

длины нерасшифрованных участков цепей достаточны для эффективного применения двоичного поиска, и сложность алгоритма Алексеева близка к оптимальной по Шеннону. При больших значениях k число цепей в разбиении возрастает, а длины нерасшифрованных участков цепей остаются небольшими, что снижает эффективность двоичного поиска. Последовательно-совместный алгоритм работает непосредственно с множествами верхних нулей и нижних единиц и использует дуализацию для их итеративного построения, что оказывается более эффективным при больших k.

Таблица 2 – Среднее число обращений к оператору B_D , $m = 100$

n, k, s	*Алгоритм Алексеева*	*ПС алгоритм*
5, 3, 10	**111**	529
5, 3, 30	**56**	343
5, 20, 30	25484	**3039**
10, 3, 10	**2709**	10172
10, 3, 30	**323**	3111
10, 5, 10	45528	**31791**
15, 3, 30	**1142**	13940

Как видно из таблицы 2, в случае небольших значений k , независимо от значения n , наилучший результат по числу обращений к оператору B_D показывает алгоритм Алексеева. Заметим, что при небольших значениях k сложность этого алгоритма почти оптимальна по Шеннону: согласно оценке из раздела 2.1, при $k \leq 3$ отношение сложности алгоритма Алексеева к сложности оптимального алгоритма не превосходит $0{,}5(\lfloor \log_2 3 \rfloor + 1) = 1$, то есть алгоритм является оптимальным. При значениях $k \geq 5$ лучший результат по числу обращений к оператору B_D показывает последовательно-совместная расшифровка функции $f_{D,s}$, что объясняется ростом отношения $0{,}5(\lfloor \log_2 k \rfloor + 1)$ и, как следствие, отдалением алгоритма Алексеева от оптимума.

Последовательно-совместный алгоритм при малых k делает больше обращений к оператору B_D, чем алгоритм Алексеева. Это обусловлено тем, что в текущей реализации последовательно-совместного алгоритма поиск верхнего нуля, содержащего данный ноль функции f, осуществляется путём последовательного увеличения координат текущего элемента, то есть фактически выполняется линейный поиск. Замена линейного поиска на более эффективную процедуру (например, двоичный поиск вдоль каждой координаты) позволила бы существенно сократить число обращений к оператору B_D. Однако даже при линейном поиске по времени работы ПС алгоритм оказывается быстрее при достаточно больших n, поскольку общее время работы определяется не только числом обращений к оператору, но и временем построения и обработки цепей в алгоритме Алексеева, которое возрастает с ростом размера задачи.

2.4. Основные результаты

В настоящей главе рассмотрена задача расшифровки двузначной монотонной функции, определённой на декартовом произведении конечных частично упорядоченных множеств.

Поставлена задача «асимптотически оптимальной» расшифровки, в рамках которой сложность алгоритма оценивается средним числом обращений к оператору B по всем функциям из V. Показана связь между задачей расшифровки и задачей совместного перечисления максимальных частых и минимальных нечастых элементов частичных порядков, что позволяет применить последовательно-совместный алгоритм из главы 1. На базе последовательно-совместного подхода к перечислению максимальных частых и минимальных нечастых элементов частичных порядков построен новый алгоритм расшифровки функции f, определённой на декартовом произведении конечных частично упорядоченных множеств. Доказана корректность предложенного алгоритма.

В ходе экспериментального исследования рассмотрен случай функции, заданной на E_k^n, и показана эффективность предлагаемого алгоритма: в случае

больших значений k и n предложенный метод работает существенно быстрее ориентированного на «худший случай» алгоритма Алексеева и требует меньшего числа обращений к оператору B. Экспериментально выявлены условия применимости каждого из подходов: алгоритм Алексеева предпочтителен при небольших k, когда его сложность близка к оптимальной по Шеннону, тогда как последовательно-совместный алгоритм выигрывает при больших k и n .

Глава 3. Корректная классификация над произведением частичных порядков: новые модели логических классификаторов

Задача классификации по прецедентам является одной из центральных задач машинного обучения и интеллектуального анализа данных. Среди различных подходов к решению этой задачи важное место занимают методы логического анализа данных, основанные на применении аппарата дискретной математики. Обучение логического классификатора сводится к поиску в исходных данных информативных фрагментов описаний прецедентов – представительных элементарных классификаторов, позволяющих различать объекты из разных классов.

В существующих направлениях логической классификации на этапе обучения для каждого класса K строятся представительные ЭК определённого вида. В направлении CVP [3, 20–22, 25, 33, 39, 46] ищутся тупиковые представительные ЭК, и на этапе их построения возникает труднорешаемая задача монотонной дуализации. В направлении LAD [36, 53, 56, 66] ищутся максимальные логические закономерности, при этом решаются сложные оптимизационные задачи линейного программирования. В направлении FCA [40, 43, 44] строятся ДСМ-представительные ЭК, и возникают дискретные перечислительные задачи. В [28] показано, что во всех трёх направлениях на этапе обучения фактически задаётся некоторый частичный порядок на множестве представительных ЭК и строится множество максимальных относительно заданного порядка ЭК.

В настоящей главе исследуется иной подход к обучению логических классификаторов, основанный на применении методов поиска частых элементов в данных, рассмотренных в главе 1. Подход основан на поиске в описаниях прецедентов каждого класса K часто встречающихся фрагментов специального вида, называемых правильными ЭК, с последующим отбором среди них представительных для класса K . В рамках нового подхода разработаны алгоритмы

REC и REC+. Алгоритм REC работает с данными, представленными в декартова произведения антицепей. Алгоритм REC+ является модификацией REC для случая декартова произведения цепей.

Для обоснования вычислительной эффективности предлагаемых алгоритмов исследованы метрические (количественные) свойства множества правильных ЭК. Получены асимптотические оценки типичного числа и типичного ранга правильных ЭК как для случая антицепей, так и для случая цепей. Сравнение с известными оценками числа тупиковых ЭК, возникающих в направлении CVP, показывает, что число правильных ЭК растёт существенно медленнее, что теоретически объясняет преимущество предлагаемого подхода по скорости обучения.

Экспериментальное исследование выполнено на реальных задачах и на случайных модельных данных. Проведено сравнение предложенных алгоритмов с классическим алгоритмом голосования по тупиковым представительным ЭК, а также с такими известными методами машинного обучения, как случайный лес и логистическая регрессия. Показано, что алгоритм REC существенно превосходит по скорости традиционный алгоритм из CVP, не уступая ему по качеству классификации, а использование линейных порядков в алгоритме REC+ позволяет дополнительно повысить качество на большинстве рассмотренных задач.

Глава организована следующим образом. В разделе 3.1 приведена формальная постановка задачи классификации по прецедентам, введены основные понятия и дан обзор трёх основных направлений логической классификации: CVP, LAD и FCA. В разделе 3.2.1 введено понятие правильного элементарного классификатора, описан алгоритм REC для случая декартова произведения антицепей, доказана теорема о метрических свойствах множества правильных элементарных классификаторов и представлены результаты экспериментального исследования на реальных и модельных данных. В разделе 3.2.2 описан алгоритм REC+ для случая декартова произведения цепей, приведены соответствующие теоретические оценки и результаты экспериментов. В разделе 3.3 сформулированы основные выводы по главе.

3.1. Логический подход к задаче классификации по прецедентам

Задача классификации по прецедентам рассматривается в следующей постановке. Исследуется некоторое множество объектов M. Объекты из M описываются в системе числовых *признаков* $x_1, \dots, x_n$. Признак x_j , $j = 1, \dots, n$, принимает ограниченное число допустимых значений, которые кодируются целыми числами. Известно, что M представимо в виде объединения l непересекающихся подмножеств $K_1, \dots, K_l$, называемых *классами*. Дан набор объектов из M , о которых известно, каким классам они принадлежат. Это прецеденты или обучающие объекты. Требуется на базе анализа множества прецедентов построить алгоритм, определяющий класс любого объекта из M .

Основное достоинство логического подхода к задаче классификации – возможность получения результата при отсутствии дополнительных предположений вероятностного характера и при небольшом числе прецедентов. Не требуется также задание метрики в пространстве описаний объектов. Построение логических процедур классификации особенно актуально в тех случаях, когда набор большого числа прецедентов затруднён или вообще невозможен. Например, такая ситуация имеет место в задачах прогнозирования месторождений редких металлов и свойств твёрдых сплавов [4, 5]. При этом большое внимание уделяется вопросам синтеза корректных алгоритмов, то есть алгоритмов, безошибочно классифицирующих материал обучения.

Пусть N_j – конечное множество допустимых значений признака x_j , на котором задан частичный порядок, $j = 1, \dots, n$. Для обозначения того, что $b \in N_j$ следует за $a \in N_j$, используется запись $a \preccurlyeq b$. Множество M представимо в виде декартова произведения $M = N_1 \times \dots \times N_n$. Пусть $S = (a_1, \dots, a_n)$ и $S^* = (b_1, \dots, b_n)$ – объекты из M. Будем считать, что элемент S^* *следует* за элементом S, если $a_j \preccurlyeq b_j$ при $j = 1, \dots, n$. В работе рассматриваются два случая:

1) множество M является декартовым произведением антицепей, то есть элементы каждого множества N_j попарно несравнимы;

2) множество M является декартовым произведением цепей, то есть любые два элемента любого множества N_j сравнимы.

Центральным понятием логического подхода к классификации является элементарный классификатор. Пусть $H = \{x_{j_1}, \dots, x_{j_r}\} \subseteq \{x_1, \dots, x_n\}, j_1 < \dots < j_r$, – набор из r различных признаков; $\sigma = (\sigma_1, \dots, \sigma_r)$, где $\sigma_i \in N_{j_i}$ и σ_i не является наибольшим элементом в N_{j_i}, $i = 1, \dots, r$. Пара (σ, H) называется *элементарным классификатором (ЭК) ранга* r.

Объект $S = (a_1, \dots, a_n)$ из M *содержит* ЭК (σ, H), если $a_{j_i} \preccurlyeq \sigma_i$ при $i = \overline{1,r}$. В случае антицепей это условие сводится к равенству $a_{j_i} = \sigma_i$.

Будем считать, что множество N_i, $i = \overline{1,r}$, содержит *наибольший* элемент h_i, который следует за любым другим элементом этого множества. Если такого элемента нет, то множество N_i можно дополнить таким элементом. В этом случае любому ЭК (σ, H) можно поставить в соответствие элемент $\tilde{\sigma} = (\tilde{\sigma}_1, \dots, \tilde{\sigma}_n)$ из M, такой что: 1) $\tilde{\sigma}_{j_i} = \sigma_i$, $i = \overline{1,r}$; 2) $\tilde{\sigma}_i = h_i$ при $i = \overline{1,n}$, $i \notin \{j_1, \dots, j_r\}$. Тогда считается, что ЭК (σ, H) следует за ЭК (σ', H'), если $\tilde{\sigma}'$ следует $\tilde{\sigma}$ за согласно порядку, заданному на элементах из M.

Пусть $K \in \{K_1, \dots, K_l\}$, $\overline{K} = \{K_1, \dots, K_l\} \setminus \{K\}$. Через $R(K)$ и $R(\overline{K})$ обозначаются совокупности прецедентов из K и из $\overline{K}$ соответственно; $|R(K)| = m_1$, $|R(\overline{K})| = m_2$.

Введём основные определения, связывающие понятия логической классификации с понятиями теории поиска частых и нечастых элементов, рассмотренными в главе 1. ЭК (σ, H) называется s *-частым* в K, если не менее s прецедентов из $R(K)$ содержат (σ, H). Фактически, s -частый ЭК (σ, H) соответствует s -частому элементу $\tilde{\sigma}$ при базе данных $D(M) = R(K)$. ЭК (σ, H) называется *максимальным* s *-частым* в K, если он s -частый в K и не существует другого s -частого ЭК в K, следующего за данным.

ЭК (σ, H) называется s *-нечастым* в $\overline{K}$, если менее s прецедентов из $R(\overline{K})$ содержат (σ, H). Фактически, s -нечастый ЭК (σ, H) соответствует s -нечастому элементу $\tilde{\sigma}$ при базе данных $D(M) = R(\overline{K})$. ЭК (σ, H) называется *минимальным* s *-нечастым* в $\overline{K}$, если он s -нечастый в $\overline{K}$ и не существует другого s -нечастого в $\overline{K}$ ЭК, предшествующего данному.

ЭК (σ, H) называется *представительным* для класса K, если хотя бы один прецедент из $R(K)$ содержит (σ, H) и ни один прецедент из $R(\overline{K})$ не содержит (σ, H), то есть (σ, H) является 1-частым в K и 1-нечастым в $\overline{K}$. Представительные ЭК позволяют безошибочно классифицировать обучающие объекты и являются основой всех направлений логической классификации. Представительный ЭК класса K называется *тупиковым представительным*, если он является минимальным 1-нечастым в $\overline{K}$. Заметим, что для существования представительных ЭК необходимо, чтобы описания объектов из разных классов были несравнимы [31]. Это всегда может быть обеспечено путём дублирования признаковых описаний объектов с обратным отношением порядка [29–31]. Для случая антицепей указанное условие эквивалентно требованию того, чтобы описания любых двух прецедентов из разных классов не совпадали.

Процедуры корректного голосования (CVP). Отечественное направление логической классификации, именуемое процедурами корректного голосования (Correct Voting Procedures, или CVP), развивается с 70-х годов прошлого века [3, 20–22, 25, 33, 39, 46]. Понятие теста, введённое С. В. Яблонским для проверки контактных схем [46], стало базовым для первых моделей логических процедур классификации. Основы направления заложены в трудах М. М. Бонгарда [5] и М. Н. Вайнцвайга, предложившего алгоритм распознавания «Кора» [6]. Модель распознающих алгоритмов с представительными наборами и системами опорных множеств предложена Л. В. Баскаковой и Ю. И. Журавлёвым [3]. Описание моделей CVP с использованием понятия элементарного классификатора впервые дано в [25, 33]. Существенное развитие направление получило в работах научной школы академика РАН Ю. И. Журавлёва [3, 20–27, 33–35, 39].

В CVP наиболее информативными считаются тупиковые представительные ЭК. Описание этих ЭК может быть дано с использованием терминологии теории логических функций. Пусть $f_K(x_1, \dots, x_n)$ – двузначная частичная (не всюду определённая) логическая функция, принимающая на целочисленных описаниях прецедентов класса K и других классов соответственно значение 1 и 0. Функция f_K называется *характеристической функцией* класса K. Решение задачи классификации заключается в доопределении f_K на наборах, не входящих в обучающую выборку. Тогда представительный ЭК класса K – это *допустимая конъюнкция (ДК)* для функции f_K, интервал истинности которой имеет непустое пересечение с множеством единиц функции f_K и пустое пересечение с множеством нулей этой функции. Тупиковый представительный ЭК класса K – это *максимальная ДК* для f_K , то есть ДК, которая перестаёт быть допустимой при удалении хотя бы одного сомножителя.

На этапе обучения для каждого класса K строится множество $P_1(K)$ тупиковых представительных ЭК. Оценка принадлежности распознаваемого объекта S классу K вычисляется по формуле [28, 29]

$$\Gamma(S,K) = \frac{1}{|P_1(K)|} \sum_{(\sigma,H)\in P_1(K)} |R_K(\sigma,H)|\, \Omega(\sigma,H,S)\,,$$

где $R_K(\sigma,H)$ – множество прецедентов из $R(K)$, содержащих (σ,H) ; $\Omega(\sigma,H,S)$ – величина, равная 1, если объект S содержит (σ,H) , и 0 иначе. Объект S относится к классу с наибольшей оценкой $\Gamma(S,K)$. При наличии нескольких классов с одинаковой максимальной оценкой классификатор отказывается от распознавания.

Построение множества тупиковых представительных ЭК класса K осуществляется в два этапа [31]. Сначала анализируется множество прецедентов $R(\overline{K})$ и строятся минимальные 1- нечастые в $\overline{K}$ ЭК, также называемые тупиковыми покрытиями для $R(\overline{K})$. Поиск тупиковых покрытий для $R(\overline{K})$ сводится к задаче дуализации над произведением частичных порядков, рассмотренной в главе 1. Затем из найденных ЭК отбираются те, которые содержатся хотя бы в одном прецеденте из $R(K)$, то есть являются представительными для класса K.

Основная вычислительная сложность в моделях CVP связана с первым этапом обучения – поиском минимальных нечастых ЭК путём дуализации данных из $R(\overline{K})$. Труднорешаемость монотонной дуализации обусловлена двумя аспектами: экспоненциальным ростом числа решений при увеличении размера задачи и сложностью нахождения одного решения [29].

Следует подчеркнуть, что вычислительная трудоёмкость обучения в моделях CVP определяется в том числе мощностью множества $R(\overline{K})$, поскольку именно это множество подвергается дуализации. В задачах с числом классов $l > 2$ мощность $R(\overline{K})$ обычно существенно превышает мощность $R(K)$: если, например, классы представлены приблизительно равным числом прецедентов, то $m_2 = (l - 1)m_1$. Таким образом, при большом числе классов время обучения модели CVP резко возрастает. Данное обстоятельство является одним из существенных ограничений направления CVP и мотивирует поиск альтернативных подходов, основанных на первоначальном анализе меньшего по мощности множества $R(K)$.

Логический анализ данных (LAD). Направление логического анализа данных (Logical Analysis of Data, или LAD) предложено П. Хаммером [56, 66] и развивается в основном за рубежом. В России методы LAD предложены практически параллельно с зарубежными авторами и получили развитие в работах школы Ю. И. Журавлёва [36, 41]. Методы LAD, как правило, работают с бинарными признаками, получаемыми из целочисленных посредством процедуры one-hot кодирования.

В терминологии LAD допустимая конъюнкция для характеристической функции f_K называется *логической закономерностью (ЛЗ)* класса K [28, 53]. В отличие от CVP, где ищутся тупиковые (минимальные по включению) представительные ЭК, в LAD ищутся ЭК, обладающие наибольшей частотой встречаемости в прецедентах класса K. Представительный ЭК (σ, H) класса K называется *наибольшей ЛЗ* (maximum pattern), если $|R_K(\sigma, H)| \geq |R_K(\sigma', H')|$ для любого другого представительного ЭК (σ', H') класса K [53]. Представительный ЭК (σ, H) называется *сильной ЛЗ* (strong pattern), если не существует другого

представительного ЭК (σ', H') класса K такого, что $R_K(\sigma, H) \subset R_K(\sigma', H')$ [53]. На этапе обучения для каждого класса K строится множество $P_2(K)$ сильных (или наибольших) ЛЗ, при этом возникают сложные оптимизационные задачи линейного программирования. Оценка принадлежности распознаваемого объекта S классу K вычисляется аналогично CVP – на основе процедуры голосования по элементам множества $P_2(K)$.

Анализ формальных понятий (FCA). Направление анализа формальных понятий (Formal Concept Analysis, или FCA) в основном представлено классификаторами, вдохновленными идеями ДСМ-метода В. К. Финна [40, 43, 44]. Методы FCA, как и методы LAD, обычно работают с бинарной информацией. Базовый классификатор из FCA нацелен на построение множества $P_3(K)$ таких представительных ЭК класса K, каждый из которых является максимальным s-частым в K для некоторого $s \geq 1$, то есть содержится в s прецедентах класса K и не допускает добавления нового признака без уменьшения числа содержащих его прецедентов. При построении множества $P_3(K)$ возникают дискретные перечислительные задачи, которые алгоритмически менее сложны, чем задача монотонной дуализации, однако также требуют значительных вычислительных ресурсов.

FCA-классификатор использует решающее правило, существенно отличающееся от процедуры голосования, применяемой в CVP и LAD. Объект S относится к классу K , если S содержит хотя бы один ЭК из $P_3(K)$ и не содержит ни одного ЭК из $P_3(K')$ для всех $K' \neq K$. В противном случае алгоритм отказывается от классификации. Такое строгое правило приводит к большому числу отказов. В [32] показано, что замена решающего правила на процедуру голосования, аналогичную используемой в CVP, приводит к повышению качества распознавания.

Общая схема логического классификатора. Несмотря на различия в терминологии, все три направления логической классификации имеют единую структуру, установленную в [28]. Обозначим через $P(K)$ множество всех

представительных ЭК класса K. В [28] показано, что каждое из направлений на этапе обучения задаёт некоторый частичный порядок на множестве $P(K)$ и ищет максимальные относительно этого порядка элементы. А именно, в [28] введены четыре отношения порядка на $P(K)$ и доказаны следующие утверждения.

1. Зададим частичный порядок $\preccurlyeq_1$ на $P(K)$: ЭК (σ_2, H_2) следует за ЭК (σ_1, H_1), если $H_2 \subseteq H_1$ и $\sigma_2 \subseteq \sigma_1$. Тогда ЭК (σ, H) является тупиковым представительным для класса K тогда и только тогда, когда (σ, H) – максимальный элемент $P(K)$ относительно порядка $\preccurlyeq_1$.

2. Зададим предпорядок $\preccurlyeq_2$ на $P(K)$: ЭК (σ_2, H_2) следует за ЭК (σ_1, H_1) , если $|R_K(\sigma_1, H_1)| \leq |R_K(\sigma_2, H_2)|$. Тогда ЭК (σ, H) является наибольшей ЛЗ класса K тогда и только тогда, когда (σ, H) – максимальный элемент $P(K)$ относительно предпорядка $\preccurlyeq_2$.

3. Зададим предпорядок $\preccurlyeq_3$ на $P(K)$: ЭК (σ_2, H_2) следует за ЭК (σ_1, H_1), если $R_K(\sigma_1, H_1) \subseteq R_K(\sigma_2, H_2)$. Тогда ЭК (σ, H) является сильной ЛЗ класса K тогда и только тогда, когда (σ, H) – максимальный элемент $P(K)$ относительно предпорядка $\preccurlyeq_3$.

4. Зададим частичный порядок $\preccurlyeq_4$ на $P(K)$: ЭК (σ_2, H_2) следует за ЭК (σ_1, H_1), если $R_K(\sigma_1, H_1) \subseteq R_K(\sigma_2, H_2)$ и $H_1 \subseteq H_2$. Тогда ЭК (σ, H) порождается положительной ДСМ-гипотезой для класса K тогда и только тогда, когда (σ, H) – максимальный элемент $P(K)$ относительно порядка $\preccurlyeq_4$.

Таким образом, каждое из трёх направлений на этапе обучения ищет максимальные элементы множества $P(K)$ относительно «своего» порядка. В каждом из направлений решение возникающих задач требует значительных вычислительных ресурсов: для CVP – это задача монотонной дуализации и её обобщения, для LAD – оптимизационные задачи линейного программирования, для FCA – сложные перечислительные задачи дискретной математики. Это делает традиционные методы логической классификации слабо применимыми для задач большой размерности.

Помимо вычислительной сложности, традиционные схемы логической классификации обладают ещё одним существенным ограничением: они ориентированы исключительно на случай, когда множество значений каждого признака представляет собой конечную антицепь и для сравнения целочисленных признаковых описаний объектов используется отношение равенства. Современные прикладные задачи классификации не всегда могут быть описаны в рамках такой постановки. Вопросы модификации процедур CVP, LAD и FCA для корректного решения задачи классификации частично упорядоченных целочисленных данных общего вида рассматривались в ряде работ [29–32, 52], однако задача разработки вычислительно эффективных алгоритмов для этого случая остаётся актуальной.

Изложенные обстоятельства мотивируют разработку принципиально нового подхода к обучению логических классификаторов. Фактически, в настоящей главе предлагается новое направление логической классификации, основанное на применении методов поиска частых и нечастых элементов декартова произведения конечных частичных порядков, развитых в главе 1. Предлагаемый подход позволяет существенно сократить временные затраты на этапе обучения и распространяется на случай частично упорядоченных данных. Описание этого подхода, построенные на его основе алгоритмы и результаты теоретического и экспериментального исследования их эффективности приводятся в последующих разделах настоящей главы.

3.2. Новые модели логических классификаторов: сложность реализации и метрические свойства

В настоящем разделе предлагается и исследуется новый подход к обучению логических классификаторов, основанный на поиске в описаниях прецедентов каждого класса K часто встречающихся фрагментов специального вида, называемых правильными элементарными классификаторами. В отличие от традиционных моделей CVP , в которых на этапе обучения первоначально анализируется множество прецедентов $R(\overline{K})$ и решается задача дуализации,

предлагаемые модели сначала анализируют множество $R(K)$ и строят правильные ЭК, базируясь на методах поиска частых элементов в данных, рассмотренных в главе 1. Идея применения методов поиска часто встречающихся фрагментов в данных на этапе обучения логического классификатора впервые предложена в [15, 17, 60].

3.2.1. Логический классификатор REC (случай антицепей)

Предполагается, что множество значений каждого признака – антицепь, то есть на каждом множестве N_j , $j = 1, \dots, n$, отношение порядка не задано и для сравнения признаковых описаний объектов используется отношение равенства.

ЭК (σ, H) ранга r называется *правильным* для класса K, если не менее чем r прецедентов класса K содержат этот ЭК, то есть ЭК (σ, H) является r -частым в K.

Правильный для класса K ЭК (σ, H) называется *правильным представительным* для класса K, если ни один прецедент из $R(\overline{K})$ не содержит (σ, H), то есть (σ, H) является 1-нечастым в $\overline{K}$. Фактически, правильный представительный ЭК для класса K соответствует r-частому элементу в M при $D(M) = R(K)$ и 1-нечастому элементу при $D(M) = R(\overline{K})$.

Построение правильных представительных ЭК для каждого класса K осуществляется в два шага [18]. Сначала строятся правильные ЭК класса K . Затем из найденных ЭК отбираются те, которые не содержатся ни в одном прецеденте из $R(\overline{K})$.

Следует подчеркнуть, что предлагаемый подход осуществляет анализ в первую очередь прецедентов из $R(K)$, тогда как алгоритмы CVP, основанные на голосовании по тупиковым представительным ЭК, работают в основном с прецедентами из $R(\overline{K})$. В задачах с числом классов более двух мощность $R(K)$ обычно существенно меньше мощности $R(\overline{K})$, что делает поиск правильных ЭК менее трудоёмким по сравнению с поиском тупиковых представительных ЭК. Проверка частоты встречаемости найденного правильного ЭК в прецедентах из $R(\overline{K})$ имеет линейную от размера $R(\overline{K})$ сложность.

Обозначим через L_K матрицу, строками которой являются признаковые описания прецедентов класса K. Квадратная подматрица матрицы L_K порядка r называется *правильной*, если она состоит из одинаковых строк. Каждая правильная подматрица порядка r определяет правильный ЭК ранга r. Таким образом, задача поиска правильных ЭК может быть сведена к задаче перечисления правильных подматриц матрицы L_K . Для решения этой задачи разработан алгоритм REC, базирующийся на методах поиска частых элементов в данных [18].

Рассмотрим случай, когда к исходным данным применена процедура one-hot кодирования. При one-hot кодировании столбец матрицы L_K , соответствующий признаку x_j со значениями из $\{0, 1, \dots, k - 1\}$, заменяется группой из k бинарных столбцов, в каждом из которых единица ставится в строке, соответствующей прецеденту, у которого признак x_j принимает данное значение. Обозначим через L бинарную матрицу, полученную из L_K в результате one-hot кодирования.

Пусть Q – набор различных столбцов матрицы L , L^Q – подматрица матрицы L, образованная столбцами набора Q . Набор столбцов Q называется r -*частым*, если L^Q содержит не менее r строк, все элементы которых равны 1. Набор столбцов Q называется r -*правильным*, если он r -частый и его мощность равна r . Несложно видеть, что поиск всех правильных ЭК ранга r эквивалентен поиску всех r -правильных наборов столбцов матрицы L .

Обозначим через $R(L, r)$ множество всех столбцов матрицы L , имеющих не менее r элементов, равных 1. Пронумеруем столбцы матрицы L слева направо, начиная с 1. Пусть $e_1(R)$ и $e_2(R)$ – столбцы соответственно с наименьшим и наибольшим номерами из R , $R \subseteq R(L, r)$. Через $U_p(L)$ обозначим множество всех r -частых наборов столбцов матрицы L , мощность которых не превосходит r . Алгоритм REC строит множество всех r -правильных наборов столбцов матрицы L , перечисляя с полиномиальной задержкой наборы из $U_r(L)$.

Определим порядок, в котором происходит перечисление наборов из $U_r(L)$. На первом шаге рассматривается набор $Q = \{e_1(R(L, r))\}$.

Пусть на шаге i ($i \geq 1$) построен набор $Q \in U_r(L)$, состоящий из столбцов с номерами $j_1, \dots, j_p$, $j_1 < \dots < j_p$, $p \leq r$. Если $Q = \{e_2(R(L,r))\}$, то алгоритм заканчивает работу. Если же $Q \neq \{e_2(R(L,r))\}$, то на шаге $i + 1$ алгоритм строит новый набор ΔQ из $U_r(L)$. При этом возможны два случая: $p < r$ и $p = r$. В первом случае алгоритм строит ΔQ согласно приведённым ниже правилам 1–4. Во втором случае алгоритм строит ΔQ по правилам 2–4.

Для описания правил построения ΔQ введём обозначения: Q_t, $t = 1, \dots, p$, – набор столбцов матрицы L с номерами $j_1, \dots, j_t$; R_t, $t = 1, \dots, p$, – множество столбцов в $R(L,r)$, номера которых больше j_t; G_t, $t = 1, \dots, p$, $p < r$, – множество столбцов из R_t, каждый из которых в объединении со столбцами из Q_t образует набор из $U_r(L)$. Положим $G_p = \emptyset$ в случае $p = r$.

Заметим, что в случае $p < r$ для построения G_t в L нужно оставить только те столбцы, номера которых больше j_t и которые имеют не менее p элементов, равных 1, в подматрице, полученной после удаления из L строк, дающих 0 в пересечении со столбцами с номерами $j_1, \dots, j_t$.

Положим $Q_0 = \emptyset$ и $G_0 = \emptyset$. Перечислим возможные случаи и в каждом из них укажем правила построения ΔQ:

1) $G_p \neq \emptyset$: $\Delta Q = Q_p \cup \{e_1(G_p)\}$;

2) $G_p = \emptyset, G_{p-1} \cap R_p \neq \emptyset$: $\Delta Q = Q_{p-1} \cup \{e_1(G_{p-1} \cap R_p)\}$;

3) $G_p = \emptyset, G_{p-1} \cap R_p = \emptyset, p = 1$: $\Delta Q = \{e_1(R_p)\}$;

4) $G_p = \emptyset, G_{p-1} \cap R_p = \emptyset, p > 1$: $\Delta Q = Q_{p-2} \cup \{e_1(G_{p-2} \cap R_{p-1})\}$.

Заметим, что $R_p \neq \emptyset$ при $p = 1$, так как $Q \neq \{e_2(R(L,r))\}$, и $G_{p-2} \cap R_{p-1} \neq \emptyset$ при $p > 1$, так как столбец с номером j_p принадлежит этому множеству.

Из описания работы алгоритма видно, что в его основе лежит процесс ветвления, который удобно представить в виде обхода дерева решений в глубину. Вершинами этого дерева являются наборы из $U_r(L)$, причём r -правильные наборы столбцов находятся среди висячих вершин. Построив очередной правильный ЭК, алгоритм оценивает его корректность путём просмотра строк матрицы $L_{\overline{K}}$,

строками которой служат описания прецедентов из $\overline{K}$, то есть проверяет, является ли построенный правильный ЭК представительным для класса K. Алгоритм REC основан на модификации метода поиска (максимальных) частых элементов AD, рассмотренного в главе 1.

Для анализа сложности алгоритма REC и теоретически обоснованного выбора его параметров необходимы асимптотические оценки типичного числа и типичного ранга правильных ЭК. Впервые подобные результаты о метрических свойствах множеств ЭК были получены в работе [42], где рассматривались тестовые алгоритмы классификации. Техника получения оценок была существенно развита Е. В. Дюковой и её учениками [19–25, 27, 30, 33, 35, 58], а также А. Е. Андреевым [39] и А. А. Кибкало [37].

Введём обозначения. Пусть M_{mn}^k – множество всех матриц размера $m \times n$ с элементами из $\{0, 1, \dots, k-1\}$; $S(L_K)$, $L_K \in M_{mn}^k$, – множество правильных подматриц в матрице L_K ; $\varphi_k(m,n)$ – интервал $(0, r(k,m,n))$, где $r(k,m,n) = 0.5log_k mn - 0.5log_k log_k mn + log_k log_k log_k n$; $b_n \sim c_n$, $n \to \infty$, означает, что $lim_{n\to\infty} b_n/c_n = 1$. Рассматривается случай, когда каждый признак принимает значения из множества $\{0, 1, \dots, k-1\}, k \geq 2$.

Выявление типичной ситуации связано с высказыванием вида «для почти всех матриц L_K из M_{mn}^k при $n \to \infty$ выполнено $F_1(L_K) \sim F_2(L_K)$» , где F_1 и F_2 – два функционала, заданные на матрицах из M_{mn}^k . Данное высказывание означает, что существуют две положительные бесконечно убывающие функции $\alpha(n)$ и $\beta(n)$, такие, что для всех достаточно больших n имеет место

$$1 - |M|/|M_{mn}^k| \leq \alpha(n) ,$$

где M – множество таких матриц L_K в M_{mn}^k , для которых

$$1 - \beta(n) < F_1(L_K)/F_2(L) < 1 + \beta(n) .$$

Теорема 1. *Если* $n^a \leq m \leq k^n$, $a > 1$, *то для почти всех матриц* L_K *из* M_{mn}^k *справедливо*

$$|S(L_K)| \sim \sum_{r\in\phi_k} C_n^r C_m^r k^{r-r^2} \text{ , при } n \to \infty ,$$

и порядки почти всех правильных подматриц из $S(L_K)$ *принадлежат интервалу* ϕ_k.

Доказательство теоремы опирается на ряд приведенных ниже лемм 1 – 7.

Пусть $v \in W_r^m$, $w \in W_r^n$. Матрица $L_K \in M_{mn}^k$ называется (v, w) -правильной, если подматрица матрицы L_K , образованная строками с номерами из v и столбцами с номерами из w , является правильной. Через $M(v, w)$ обозначается множество всех (v, w) -правильных матриц из M_{mn}^k . Через $L_K[v, w]$ обозначается подматрица матрицы L_K с номерами строк из v и номерами столбцов из w .

Лемма 1. *Если* $v \in W_r^m$, $w \in W_r^n$, *то*

$$|M(v, w)| = k^{mn+r-r^2}.$$

Доказательство. Правильная подматрица $L_K[v, w]$ определяется своей первой строкой. Поэтому элементы правильной подматрицы $L_K[v, w]$ могут быть выбраны k^r способами. Остальные $mn - r^2$ элементов матрицы L_K могут быть выбраны k^{mn-r^2} способами.

Лемма 2. *Если* $v_1 \in W_r^m$, $w_1 \in W_r^n$, $v_2 \in W_l^m$, $w_2 \in W_l^n$ *и наборы* v_1 , v_2 *пересекаются по* a $(a \geq 0)$ *элементам, а наборы* w_1 , w_2 *пересекаются по* b $(b \geq 0)$ *элементам, то*

$$|M(v_1, w_1) \cap M(v_2, w_2)| = k^{mn+r-r^2+l-l^2+ab-a}.$$

Доказательство. Положим $M = M(v_1, w_1) \cap M(v_2, w_2)$. Рассмотрим матрицу $L_K \in M$. Подматрицы $L_K[v_1, w_1]$ и $L_K[v_2, w_2]$ являются правильными и содержат r^2 и l^2 элементов соответственно. Подматрицы $L_K[v_1, w_1]$ и $L_K[v_2, w_2]$ пересекаются по ab элементам. Правильные подматрицы $L_K[v_1, w_1]$ и $L_K[v_2, w_2]$ определяются своими первыми строками. Они могут быть выбраны k^{r+l-a} способами. Остальные $mn - r^2 - l^2 + ab$ элементы матрицы L_K могут быть выбраны $k^{mn-r^2-l^2+ab}$ способами.

Лемма 3. *Если* $n^a \leq m \leq k^n$, $a > 1$, *то при* $n \to \infty$ *имеет место*

$$\sum_{r=1}^{n} C_n^r C_m^r k^{r-r^2} \sim \sum_{r \in \phi_k(m,n)} C_n^r C_m^r k^{r-r^2}.$$

Доказательство. Пусть $p = 0.5 log_k mn - 0.5 log_k log_k mn$, $q = log_k log_k log_k n$, $a_r = C_n^r C_m^r k^{r-r^2}$.

Пусть $r \geq p + q - 1$, тогда

$\frac{a_{r+1}}{a_r} \leq \frac{(m-r)(n-r)}{r^2} k^{-2r} \leq \frac{mn}{p^2} k^{-2p-2q+2} \leq_n k^{-2q+2}$.

Следовательно, $\sum_{r\in[p+q,n]} a_r = o(\sum_{r\in\phi_k(m,n)} a_r)$ при $n \to \infty$.

Лемма 4. *Если* $n^a \leq m$, $a > 1$ *и* $r < log_k mn$, $l < log_k mn$, *то имеет место*

$$\sum_{b=0}^{min(r,l)} k^{lb} C_n^r C_r^b C_{n-r}^{l-b} < C_n^r C_n^l (1 + \delta(n)) ,$$

где $\delta(n) \to 0$ *при* $n \to \infty$.

Утверждение леммы совпадает с утверждением леммы 3.3.3 из работы [33] в случае $k = 2$ и доказывается аналогично.

Лемма 5 [42]. *Пусть для случайных величин* $X_1(L_K)$ *и* $X_2(L_K)$, *определенных на* M_{mn}^k, *выполнено* $X_1(L_K) \geq X_2(L_K) \geq 0$ *и при* $n \to \infty$ *верно* $\boldsymbol{M}X_1(L_K) \sim \boldsymbol{M}X_2(L_K)$, $\boldsymbol{D}X_2(L_K)/(\boldsymbol{M}X_2(L_K))^2 \to 0$. *Тогда для почти всех матриц* L_K *из* M_{mn} *имеет место* $X_1(L_K) \sim X_2(L_K) \sim \boldsymbol{M}X_2(L_K)$, $n \to \infty$.

На множестве элементарных событий $M_{mn}^k = \{L_K\}$ зададим случайную величину $\zeta_{(v,w)}(L_K)$, принимающую значение 1, если $L_K \in M(v,w)$, и значение 0 иначе. Положим

$$\zeta_1(L_K) = \sum_{r=1}^n \sum_{v\in W_r^m, w\in W_r^n} \zeta_{(v,w)}(L_K) ,$$

$$\zeta_2(L_K) = \sum_{r\in\phi_k(m,n)} \sum_{v\in W_r^m, w\in W_r^n} \zeta_{(v,w)}(L_K) .$$

Нетрудно видеть, что $|S(L_K)| = \zeta_1(L_K)$, а $\zeta_2(L_K)$ – это число таких правильных подматриц в $S(L_K)$, порядки которых принадлежат интервалу $\phi_k(m,n)$.

Через $P(\zeta_{(v,w)}(L_K) = 1)$ обозначается вероятность события $\zeta_{(v,w)}(L_K) = 1$. Очевидно, в силу леммы 1

$$P\big(\zeta_{(v,w)}(L_K) = 1\big) = \frac{|M(v,w)|}{|M_{mn}|} = \frac{k^{mn+r-r^2}}{k^{mn}} = k^{r-r^2} .$$

Лемма 6. *Если* $n^a \leq m \leq k^n$, $a > 1$, *то имеет место*

$$\boldsymbol{M}\zeta_1(L_K) \sim \boldsymbol{M}\zeta_2(L_K) \sim \sum_{r\in\phi} C_n^r C_m^r k^{r-r^2}, \text{ при } n \to \infty .$$

Доказательство. Действительно,

$$\boldsymbol{M}\zeta_1(L_K) = \sum_{r=1}^n \sum_{v\in W_r^m, w\in W_r^n} P\big(\zeta_{(v,w)}(L_K) = 1\big) = \sum_{r=1}^n C_n^r C_m^r k^{r-r^2},$$

$$\boldsymbol{M}\zeta_2(L_K) = \sum_{r\in\phi_k(m,n)} \sum_{v\in W_r^m, w\in W_r^n} P(\zeta_{(v,w)}(L_K) = 1) = \sum_{r\in\phi_k(m,n)} C_n^r C_m^r k^{r-r^2} .$$

Отсюда и из леммы 3 следует утверждение леммы 6.

Лемма 7. *Если* $n^a \le m \le k^n$, $a > 1$, *то имеет место*

$$\boldsymbol{D}\zeta_2(L_K)/(\boldsymbol{M}\zeta_2(L_K))^2 \to 0, \text{ при } n \to \infty.$$

Доказательство. Действительно,

$$\boldsymbol{D}\zeta_2(L_K) = \boldsymbol{M}(\zeta_2(L_K))^2 - (\boldsymbol{M}\zeta_2(L_K))^2.$$

Нетрудно видеть, что

$$\boldsymbol{M}(\zeta_2(L_K))^2 = \sum_{r,l\in\phi_k(m,n)} \sum_{v_1\in W_r^m, w_1\in W_r^n, v_2\in W_l^m, w_2\in W_l^n} \frac{|M|}{k^{mn}},$$

где $M = M(v_1, w_1) \cap M(v_2, w_2)$.

В силу леммы 2 и леммы 4

$$\boldsymbol{M}(\zeta_2(L_K))^2 \le \sum_{r,l\in\phi_k(m,n)} \sum_{b=0}^{\min(r,l)} C_n^r C_r^b C_{n-r}^{l-b} C_m^r C_m^l k^{r-r^2+l-l^2+lb-l} <$$

$$< \sum_{r,l\in\phi_k(m,n)} C_n^r C_n^l C_m^r C_m^l k^{r-r^2+l-l^2} (1+\delta(n)).$$

В силу леммы 6

$$(\boldsymbol{M}\zeta_2(L_K))^2 = \sum_{r,l\in\phi_k(m,n)} C_n^r C_m^r C_n^l C_m^l k^{r-r^2+l-l^2}$$

Таким образом, из лемм 6 и 7 следует, что для случайных величин $X_1 = \zeta_1(L)$ и $X_2 = \zeta_2(L)$ выполнены все условия леммы 5. Следовательно, утверждение теоремы 1 полностью доказано.

Так как правильные ЭК порождаются правильными подматрицами из $S(L)$, то приведенная в теореме 1 оценка типичного порядка правильной подматрицы косвенно свидетельствует о том, что в случае, когда число прецедентов m в $R(K)$ существенно больше числа признаков n, типичный ранг правильного представительного ЭК в $R(K)$ не превосходит $r(k, m, n)$.

Замечание. В [19] получены асимптотические оценки типичного числа правильных ЭК в $R(K)$ для двух других случаев: 1) $m_1^a \le n \le k^{m_1^\beta}$, $a > 1, \beta < 1$; 2) $n \le m_1 \le k^{n^\beta}$, $\beta < 1/2$. Показано, что типичный ранг правильного ЭК в случае 1) принадлежит интервалу $\varphi_k(m_1, n)$, а в случае 2) не превосходит $log_k m_1 + log_k log_k m_1$.

Асимптотическая оптимальность REC. Приведём описание алгоритма REC, предназначенное для работы непосредственно с целочисленными

признаковыми описаниями прецедентов без предварительной бинаризации данных и без ограничения на ранг искомых ЭК [16, 18]. Данное описание позволяет, в частности, установить асимптотическую оптимальность алгоритма REC.

Элементы $a_{i_1 j_1}$ и $a_{i_2 j_2}$ матрицы L_K назовём *совместимыми*, если $i_1 \neq i_2$, $j_1 \neq j_2$ и $a_{i_2 j_1} = a_{i_1 j_1}$, $a_{i_1 j_2} = a_{i_2 j_2}$. Набор Q из r элементов матрицы L_K называется *совместимым*, если выполнено одно из следующих двух условий: 1) $r = 1$; 2) $r \geq 2$ и любые два элемента набора Q совместимы.

Пусть $F(L_K)$ – совокупность всех совместимых наборов элементов матрицы L_K. Нетрудно видеть, что ЭК (σ, H), $\sigma = (\sigma_1, \dots, \sigma_r)$, $H = \{x_{j_1}, \dots, x_{j_r}\}$, будет правильным для K тогда и только тогда, когда в $F(L_K)$ существует набор $\{a_{i_1 j_1}, \dots, a_{i_r j_r}\}$ такой, что $a_{i_q j_q} = \sigma_q$ при $q = 1, \dots, r$.

Элементу a_{ij}, $i = 1, \dots, m_1$, $j = 1, \dots, n$, матрицы L_K присвоим номер $N[i,j] = (j-1)m_1 + i$. Пусть $R(L_K)$ – множество всех элементов матрицы L_K. Элементы с минимальным и максимальным номерами в $R \subseteq R(L_K)$ обозначим $e_1(R)$ и $e_2(R)$; при $R = R(L_K)$ положим $e_1(R) = e_1$, $e_2(R) = e_2$.

Упорядочим $F(L_K)$, указав для каждого набора Q из $F(L_K)$, $Q \neq e_2$, следующий за ним набор из $F(L_K)$, обозначаемый ΔQ.

Пусть $Q = \{a_{i_1 j_1}, \dots, a_{i_r j_r}\}$ и $N[i_{t+1}, j_{t+1}] > N[i_t, j_t]$ при $t = 1, \dots, r-1$. Положим $Q_t = \{a_{i_1 j_1}, \dots, a_{i_t j_t}\}$, $t = 1, \dots, r$. Элемент $a_{ij} \in F(L_K)$ назовём *t-допустимым*, если $(Q_t \cup \{a_{ij}\}) \in F(L_K)$ и в $R(L_K)$ не существует элемента a_{pj} такого, что $(Q_t \cup \{a_{pj}\}) \in F(L_K)$ и $p < i$.

Обозначим через R_t, $t = 1, \dots, r$, совокупность всех элементов в $R(L_K)$, номера которых больше $N[i_t, j_t]$, и через G_t, $t = 1, \dots, r$, совокупность всех t-допустимых элементов в $R(L_K)$. Положим G_0 – множество элементов столбца с номером j_1, отличных от $a_{i_1 j_1}$. Возможны следующие случаи.

Случай 1. $G_r \cap R_r \neq \emptyset$. Тогда $\Delta Q = Q \cup \{e_1(G_r \cap R_r)\}$.

Случай 2. $G_r \cap R_r = \emptyset$:

а) $r = 1$; тогда $\Delta Q = e_1(G_0 \cap R_1)$;

б) $r > 1$ и $G_{r-1} \cap R_r \neq \emptyset$; тогда $\Delta Q = \left(Q \setminus \{a_{i_r j_r}\}\right) \cup \{e_1(G_{r-1} \cap R_r)\}$;

в) $r > 1$ и $G_{r-1} \cap R_r = \emptyset$; тогда $\Delta Q = \left(Q \setminus \{a_{i_{r-1} j_{r-1}}, a_{i_r j_r}\}\right) \cup \{e_1(G_{r-2} \cap R_{r-1})\}$.

Заметим, что $G_{r-2} \cap R_{r-1} \neq \emptyset$ при $r \geq 2$, так как $a_{i_r j_r}$ принадлежит этому множеству. Если $G_r \cap R_r = \emptyset$, то набор Q называется *максимальным совместимым*.

Набор $Q = \{a_{i_1 j_1}, \dots, a_{i_r j_r}\}$ из $F(L_K)$, в котором $N[i_{t+1}, j_{t+1}] > N[i_t, j_t]$ при $t = 1, \dots, r-1$, называется *верхним* , если из того, что $a_{ij_t} = a_{i_t j_t}$, $t = 1, \dots, r$, $i \notin \{i_1, \dots, i_r\}$, следует $i > i_r$. Обозначим через $\tilde{F}(L_K)$ множество всех верхних наборов в $F(L_K)$.

Алгоритм REC строит множество всех правильных представительных ЭК класса K , перечисляя наборы из $F(L_K)$ и отбирая из них те, которые принадлежат $\tilde{F}(L_K)$. Построив на очередном шаге набор Q из $\tilde{F}(L_K)$, алгоритм проверяет корректность правильного ЭК (σ, H) , порождаемого набором Q , то есть проверяет, является ли построенный ЭК представительным для класса K . Проверка корректности осуществляется путём просмотра строк матрицы $L_{\overline{K}}$.

Таким образом, алгоритм REC строит множество всех правильных представительных ЭК класса K за $|F(L_K)|$ шагов, где $|F(L_K)|$ – мощность $F(L_K)$. Работу алгоритма можно представить в виде обхода дерева решений в глубину: корнем служит пустой набор; вершины – правильные ЭК класса K , которые либо являются правильными представительными ЭК класса K , найденными впервые, либо соответствуют лишним шагам. Висячие вершины порождаются максимальными совместимыми наборами.

В [23] доказаны теоремы, свидетельствующие об асимптотическом равенстве числа совместимых наборов из $F(L_K)$ и числа правильных представительных ЭК класса K . Это означает, что число лишних шагов алгоритма REC (на которых строятся правильные ЭК, не являющиеся представительными) имеет меньший порядок роста по сравнению с числом всех правильных ЭК. Таким образом,

алгоритм REC является асимптотически оптимальным. Полиномиальная временная оценка одного шага алгоритма равна $O(m_1^2 n)$, проверка представительности ЭК выполняется за время $O(m_2 n)$.

Экспериментальное исследование. Опишем модели корректных логических классификаторов, основанные на поиске правильных представительных ЭК [18]. Пусть $r \in \{1, 2, \dots, m_1\}$ – настраиваемый параметр, задающий минимальную частоту встречаемости ЭК в прецедентах класса K . Введём следующие обозначения.

Обозначим через $T_0(r, K)$ множество всех тупиковых r -представительных ЭК класса K , то есть множество всех таких тупиковых представительных ЭК класса K , которые являются r -частыми в K . Через $T_1(r, K)$ обозначим множество всех тупиковых r -представительных ЭК класса K , имеющих ранг r . Через $T_2(r, K)$ обозначим множество всех правильных r -представительных ЭК класса K . Через $T_3(r, K)$ обозначим множество тупиковых r -представительных ЭК класса K , порождаемых элементами из $T_2(r, K)$.

Рассматриваются четыре модели корректного голосования A_0 , A_1 , A_2 и A_3 . Модель A_0 – это классическая модель CVP: на этапе обучения строится множество $T_0(r, K)$ путём решения задачи дуализации с последующим отбором r -частых в K ЭК. Модель A_1 действует по схеме модели A_0 , но в голосовании участвуют только те тупиковые r -представительные ЭК, которые имеют ранг r , то есть на этапе обучения строится множество $T_1(r, K)$.

Модели A_2 и A_3 действуют по принципиально иной схеме. Первоначально анализируются описания прецедентов класса K и строятся правильные ЭК ранга r. Затем просматриваются прецеденты из других классов и в модели A_2 из найденных ЭК отбираются представительные, то есть строится множество $T_2(r, K)$, а в модели A_3 отбираются тупиковые представительные ЭК, то есть строится $T_3(r, K)$.

Отметим, что модель A_2 фактически представляет собой классификатор на основе алгоритма REC, описанного выше.

Таким образом, на этапе обучения модели A_0 и A_1 решают задачу монотонной дуализации (анализ множества $R(\overline{K})$), а модели A_2 и A_3 осуществляют поиск правильных ЭК (анализ множества $R(K)$). Процедура вычисления оценки принадлежности распознаваемого объекта S классу K во всех четырёх моделях одинакова. Пусть $P^i_{(\sigma,H)}$, $(\sigma,H) \in T_i(r,K)$, $i \in \{0,1,2,3\}$, – число прецедентов из $R(K)$, содержащих (σ,H) . Оценка принадлежности объекта S классу K имеет вид

$$\Gamma_i(S,K) = \frac{1}{|T_i(r,K)|} \sum_{(\sigma,H) \in T_i(r,K)} P^i_{(\sigma,H)} \cdot \Omega(\sigma,H,S) .$$

Объект S относится к классу с наибольшей оценкой. Если таких классов несколько, то объект относится к классу с наибольшим числом прецедентов.

Из описания моделей видно, что время работы моделей A_0 и A_1 существенно зависит от мощности множества $R(\overline{K})$, поскольку на первом этапе обучения решается задача дуализации данных из $R(\overline{K})$. При этом зависимость от параметра r невелика, так как проверка частоты встречаемости найденного минимального нечастого ЭК в $R(K)$ имеет линейную по m_1 сложность. Время работы моделей A_2 и A_3 , напротив, существенно зависит от параметра r , поскольку при увеличении r число правильных ЭК в $R(K)$ быстро убывает. По сравнению с моделями A_0 и A_1 , время работы моделей A_2 и A_3 в меньшей степени зависит от m_2 , так как проверка нечастости найденного правильного ЭК в $R(\overline{K})$ имеет линейную по m_2 сложность.

Экспериментальное исследование на реальных данных. Задачи взяты из репозитория UCI [72] и репозитория ВЦ ФИЦ ИУ РАН. Размеры рассмотренных задач приведены в таблице 3, где для каждой задачи указано число прецедентов в классах m_1 и m_2 , число признаков n и средняя значность признака $\bar{k}$ (среднее число допустимых значений одного признака).

При реализации классификаторов к исходным данным применялась процедура one-hot кодирования. Алгоритмы A_1 , A_2 и A_3 реализованы на языке программирования C++. В тестировании на качество классификации также участвовали такие хорошо известные алгоритмы машинного обучения, как случайный лес (RF) и логистическая регрессия (LR). Дополнительная настройка алгоритмов RF и LR не производилась.

Таблица 3 – Размеры задач

Задача	$\boldsymbol{m_1}$; $\boldsymbol{m_2}$	$\boldsymbol{n}$	$\boldsymbol{\bar{k}}$
Манелис	38; 107	35	10
Остеосаркома	50; 217	19	25
Крестики-нолики (UCI)	626; 332	9	3
Инсульт	16; 63	81	2
Шахматы (UCI)	1527; 1668	36	2
Мол. биол. 1 (UCI)	767; 768	60	5
Задача 5	1051; 1005	34	7
Мол. биол. 2 (UCI)	1540; 1650	62	5

Результаты счёта усреднялись по 10 случайным независимым разбиениям прецедентов, 80% которых использовалось для обучения моделей, а 20% – для оценки качества классификации. В каждом из разбиений распределение прецедентов по классам сохранялось неизменным. Функционалом качества выбрана сбалансированная точность классификации, вычисляемая по формуле

$$\psi = \frac{1}{l}\sum_{i=1}^{l} q_i ,$$

где q_i – доля верно классифицированных объектов класса K_i . Данный функционал хорошо себя зарекомендовал при несбалансированных классах. В случае равномощных классов сбалансированная точность совпадает с долей верно классифицированных объектов. Результаты счёта на реальных целочисленных задачах приведены в таблице 4. Для каждой задачи в таблице 4 указан ранг r_i , $i \in \{1, 2, \dots, l\}$, голосующих ЭК класса K_i . Время работы алгоритмов указано в миллисекундах.

Таблица 4 – Время обучения и качество классификации на реальных задачах

Задача	$(\boldsymbol{r_1}, \dots, \boldsymbol{r_l})$	**Время A_1, мс**	**Время A_2, мс**	**Время A_3, мс**	$\boldsymbol{A_1, A_3}$	$\boldsymbol{A_2}$ **(REC)**	**RF**	**LR**
Манелис	(3, 3)	512,1	**47,0**	48,6	0,691	0,735	0,742	**0,774**
Остеосаркома	(3, 4)	289,2	**18,3**	18,4	0,560	0,570	0,545	**0,578**
Крестики-нолики (UCI)	(3, 3)	71,7	1,0	1,0	**0,976**	**0,976**	0,939	0,639
Инсульт	(2, 3)	238,4	**140,0**	150,0	0,614	**0,623**	0,542	0,553
Шахматы (UCI)	(4, 4)	5 294	**903,7**	1 062	0,903	0,974	**0,988**	0,956
Мол. биол. 1 (UCI)	(5, 5)	2 763 106	**59 265**	69 387	0,960	**0,971**	0,960	0,922
Задача 5	(4, 4, 4)	35 471	**8,3**	9,4	0,641	0,770	**0,905**	0,790
Мол. биол. 2 (UCI)	(5, 5, 5)	10 487 213	**235 045**	315 275	0,793	0,794	**0,946**	0,831

Как видно из таблицы 4, модель A_2 превосходит по качеству и времени работы модели A_1 и A_3 на всех рассмотренных данных, кроме задачи Крестики-нолики, и в среднем не уступает по качеству ни случайному лесу, ни логистической регрессии. На трёх задачах (Крестики-нолики, Инсульт, Молекулярная биология 1) модель A_2 превосходит все модели.

Модель A_1 работает существенно медленнее модели A_3 при том, что оба алгоритма строят множество всех тупиковых r -представительных ЭК ранга r . Однако модель A_1 на первом этапе обучения ищет в $R(\overline{K})$ тупиковые ЭК ранга r , а модель A_3 перечисляет правильные ЭК ранга r для K . Стоит отметить, что на шести задачах (Манелис, Остеосаркома, Крестики-нолики, Инсульт, Шахматы, Задача 5) модели A_2 и A_3 обучались менее чем за 1 с, что свидетельствует об их высокой вычислительной эффективности. На задаче Молекулярная биология 1 модель A_1 обучалась более 46 минут, тогда как A_2 – менее одной минуты.

Рассмотрим результаты подробнее. На задаче Молекулярная биология 1, содержащей 1532 прецедента и 284 признака после one-hot кодирования, модель A_1 обучалась более 46 минут (2 763 106 мс), тогда как модель A_2 – менее одной минуты (59 265 мс), то есть примерно в 47 раз быстрее. При этом качество модели A_2 (0,971) выше качества модели A_1 (0,960). Аналогичная ситуация наблюдается на задаче Молекулярная биология 2, где модель A_1 обучалась более 2,9 часа, тогда как модель A_2 – менее 4 минут.

На задачах небольшой размерности (Крестики-нолики, Инсульт, Манелис) модели A_2 и A_3 обучаются менее чем за 0,2 секунды. На задаче Шахматы, несмотря на значительное число прецедентов (3195), модели A_2 и A_3 обучаются менее чем за 1,1 секунды, тогда как модель A_1 – более 5 секунд.

Как видно из результатов счёта, алгоритм REC в среднем сопоставим по качеству классификации алгоритмам RF и LR. Классификатор REC демонстрирует наилучший результат по качеству классификации на трёх задачах (Крестики-нолики, Инсульт, Молекулярная биология 1), RF – на трёх задачах (Молекулярная биология 2, Шахматы, Задача 5), LR – на двух задачах (Манелис, Остеосаркома).

Экспериментально установлено также, что время обучения модели A_2 существенно зависит от параметра r . При увеличении r число правильных ЭК быстро убывает, что приводит к значительному сокращению времени работы алгоритма. Данное наблюдение подтверждает теоретический вывод о том, что типичный ранг правильного ЭК не превосходит величины $r(k, m_1, n)$: при r близком к $r(k, m_1, n)$, алгоритм работает наиболее эффективно как по времени, так и по качеству классификации.

Замечание. В экспериментах ранг r_i , $i \in \{1, 2, \dots, l\}$, голосующих ЭК класса K_i брался равным числу $0.5log_2 m_i n_1 - 0.5log_2 log_2 m_i n_1 - log_2 log_2 log_2 n_1$, где m_i – число прецедентов класса K_i . Данный выбор мотивирован оценкой типичного порядка правильной подматрицы, приведённой в Теореме 1.

Эксперименты на модельных данных. На рисунке 8 и рисунке 9 приведено время обучения моделей A_1 (CVP) и A_2 (REC) на случайных модельных данных из равномерного распределения при $l = 2$, $k = 2$, m_1 – число прецедентов в каждом классе, n_1 – число признаков. Результаты счёта усреднены по 20 независимым запускам. Время работы алгоритмов указано в секундах. Время счёта модели A_3 не приводится на графиках, так как в рассматриваемых примерах оно практически совпадает с временем работы A_2 .

На рисунке 8 показан экспоненциальный рост временных затрат на этапе обучения классификаторов A_1 и A_2 при $m_1 = 250$ в зависимости от числа признаков n_1 . Видно, что при относительно небольшом n_1 разрыв во времени счёта для A_1 и A_2 незначителен. При $n_1 \geq 150$ алгоритм A_1 работает значительно медленнее алгоритма A_2 . Например, A_1 обучается примерно в 1,3 раза медленнее A_2 при $n_1 = 150$, а при $n_1 = 250$ – в 1,7 раз медленнее.

На рисунке 9 продемонстрирован линейный рост временных затрат на этапе обучения классификаторов A_1 и A_2 при $n_1 = 100$ в зависимости от числа прецедентов m_1 . Видно, что время работы A_1 растёт быстрее по сравнению с временем работы A_2 . Например, A_1 обучается примерно в 1,2 раза медленнее A_2 при $m_1 = 100$ и почти в 2 раза медленнее при $m_1 = 700$.

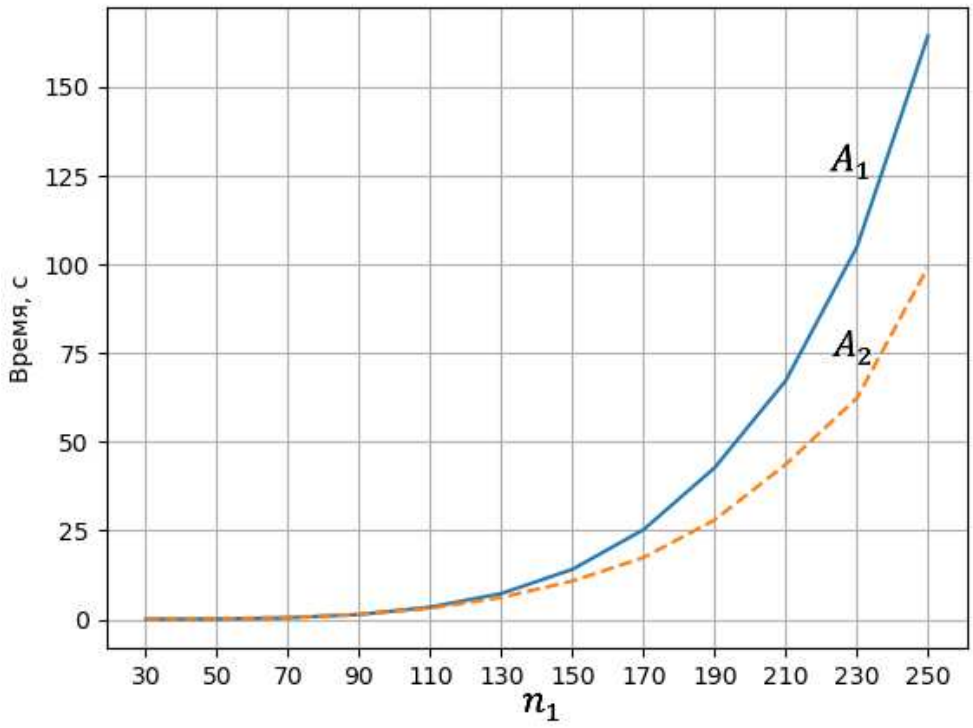

Рисунок 8 – Зависимость времени обучения моделей A_1 и A_2 от числа признаков при $m_1 = 250$.

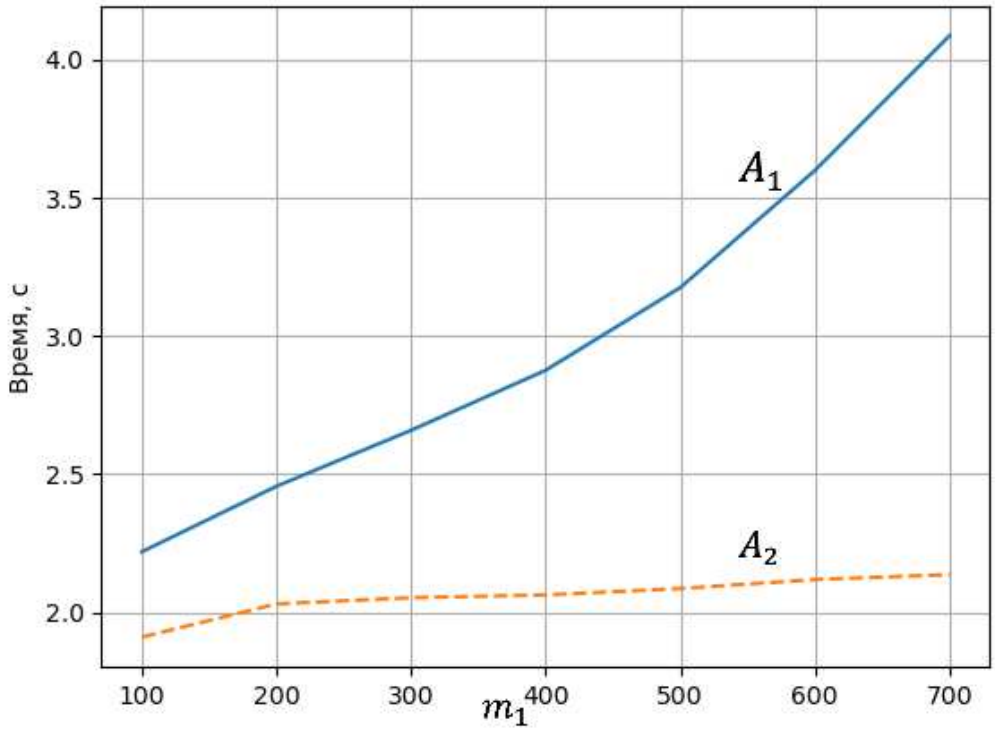

Рисунок 9 – Зависимость времени обучения моделей A_1 и A_2 от числа прецедентов при $n_1 = 100$.

Результаты экспериментального исследования свидетельствуют о целесообразности предлагаемого подхода к построению логических классификаторов. Модель A_2 (REC), основанная на поиске правильных представительных ЭК, является наиболее эффективной среди рассмотренных моделей: она обучается существенно быстрее классической модели A_1 (CVP), основанной на решении задачи монотонной дуализации, и при этом не уступает ей по качеству классификации. Сравнение с такими хорошо известными алгоритмами машинного обучения, как случайный лес и логистическая регрессия, показывает, что предложенные алгоритмы в среднем не уступают по качеству классификации указанным методам, а на ряде задач превосходят их.

Следует также отметить, что алгоритм REC хорошо себя зарекомендовал при решении важной задачи генной инженерии – задачи предсказания промотеров [24]. С использованием модельного организма *Drosophila melanogaster* решалась задача бинарной классификации, в которой положительными примерами являлись участки промотеров, а отрицательные примеры представляли собой участки экзонов. На несбалансированной выборке большого объёма проведены эксперименты, в которых помимо REC участвовали такие известные алгоритмы, как случайный лес, логистическая регрессия и различные модели градиентного бустинга. Была рассмотрена оригинальная методика прямого применения классификатора к исходным символьным последовательностям промотеров и экзонов, позволяющая работать с целочисленными признаками небольшой значности. Показано, что в этом случае качество логической классификации существенно выше и составляет с использованием ансамблевого подхода 94,3% по показателю ROC-AUC, при этом логический классификатор REC незначительно уступил только классификатору CatBoost [24].

3.2.2. Логический классификатор REC+ (случай цепей)

В настоящем разделе рассматривается случай, когда на множествах значений признаков заданы линейные порядки, то есть указанные множества – конечные цепи. Предлагается классификатор REC+, работающий с частично

упорядоченными данными. Задание частичных порядков на множествах значений признаков является естественным расширением модели REC. Алгоритм REC+ работает по схеме оригинального алгоритма. Единственное отличие состоит в определении совместимости двух элементов матрицы L_K : знак равенства заменяется на знак предшествования «$\preccurlyeq$». А именно, элементы $a_{i_1 j_1}$ и $a_{i_2 j_2}$ матрицы L_K называются *совместимыми*, если $i_1 \neq i_2$, $j_1 \neq j_2$ и $a_{i_2 j_1} \preccurlyeq a_{i_1 j_1}$, $a_{i_1 j_2} \preccurlyeq a_{i_2 j_2}$.

Работу алгоритма REC+ можно представить в виде обхода дерева решений в глубину, аналогично алгоритму REC. Все определения и построения из раздела 3.2 (совместимый набор, верхний набор, максимальный совместимый набор, нумерация элементов, правила построения ΔQ) переносятся на случай REC+ дословно с указанной заменой знака равенства на знак предшествования.

Для применения алгоритма REC+ необходимо задать линейные порядки на множествах значений признаков. В ряде прикладных задач такие порядки известны заранее из содержательных соображений. Однако в общем случае линейные порядки необходимо строить на основе анализа обучающих данных. Приведём описание используемой в экспериментах процедуры линейного упорядочения значений признаков [32, 33]. Пусть $R_1(K)$ и $R_2(K)$ – множества прецедентов из класса K и не из K соответственно, $|R_t(K)|$ – мощность $R_t(K)$, $t = 1, 2$; $S = (a_1, \dots, a_n)$ – объект из M , $a \in N_j$, $j = 1, \dots, n$. Положим

$B_j(S, a) = 1$, если $a_j = a$, иначе $B_j(S, a) = 0$;

$$\mu_j^{(t)}(a) = \frac{1}{|R_t(K)|} \sum_{S \in R_t(K)} B_j(S, a) , t = 1, 2 ;$$

$$\mu_j(a) = \mu_j^{(1)}(a) - \mu_j^{(2)}(a) .$$

Величина $\mu_j(a)$, $a \in N_j$, $j = 1, \dots, n$, служит оценкой информативности значения a признака x_j в классе K и позволяет установить на множестве значений признака x_j , встречающихся в описаниях прецедентов из K , линейный порядок, согласно которому $a \preccurlyeq b$, $b \in N_j$, если $\mu_j(a) \leq \mu_j(b)$.

Содержательный смысл величины $\mu_j(a)$ состоит в следующем. Значение $\mu_j^{(1)}(a)$ представляет собой относительную частоту встречаемости значения a признака x_j в прецедентах класса K, а $\mu_j^{(2)}(a)$ – относительную частоту встречаемости этого значения в прецедентах из других классов. Таким образом, положительные значения $\mu_j(a)$ указывают на то, что значение a признака x_j более характерно для класса K, а отрицательные – для других классов.

В экспериментальном исследовании модели REC+ признаковое описание объекта дублируется, причём на дублированных признаках устанавливается обратный линейный порядок. Дублирование необходимо для обеспечения несравнимости описаний прецедентов из разных классов, что является обязательным условием существования представительных ЭК. Следует отметить, что дублирование приводит к увеличению числа признаков в два раза, что существенно влияет на время работы алгоритма. Ищутся правильные представительные ЭК заданного ранга r. Параметр r выбирается с использованием верхних оценок типичного ранга правильного ЭК, приведённых далее в теоремах 2 и 3.

Метрические свойства множества правильных ЭК для случая цепей. Приводимые ниже теоретические оценки получены в предположении, что $N_j = \{0, 1, \dots, k-1\}$, $k \geq 2$, $j = 1, \dots, n$, и элементы в N_j линейно упорядочены в порядке возрастания.

Пусть $\mathfrak{A}(L_K)$, где $L_K \in M_{m_1 n}^k$, – это множество всех правильных ЭК в $R(K)$; $d = k/(k-1)$; $\sigma \in E_{k-1}^r$, $\sigma = (\sigma_1, \dots, \sigma_r)$; $Q_r(\sigma) = (\sigma_1 + 1)^r \times \dots \times (\sigma_r + 1)^r$; $r_2 =]\log_d m_1 + \log_d \log_d m_1[$, здесь $]q[$ – наименьшее целое, превосходящее q; $r_3 =]0.5\log_d m_1 n - 0.5 \log_d \log_d m_1 n + \log_d \log_d log_d n[$; φ_2 – интервал $[1, r_2]$, φ_3 – интервал $[1, r_3]$; $b_n \lesssim c_n$, $n \to \infty$, означает, что $lim_{n\to\infty} b_n/c_n \leq 1$.

Теорема 2. Если $n \leq m_1$, то для почти всех матриц L_K из $M_{m_1 n}^k$ верно

$$|\mathfrak{A}(L_K)| \lesssim \sum_{r\in\varphi_2} \sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2}, \text{ при } n \to \infty,$$

и ранги почти всех правильных ЭК из $\mathfrak{A}(L_K)$ принадлежат интервалу φ_2.

Пусть $v = (i_1, \dots, i_r) \in W_r^{m_1}, w = (j_1, \dots, j_r) \in W_r^n$, $\sigma \in E_{k-1}^r$. Матрица $L_K = (a_{ij}) \in M_{m_1 n}^k$ называется (v, w, σ) -допустимой, если $a_{i_t j_t} \le \sigma_t$, $t = \overline{1, r}$. Через $M(v, w, \sigma)$ обозначается множество всех (v, w, σ) -допустимых матриц из $M_{m_1 n}^k$. Через $L_K[v, w]$ обозначается подматрица матрицы L_K с номерами строк из v и номерами столбцов из w .

Лемма 8. *Если* $v \in W_r^{m_1}, w \in W_r^n, \sigma \in E_{k-1}^r$, *то*

$$|M(v, w, \sigma)| = Q_r(\sigma)\, k^{m_1 n - r^2}.$$

Доказательство. (v, w, σ) -допустимая подматрица $L_K[v, w]$ может быть определена $Q_r(\sigma)$ способами. Остальные $m_1 n - r^2$ элементов матрицы L_K могут быть выбраны $k^{m_1 n - r^2}$ способами.

Лемма 9. *Если* $v_1 \in W_r^{m_1}, w_1 \in W_r^n, v_2 \in W_l^{m_1}, w_2 \in W_l^n$, $\sigma' \in E_{k-1}^r$, $\sigma'' \in E_{k-1}^l$ *и наборы* v_1 , v_2 *пересекаются по* a $(a \ge 0)$ *элементам, а наборы* w_1 , w_2 *пересекаются по* b $(b \ge 0)$ *элементам, то*

$$|M(v_1, w_1, \sigma') \cap M(v_2, w_2, \sigma'')| \le Q_r(\sigma')\, Q_r(\sigma'')\, k^{m_1 n + ab - r^2 - l^2}.$$

Доказательство. Элементы матрицы L_K , расположенные на пересечении строк с номерами из v_1 и столбцов с номерами из w_1 , а также те её элементы, которые расположены на пересечении строк с номерами из v_2 и столбцов с номерами из w_2 могут быть выбраны не более, чем $Q_r(\sigma')\, Q_r(\sigma'')$ способами. Остальные элементы могут быть выбраны $k^{m_1 n + ab - r^2 - l^2}$ способами.

Лемма 10. Если $n \le m_1$, *то при* $n \to \infty$ *справедливо*

$$\sum_{r=1}^{m_1} \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2} \sim \sum_{r \in \phi_2} \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2} .$$

Доказательство. Пусть $a_r = \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2}$, $r > \log_d m_1 + \log_d \log_d m_1 - 1$. Оценим отношение a_{r+1}/a_r . Рассмотрим некоторый фиксированный набор $\sigma \in E_{k-1}^r$ и набор $\sigma' \in E_{k-1}^{r+1}$ такой, что $\sigma \subset \sigma'$. Тогда $Q_{r+1}(\sigma') = Q_r(\sigma) \times (\sigma_1 + 1) \times \dots \times (\sigma_r + 1) \times ({\sigma'}_{r+1} + 1)^{r+1}$.

Следовательно, $\frac{Q_{r+1}(\sigma')}{Q_r(\sigma)} \le (k-1)^{2r+1}$.

Так как каждому набору σ длины r соответствует $(k-1)$ набор σ' длины $r+1$, то

$\frac{\sum_{\sigma' \in E_{k-1}^{r+1}} Q_{r+1}(\sigma')}{\sum_{\sigma \in E_{k-1}^{r}} Q_r(\sigma)} \le (k-1)^{2r+2}$.

Поэтому $\frac{a_{r+1}}{a_r} \le \frac{(m_1-r)(n-r)}{(r+1)^2} k^{-2r-1}(k-1)^{2r+2} \le \frac{m_1 n}{(\log_d m_1 + \log_d \log_d m_1)^2} d^{-2r-1}(k-1) \le \frac{(k-1)m_1 n}{(\log_d m_1 + \log_d \log_d m_1)^2} d^{-2\log_d m_1 + 1} \le \frac{k}{(\log_d m_1 + \log_d \log_d m_1)^2}$.

Таким образом, $a_{r+1} = o(a_r)$ при $r \ge r_2 - 1$, $n \to \infty$. Следовательно, $\sum_{r \ge r_2} a_r \sim a_{r_2}$, $n \to \infty$.

Лемма 11. *Если* $n \le m_1$, $r \in \varphi_2$, $l \in \varphi_2$, *то имеет место*

$$\sum_{b=0}^{min(r,l)} k^{lb} C_n^r C_r^b C_{n-r}^{l-b} < C_n^r C_n^l (1 + \delta(n)) ,$$

где $\delta(n) \to 0$ *при* $n \to \infty$.

Утверждение леммы доказывается аналогично утверждению леммы 3.3.3 из работы [33].

На множестве элементарных событий $M_{m_1 n}^k = \{L_K\}$ зададим случайную величину $\zeta_{(v,w,\sigma)}(L_K)$, принимающую значение 1, если $L_K \in M(v,w,\sigma)$, и значение 0 иначе. Положим

$$\zeta_1(L_K) = \sum_{r=1}^{n} \sum_{v \in W_r^{m_1}, w \in W_r^n} \sum_{\sigma \in E_{k-1}^r} \zeta_{(v,w,\sigma)}(L_K) ,$$

$$\zeta_2(L_K) = \sum_{r \in \phi_2} \sum_{v \in W_r^{m_1}, w \in W_r^n} \sum_{\sigma \in E_{k-1}^r} \zeta_{(v,w,\sigma)}(L_K) .$$

Через $P(\zeta_{(v,w,\sigma)}(L_K) = 1)$ обозначается вероятность события $\zeta_{(v,w,\sigma)}(L_K) = 1$. Через $\boldsymbol{M}\zeta_1(L_K)$ и $\boldsymbol{D}\zeta_1(L_K)$ обозначаются соответственно математическое ожидание и дисперсия случайной величины $\zeta_1(L_K)$.

Очевидно, в силу леммы 8

$$P\left(\zeta_{(v,w,\sigma)}(L_K) = 1\right) = \frac{|M(v,w,\sigma)|}{|M_{m_1 n}^k|} = \frac{Q_r(\sigma)\, k^{m_1 n - r^2}}{k^{m_1 n}} = Q_r(\sigma) k^{-r^2} .$$

Лемма 12. *Если* $n \le m_1$, *то имеет место*

$$\boldsymbol{M}\zeta_1(L_K) \sim \boldsymbol{M}\zeta_2(L_K) \sim \sum_{r \in \varphi_2} \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2}, \text{ при } n \to \infty .$$

Доказательство. Действительно,

$\boldsymbol{M}\zeta_1(L_K) = \sum_{r=1}^{n} \sum_{v \in W_r^m, w \in W_r^n} \sum_{\sigma \in E_{k-1}^r} P\left(\zeta_{(v,w,\sigma)}(L_K) = 1\right) =$

$\sum_{r=1}^{n} \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2}$,

$\boldsymbol{M}\zeta_2(L_K) = \sum_{r\in\phi_2}\sum_{v\in W_r^m, w\in W_r^n}\sum_{\sigma\in E_{k-1}^r} P\left(\zeta_{(v,w,\sigma)}(L_K)=1\right) =$

$\sum_{r\in\varphi_2}\sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2}$.

Отсюда и из леммы 10 следует утверждение леммы 12.

Лемма 13. *Если* $n \le m_1$, *то имеет место*

$$\boldsymbol{D}\zeta_2(L_K)/(\boldsymbol{M}\zeta_2(L_K))^2 \to 0 \text{ , при } n \to \infty .$$

Доказательство. Действительно,

$\boldsymbol{D}\zeta_2(L_K) = \boldsymbol{M}(\zeta_2(L_K))^2 - (\boldsymbol{M}\zeta_2(L_K))^2$.

Нетрудно видеть, что

$\boldsymbol{M}(\zeta_2(L_K))^2 = \sum_{r,l\in\phi_2}\sum_{v_1\in W_r^m, w_1\in W_r^n, v_2\in W_l^m, w_2\in W_l^n}\sum_{\sigma'\in E_{k-1}^r,\sigma''\in E_{k-1}^l}\frac{|M|}{k^{m_1 n}}$,

где $M = M(v_1, w_1, \sigma') \cap M(v_2, w_2, \sigma'')$.

В силу леммы 9 и леммы 11

$\boldsymbol{M}(\zeta_2(L_K))^2 \le$

$\le \sum_{r,l\in\phi_2}\sum_{\sigma'\in E_{k-1}^r,\sigma''\in E_{k-1}^l} Q_r(\sigma')\, Q_r(\sigma'')\, \sum_{b=0}^{\min(r,l)} C_n^r C_r^b C_{n-r}^{l-b} C_{m_1}^r C_{m_1}^l k^{lb-r^2-l^2} <$

$< \sum_{r,l\in\phi_2}\sum_{\sigma'\in E_{k-1}^r,\sigma''\in E_{k-1}^l} Q_r(\sigma')\, Q_r(\sigma'')\ C_n^r C_n^l C_{m_1}^r C_{m_1}^l k^{-r^2-l^2}(1+\delta(n))$.

В силу леммы 12

$(\boldsymbol{M}\zeta_2(L_K))^2 = \sum_{r,l\in\phi_2}\sum_{\sigma'\in E_{k-1}^r,\sigma''\in E_{k-1}^l} Q_r(\sigma')\, Q_r(\sigma'') C_n^r C_n^l C_{m_1}^r C_{m_1}^l k^{-r^2-l^2}$.

Таким образом, из лемм 12 и 13 следует, что для случайных величин $X_1 = \zeta_1(L_K)$ и $X_2 = \zeta_2(L_K)$ выполнены все условия леммы 5. Следовательно, в силу того что $|\mathfrak{A}(L_K)| \le \boldsymbol{M}\zeta_1(L_K)$, утверждение Теоремы 2 полностью доказано.

Теорема 2 даёт верхнюю асимптотическую оценку типичного числа правильных ЭК для случая, когда число прецедентов не меньше числа признаков.

Теорема 3. Если $m_1^\alpha \le n \le d^{m_1}$, $\alpha > 1$, то для почти всех матриц L_K из $M_{m_1 n}^k$ верно

$$|\mathfrak{A}(L_K)| \sim \sum_{r\in\varphi_3}\sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2} \text{ , при } n \to \infty ,$$

и ранги почти всех правильных ЭК из $\mathfrak{A}(L_K)$ принадлежат интервалу φ_3 .

Пусть $v = (i_1, \dots, i_r) \in W_r^{m_1}, w = (j_1, \dots, j_r) \in W_r^n$, $\sigma \in E_{k-1}^r$. Обозначим через $M^*(v, w, \sigma)$ множество всех матриц из $M(v, w, \sigma)$, для которых из условия $v_1 \in W_r^{m_1}$, $v_1 \neq v$, следует, что матрица не является (v_1, w, σ) -матрицей.

Лемма 14. *Если* $v \in W_r^{m_1}$, $w \in W_r^n$, $\sigma \in E_{k-1}^r$, *то*

$$|M^*(v, w, \sigma)| \geq Q_r(\sigma)(1 - d^{-r})^{m_1 - r} k^{m_1 n - r^2}.$$

Доказательство. (v, w, σ) -допустимая подматрица $L_K[v, w]$ может быть определена $Q_r(\sigma)$ способами. Элементы из столбцов, не принадлежащих w , могут быть выбраны $k^{m_1(n-r)}$ способами. Остальные элементы могут быть выбраны не менее, чем $(k^r - (k-1)^r)^{m_1 - r}$ способами. Имеем, что $|M^*(v, w, \sigma)| \geq Q_r(\sigma) k^{m_1(n-r)} (k^r - (k-1)^r)^{m_1 - r} = Q_r(\sigma)(1 - d^{-r})^{m_1 - r} k^{m_1 n - r^2}$.

Лемма 15. Если $m_1^\alpha \leq n \leq d^{m_1}$, $\alpha > 1$, *то при* $n \to \infty$ *справедливо*

$$\sum_{r=1}^{m_1} \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2} \sim \sum_{r \in \phi_3} \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2} .$$

Лемма 16. *Если* $m_1^\alpha \leq n \leq d^{m_1}$, $\alpha > 1$, $r \in \varphi_3$, $l \in \varphi_3$, *то имеет место*

$$\sum_{b=0}^{min(r,l)} k^{lb} C_n^r C_r^b C_{n-r}^{l-b} < C_n^r C_n^l (1 + \delta(n)) ,$$

где $\delta(n) \to 0$ *при* $n \to \infty$.

Утверждения лемм 15 и 16 доказываются аналогично утверждениям лемм 10 и 11 соответственно.

Пусть $w \in W_r^n$, $w = \{x_{j_1}, \dots, x_{j_r}\}$, $\sigma \in E_{k-1}^r$. На множестве элементарных событий $M_{m_1 n}^k = \{L_K\}$ зададим случайную величину $\xi_{(w,\sigma)}(L_K)$, принимающую значение 1, если ЭК (σ, H) , $H = \{x_{j_1}, \dots, x_{j_r}\}$, принадлежит $\mathfrak{A}(L_K)$. Положим

$$\xi_1(L_K) = \sum_{r=1}^{m_1} \sum_{w \in W_r^n} \sum_{\sigma \in E_{k-1}^r} \xi_{(w,\sigma)}(L_K) ,$$

$$\xi_2(L_K) = \sum_{r \in \phi_3} \sum_{w \in W_r^n} \sum_{\sigma \in E_{k-1}^r} \xi_{(w,\sigma)}(L_K) .$$

Нетрудно видеть, что $\xi_1(L_K) = |\mathfrak{A}(L_K)|$, а $\xi_2(L_K)$ совпадает с числом таких ЭК из $\mathfrak{A}(L_K)$, ранги которых принадлежат интервалу ϕ_3 .

Лемма 17. *Если* $m_1^\alpha \leq n \leq d^{m_1}$, $\alpha > 1$, то при $r \in \varphi_3$ *имеет место*

$$\boldsymbol{M}\xi_1(L_K) \sim \boldsymbol{M}\xi_2(L_K) \sim \sum_{r \in \varphi_3} \sum_{\sigma \in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2}, \text{ при } n \to \infty .$$

Доказательство. Оценим $\boldsymbol{M}\xi_1(L_K)$ и $\boldsymbol{M}\xi_2(L_K)$. Очевидно, что

$\boldsymbol{M}\xi_1(L_K) \geq \boldsymbol{M}\xi_2(L_K) = \sum_{r\in\phi_3} \sum_{w\in W_r^n} \sum_{\sigma\in E_{k-1}^r} P(\xi_{(w,\sigma)}(L_K) = 1)$.

В силу леммы 14,

$\boldsymbol{M}\xi_2(L_K) \geq \sum_{r\in\phi_3} \sum_{v\in W_r^{m_1}} \sum_{\sigma\in E_{k-1}^r} C_n^r \frac{|M^*(v,w,\sigma)|}{k^{m_1 n}} \geq$

$\geq \sum_{r\in\phi_3} \sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r (1-d^{-r})^{m_1-r} k^{-r^2} \geq$

$\geq \sum_{r\in\phi_3} \sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r (1-2m_1 d^{-r})^{m_1-r} k^{-r^2} \geq$

$\geq \sum_{r\in\phi_3} \sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2} F(n)$, *где* $F(n) \to 1$ при $n \to \infty$.

В силу леммы 8 и леммы 15,

$\boldsymbol{M}\xi_1(L_K) \leq \sum_{r=1}^{m_1} \sum_{v\in W_r^{m_1}} \sum_{\sigma\in E_{k-1}^r} C_n^r \frac{|M(v,w,\sigma)|}{k^{m_1 n}} =$

$= \sum_{r=1}^{m_1} \sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2} \sim \sum_{r\in\phi_3} \sum_{\sigma\in E_{k-1}^r} Q_r(\sigma) C_n^r C_{m_1}^r k^{-r^2}$.

Таким образом, исходя из приведенных выше оценок следует утверждение леммы 17.

Лемма 18. *Если* $m_1^\alpha \leq n \leq d^{m_1}$, $\alpha > 1$, *то имеет место*

$$\boldsymbol{D}\xi_2(L_K)/(\boldsymbol{M}\xi_2(L_K))^2 \to 0 \text{ , при } n \to \infty .$$

Доказательство утверждения леммы 18 аналогично доказательству утверждения леммы 13.

Таким образом, из лемм 17 и 18 следует, что для случайных величин $X_1 = \xi_1(L_K)$ и $X_2 = \xi_2(L_K)$ выполнены все условия леммы 5. Следовательно, утверждение Теоремы 3 полностью доказано.

Теорема 3 даёт оценку типичного числа правильных ЭК для случая, когда число прецедентов меньше числа признаков.

В [29] показано, что в случае задания линейных порядков на множествах значений признаков на этапе обучения процедур CVP возникает задача дуализации над произведением частичных порядков, то есть требуется перечислить все минимальные 1-нечастые ЭК в $\overline{K}$. Обозначим множество всех минимальных 1-нечастых ЭК через $B(L_{\overline{K}})$. Пусть $r_4 =]0.5\, log_d\, m_2 n - 0.5\, log_d\, log_d\, m_2 n + log_d\, log_d\, log_d\, n[$; φ_4 – интервал $[1, r_4]$. В [30] доказана следующая

Теорема 4. *Если* $m_2^{\alpha} \leq n \leq d^{m_2}$, $\alpha > 1$, *то для почти всех матриц* $L_{\overline{K}}$ *из* $M_{m_2 n}^{k}$ *при* $n \to \infty$ *верно*

$$|B(L_{\overline{K}})| \sim \sum_{r \in \varphi_4} \sum_{\sigma \in E_{k-1}^{r}} Q_r(\sigma)\, C_n^r\, C_{m_2}^r\, r!\; k^{-r^2},$$

и ранги почти всех ЭК из $B(L_{\overline{K}})$ *принадлежат интервалу* φ_4.

Таким образом, если $m_1 \leq m_2$, то согласно оценкам, приведённым в Теоремах 3 и 4, число правильных ЭК в K существенно меньше числа минимальных нечастых ЭК в $\overline{K}$ благодаря наличию множителя $r!$ в оценке для $|B(L_{\overline{K}})|$. При $m_1 = m_2$ типичный ранг каждого из двух видов ЭК принадлежит одному интервалу.

Приведённые теоретические оценки свидетельствуют о том, что предлагаемый подход к классификации, основанный на поиске правильных ЭК, обладает существенным преимуществом по сравнению с традиционными процедурами CVP и в случае задания линейных порядков на множествах значений признаков. Число искомых ЭК растёт значительно медленнее, что позволяет ожидать более быструю работу алгоритма REC+ по сравнению с классическими алгоритмами дуализации.

Экспериментальное сравнение REC и REC+. На реальных задачах проведено экспериментальное сравнение качества работы алгоритмов REC, REC+, случайного леса (RF) и логистической регрессии (LR). Алгоритмы REC и REC+ реализованы на языке C++. Алгоритмы RF и LR импортированы из библиотеки scikit-learn. Дополнительная настройка алгоритмов не проводилась. Рассмотрены шесть задач с двумя классами, размеры которых приведены в таблице 3. Итоговая оценка качества классификации получена усреднением значения функционала сбалансированной точности по 10 независимым запускам. В каждом запуске исходные данные случайным образом разделялись на обучающую и тестовую выборки в соотношении 4 : 1. Результаты счёта приведены в таблице 5.

Нетрудно видеть, что алгоритм REC+ лидирует по качеству классификации на четырёх задачах из шести (Остеосаркома, Инсульт, Шахматы, Крестики-нолики). Классификатор REC+ превосходит по качеству работы классификатор REC на пяти

задачах из шести, однако работает существенно медленнее REC из-за необходимого увеличения числа признаков в два раза (для корректности работы алгоритма признаковые описания объектов дублируются с обратным отношением порядка). На каждой из рассмотренных задач время работы REC не превышает 1 секунды.

Таблица 5 – Качество классификации на реальных задачах

Задача	**REC**	**REC+**	**RF**	**LR**
Остеосаркома	0,57	**0,60**	0,55	0,58
Инсульт	0,62	**0,70**	0,54	0,55
Манелис	0,74	0,75	0,74	**0,77**
Мол. биол. 1 (UCI)	**0,97**	**0,97**	0,96	0,92
Шахматы (UCI)	0,97	0,98	**0,99**	0,96
Крестики-нолики (UCI)	0,98	**0,99**	0,94	0,64

Наиболее значительное преимущество REC+ наблюдается на задаче Инсульт, где качество классификации повысилось с 0,62 (REC) до 0,70 (REC+), при этом оба логических классификатора существенно превосходят как случайный лес (0,54), так и логистическую регрессию (0,55). На задаче Крестики-нолики алгоритм REC+ достигает качества 0,99, превосходя RF (0,94) и значительно превосходя LR (0,64). Единственная задача, на которой REC+ не улучшает результат REC – Молекулярная биология 1, где оба алгоритма показывают одинаковый результат 0,97.

Результаты экспериментального исследования свидетельствуют о том, что использование линейных порядков на множествах значений признаков позволяет повысить качество логической классификации. При этом процедура линейного упорядочения значений признаков, основанная на оценке информативности значений признаков, демонстрирует практическую эффективность на всех рассмотренных задачах.

3.3. Основные результаты

В настоящей главе предложено и исследовано новое направление логической классификации, основанное на поиске в описаниях прецедентов каждого класса K часто встречающихся фрагментов специального вида – правильных

представительных элементарных классификаторов. Построение правильных представительных ЭК осуществляется в два этапа: сначала строятся правильные ЭК на основе анализа множества прецедентов $R(K)$, а затем из них отбираются представительные путём проверки частоты встречаемости в $R(\overline{K})$.

Для случая декартова произведения антицепей разработан алгоритм REC перечисления правильных представительных ЭК. Алгоритм имеет полиномиальную временную оценку шага. Алгоритм REC осуществляет поиск правильных представительных ЭК класса K путём анализа, в первую очередь, прецедентов из K , тогда как алгоритмы CVP, основанные на голосовании по тупиковым представительным ЭК, работают в основном с прецедентами из $\overline{K}$. В задачах с числом классов более двух мощность $R(\overline{K})$ обычно существенно больше мощности $R(K)$, что делает поиск правильных ЭК менее трудоёмким по сравнению с поиском тупиковых ЭК.

Получена асимптотическая оценка типичного числа и типичного порядка правильных подматриц целочисленной матрицы L_K для случая, когда число прецедентов существенно больше числа признаков (Теорема 1). Данная оценка является верхней асимптотической оценкой числа правильных представительных ЭК. Показано, что типичный ранг правильного представительного ЭК не превосходит величины $r(k, m_1, n)$, что позволяет теоретически обоснованно выбирать ранг искомых ЭК во время экспериментального исследования.

Проведено экспериментальное сравнение алгоритма REC с алгоритмом голосования по тупиковым представительным ЭК из направления CVP. Показано, что REC работает существенно быстрее традиционного алгоритма из направления CVP и, как правило, превосходит его по качеству классификации. Сравнение с такими известными алгоритмами, как случайный лес и логистическая регрессия, показало, что REC в среднем не уступает по качеству указанным методам, а на ряде задач превосходит их.

Разработана модификация алгоритма REC для работы с данными, представленными в виде декартова произведения конечных цепей, названная

REC+. Получены теоретические оценки типичного числа и типичного ранга правильных ЭК для случая цепей (Теоремы 2 и 3). Проведено сравнение с оценкой числа минимальных 1-нечастых ЭК (Теорема 4), подтверждающее, что число правильных ЭК существенно меньше числа минимальных 1-нечастых (тупиковых).

Экспериментальное сравнение алгоритмов REC и REC+ на шести реальных задачах показало, что использование линейных порядков на множествах значений признаков позволяет повысить качество классификации. Классификатор REC+ превосходит по качеству классификатор REC на пяти задачах из шести. Полученное повышение качества достигается без привлечения дополнительной информации – линейные порядки строятся исключительно на основе анализа обучающей выборки.

Заключение

В настоящей диссертационной работе разработаны вычислительно эффективные методы логического анализа информации, представленной в виде декартова произведения конечных частично упорядоченных множеств. Сформулируем основные результаты, полученные в работе.

1. Предложен последовательно-совместный подход к перечислению множеств X_{max} максимальных частых и Y_{min} минимальных нечастых элементов декартова произведения частичных порядков. Доказана корректность предлагаемого метода (Утверждения 1–3, раздел 1.2). Экспериментально показано, что последовательно-совместный алгоритм наиболее эффективен в случае, когда мощности множеств частых и нечастых элементов примерно равны.

2. На базе последовательно-совместного подхода к поиску X_{max} и Y_{min} построен алгоритм асимптотически оптимальной расшифровки двузначной монотонной функции, заданной на декартовом произведении частично упорядоченных множеств. Доказана его корректность (Утверждения 4–6, раздел 2.2). Экспериментально показано, что в случае задания рассматриваемой функции на множестве E_k^n , предложенный алгоритм в среднем работает быстрее классического алгоритма из [1] и требует меньшего числа обращений к оракулу при больших n .

3. На основе поиска частых элементов в данных разработаны новые быстрые алгоритмы логической классификации, базирующиеся на перечислении корректных ЭК специального вида, названных правильными представительными. Для случая декартова произведения антицепей построен алгоритм REC перечисления правильных представительных ЭК. Для случая декартова произведения цепей разработана его модификация REC+. Получены асимптотические оценки типичного числа и типичного ранга правильных ЭК для случаев антицепей (Теорема 1, раздел 3.2.1) и цепей (Теоремы 2 и 3, раздел 3.2.2). Полученные оценки свидетельствуют о том, что число правильных представительных ЭК существенно меньше числа тупиковых представительных

ЭК, поиск которых осуществляется в направлении CVP (Теорема 4, раздел 3.2.2). Экспериментально подтверждено, что алгоритм REC обучается существенно быстрее классической модели из CVP и не уступает ей по качеству классификации. Алгоритм REC+ превосходит алгоритм REC по качеству классификации на большинстве рассмотренных задач, что свидетельствует о целесообразности задания линейных порядков на множестве значений признаков.

Таким образом, в диссертационной работе разработаны вычислительно эффективные методы логического анализа частично упорядоченных данных. Указанные методы применены к задачам перечисления максимальных частых и минимальных нечастых элементов декартова произведения частичных порядков, расшифровки двузначных монотонных логических функций и классификации по прецедентам. Предложенные методы обоснованы теоретически и подтверждены экспериментально. Результаты работы опубликованы в 11 научных работах, из которых 5 статей опубликованы в журналах, рекомендованных ВАК, 2 статьи – в журналах, индексируемых в Web of Science Core Collection, и 5 работ – в изданиях, индексируемых в Scopus.

Дальнейшие исследования предполагается направить на развитие методов логического анализа частично упорядоченных данных и расширение области их практического применения.

Список литературы

[1] Алексеев, В. Б. О расшифровке некоторых классов монотонных многозначных функций / В. Б. Алексеев // Ж. вычисл. матем. и матем. физ. – 1976. – Т. 16, № 1. – С. 189–198.

[2] Ансель, Ж. О числе монотонных булевых функций n переменных / Ж. Ансель // Кибернетич. сб. Нов. сер. – 1968. – Вып. 5. – С. 53–57.

[3] Баскакова, Л. В. Модель распознающих алгоритмов с представительными наборами и системами опорных множеств / Л. В. Баскакова, Ю. И. Журавлёв // Ж. вычисл. матем. и матем. физ. – 1981. – Т. 21, № 5. – С. 1264–1275.

[4] Бонгард, М. М. Использование обучающейся программы для выявления нефтеносных пластов / М. М. Бонгард, М. Н. Вайнцвайг, Ш. А. Губерман, М. Л. Извекова, М. С. Смирнов // Геология и геофизика. – 1966. – № 6. – С. 96–105.

[5] Бонгард, М. М. Проблема узнавания / М. М. Бонгард. – М.: Физматгиз, 1967. – 321 с.

[6] Вайнцвайг, М. Н. Алгоритм обучения распознаванию образов «Кора» / М. Н. Вайнцвайг // Алгоритмы обучения распознаванию образов / под ред. В. Н. Вапника. – М.: Советское радио, 1973. – С. 110–116.

[7] Генрихов, И. Е. Классификация на основе полных решающих деревьев / И. Е. Генрихов, Е. В. Дюкова // Ж. вычисл. матем. и матем. физ. – 2012. – Т. 52, № 4. – С. 750–761.

[8] Генрихов, И. Е. Поиск частых элементов произведения частичных порядков с использованием параллельных вычислений / И. Е. Генрихов, Е. В. Дюкова // Автоматика и телемеханика. – 2021. – № 10. – С. 13–24.

[9] Драгунов, Н. А. О метрических (количественных) свойствах логических классификаторов / Н. А. Драгунов // Труды ИСА РАН. – 2024. – Т. 74, № 4. – С. 14–19.

[10] Драгунов, Н. А. Поиск минимальных нечастых и максимальных частых наборов в частично упорядоченных данных / Н. А. Драгунов, Е. В. Дюкова //

Математические методы распознавания образов: тезисы докл. 19-й Всеросс. конф. с междунар. участием. – 2019. – С. 10–12.

[11] Драгунов, Н. А. Асимптотически оптимальная расшифровка двузначной монотонной функции / Н. А. Драгунов, Е. В. Дюкова // Математические методы распознавания образов: тезисы докл. 20-й Всеросс. конф. с междунар. участием. – 2021. – С. 38–40.

[12] Драгунов, Н. А. Поиск частых и нечастых элементов произведения частичных порядков и задача расшифровки двузначной монотонной функции / Н. А. Драгунов, Е. В. Дюкова // Информационные технологии и нанотехнологии: сб. тр. VII Междунар. конф. и молодёж. школы. – 2021. – С. 031852.

[13] Драгунов, Н. А. О поиске максимальных частых и минимальных нечастых наборов произведения частичных порядков / Н. А. Драгунов, Е. В. Дюкова // Информатика и её применения. – 2022. – Т. 16, вып. 1. – С. 82–87.

[14] Драгунов, Н. А. Об одном подходе к расшифровке монотонной логической функции / Н. А. Драгунов, Е. В. Дюкова // Автоматика и телемеханика. – 2022. – Вып. 10. – С. 134–143.

[15] Драгунов, Н. А. Поиск частых элементов в данных и обучение по прецедентам / Н. А. Драгунов, Е. В. Дюкова // Интеллектуализация обработки информации: тезисы докл. 14-й Междунар. конф. – 2022. – С. 47–49.

[16] Драгунов, Н. А. Правильные представительные элементарные классификаторы над произведением частичных порядков / Н. А. Драгунов, Е. В. Дюкова // Информатика и её применения. – 2025. – Т. 19, вып. 4. – С. 43–52.

[17] Драгунов, Н. А. Классификация по прецедентам и поиск в данных частых элементов / Н. А. Драгунов, Е. В. Дюкова, А. П. Дюкова // Информационные технологии и нанотехнологии: сб. тр. VIII Междунар. конф. и молодёж. школы. – 2022. – С. 050712.

[18] Драгунов, Н. А. Логическая классификация на основе поиска правильных представительных элементарных классификаторов / Н. А. Драгунов, Е. В. Дюкова, А. П. Дюкова // Известия РАН. Теория и системы управления. – 2024. – № 4. – С. 86–92.

[19] Дюкова, А. П. О числе решений некоторых специальных задач логического анализа целочисленных данных / А. П. Дюкова, Е. В. Дюкова // Известия РАН. Теория и системы управления. – 2023. – № 5. – С. 57–66.

[20] Дюкова, Е. В. Об асимптотически оптимальном алгоритме построения тупиковых тестов / Е. В. Дюкова // ДАН СССР. – 1977. – Т. 233, № 4. – С. 527–530.

[21] Дюкова, Е. В. О сложности реализации некоторых процедур распознавания / Е. В. Дюкова // Ж. вычисл. матем. и матем. физ. – 1987. – Т. 27, № 1. – С. 114–127.

[22] Дюкова, Е. В. Асимптотические оценки некоторых характеристик множества представительных наборов и задача об устойчивости / Е. В. Дюкова // Ж. вычисл. матем. и матем. физ. – 1995. – Т. 35, № 1. – С. 123–134.

[23] Дюкова, Е. В. О сложности реализации логических процедур классификации по прецедентам / Е. В. Дюкова // Ж. вычисл. матем. и матем. физ. – 2025. – Т. 65, № 8. – С. 2025–2034.

[24] Дюкова, Е. В. О методах машинного обучения в задаче предсказания промотеров / Е. В. Дюкова, А. П. Дюкова // Докл. Междунар. конф. «Математическая биология и биоинформатика». – 2024. – Т. 10, № e42. – С. 1–4.

[25] Дюкова, Е. В. Дискретный анализ признаковых описаний в задачах распознавания большой размерности / Е. В. Дюкова, Ю. И. Журавлёв // Ж. вычисл. матем. и матем. физ. – 2000. – Т. 40, № 8. – С. 1264–1278.

[26] Дюкова, Е. В. Задача монотонной дуализации и её обобщения: асимптотические оценки числа решений / Е. В. Дюкова, Ю. И. Журавлёв // Ж. вычисл. матем. и матем. физ. – 2018. – Т. 58, № 12. – С. 2153–2168.

[27] Дюкова, Е. В. Об асимптотически оптимальном построении тупиковых покрытий целочисленной матрицы / Е. В. Дюкова, С. А. Инякин // Математические вопросы кибернетики. – 2008. – № 17. – С. 247–262.

[28] Дюкова, Е. В. Логические методы корректной классификации данных / Е. В. Дюкова, Г. О. Масляков, А. П. Дюкова // Информатика и её применения. – 2023. – Т. 17, вып. 3. – С. 64–70.

[29] Дюкова, Е. В. О дуализации над произведением частичных порядков / Е. В. Дюкова, Г. О. Масляков, П. А. Прокофьев // Машинное обучение и анализ данных. – 2017. – Т. 3, № 4. – С. 239–249.

[30] Дюкова, Е. В. Дуализация над произведением цепей: асимптотические оценки числа решений / Е. В. Дюкова, Г. О. Масляков, П. А. Прокофьев // ДАН. – 2018. – Т. 483, № 2. – С. 130–133.

[31] Дюкова, Е. В. О логическом анализе данных с частичными порядками в задаче классификации по прецедентам / Е. В. Дюкова, Г. О. Масляков, П. А. Прокофьев // Ж. вычисл. матем. и матем. физ. – 2019. – Т. 59, № 9. – С. 1605–1616.

[32] Дюкова, Е. В. Корректная классификация по прецедентам: ДСМ-метод над произведением частичных порядков / Е. В. Дюкова, Г. О. Масляков, Д. С. Янаков // Информатика и её применения. – 2024. – Т. 18, вып. 3. – С. 61–68.

[33] Дюкова, Е. В. Поиск информативных фрагментов описаний объектов в дискретных процедурах распознавания / Е. В. Дюкова, Н. В. Песков // Ж. вычисл. матем. и матем. физ. – 2002. – Т. 42, № 5. – С. 741–753.

[34] Дюкова, Е. В. Об асимптотически оптимальных алгоритмах дуализации / Е. В. Дюкова, П. А. Прокофьев // Ж. вычисл. матем. и матем. физ. – 2015. – Т. 55, № 5. – С. 895–910.

[35] Дюкова, Е. В. О сложности задачи дуализации / Е. В. Дюкова, Р. М. Сотнезов // Ж. вычисл. матем. и матем. физ. – 2012. – Т. 52, № 10. – С. 1926–1935.

[36] Журавлёв, Ю. И. Распознавание. Математические методы. Программная система. Практические применения / Ю. И. Журавлёв, В. В. Рязанов, О. В. Сенько. – М.: ФАЗИС, 2006. – 159 с.

[37] Кибкало, А. А. О Т-алгоритмах распознавания, использующих короткие тесты : дис. … канд. физ.-мат. наук / А. А. Кибкало. – М.: МГУ, 1988.

[38] Коробков, В. К. О монотонных функциях алгебры логики / В. К. Коробков // Проблемы кибернетики. – Вып. 13. – М.: Наука, 1965. – С. 5–28.

[39] Кудрявцев, В. Б. Теория тестового распознавания / В. Б. Кудрявцев, А. Е. Андреев // Интеллектуальные системы. – 2006. – Т. 10. – С. 95–140.

[40] Кузнецов, С. О. Об одной модели обучения и классификации, основанной на операции сходства / С. О. Кузнецов, В. К. Финн // Обозрение прикладной и промышленной математики. – 1996. – Т. 3, № 1. – С. 66–90.

[41] Масич, И. С. Метод оптимальных логических решающих правил для задач распознавания и прогнозирования / И. С. Масич // Системы управления и информационные технологии. – 2019. – Т. 75, № 1. – С. 31–37.

[42] Носков, В. Н. О числе тупиковых тестов для одного класса таблиц / В. Н. Носков, В. А. Слепян // Кибернетика. – 1972. – № 1. – С. 60–65.

[43] Финн, В. К. О возможности формализации правдоподобных рассуждений средствами многозначных логик / В. К. Финн // Всесоюзн. симп. по логике и методологии науки. – 1976. – С. 82–83.

[44] Финн, В. К. Базы данных с неполной информацией и новый метод автоматического порождения гипотез / В. К. Финн // В кн.: Диалоговые и фактографические системы информационного обеспечения. – М., 1981. – С. 153–156.

[45] Хачиян, Л. Г. Избранные труды / Л. Г. Хачиян. – М.: МЦНМО, 2009. – 519 с.

[46] Яблонский, С. В. О тестах для электрических схем / С. В. Яблонский, И. А. Чегис // Усп. матем. наук. – 1955. – Т. 10, вып. 4(66). – С. 182–184.

[47] Agrawal, R. Mining association rules between sets of items in large databases / R. Agrawal, T. Imielinski, A. Swami // Proc. ACM SIGMOD Int. Conf. on Management of Data. – 1993. – P. 207–216.

[48] Agrawal, R. Fast algorithms for mining association rules in large databases / R. Agrawal, R. Srikant // Proc. 20th Int. Conf. Very Large Data Bases. – 1994. – P. 487–499.

[49] Agrawal, R. Mining generalized association rules / R. Agrawal, R. Srikant // Future Gener. Computer Syst. – 1997. – Vol. 13, iss. 2–3. – P. 161–180.

[50] Agarwal, R. Depth-first Generation of Long Patterns / R. Agarwal, C. C. Aggarwal, V. V. V. Prasad // Proc. 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining. – 2000. – P. 108 – 118.

[51] Aggarwal, C. Frequent Pattern Mining / C. Aggarwal. – Springer Int. Publ., 2014. – 471 p.

[52] Anisimova, D. V. Supervised classification problem: searching for maximum patterns / D. V. Anisimova, E. V. Djukova, A. P. Djukova // Proc. X Int. Conf. Information Technology and Nanotechnology (ITNT-2024). – 2024. – P. 1–4.

[53] Bonates, T. O. Maximum patterns in datasets / T. O. Bonates, P. L. Hammer, A. Kogan // Discrete Appl. Math. – 2008. – Vol. 156, № 6. – P. 846–861.

[54] Boros, E. An efficient incremental algorithm for generating all maximal independent sets in hypergraphs of bounded dimension / E. Boros, V. Gurvich, K. Elbassioni, L. Khachiyan // Parallel Process. Lett. – 2000. – Vol. 10, № 4. – P. 253–266.

[55] Burdick, D. MAFIA: a maximal frequent itemset algorithm for transactional databases / D. Burdick, M. Calimlim, J. Gehrke // Proc. Int. Conf. Data Engineering. – 2001. – P. 443–452.

[56] Crama, Y. Cause-effect relationships and partially defined Boolean functions / Y. Crama, P. L. Hammer, T. Ibaraki // Ann. Oper. Res. – 1988. – Vol. 16, № 1. – P. 299–325.

[57] Djukova, E. V. On an asymptotically optimal algorithm for constructing irredundant tests / E. V. Djukova // Dokl. Akad. Nauk SSSR. – 1977. – Vol. 233, № 4. – P. 423–426.

[58] Djukova, E. V. The Complexity of Transformation of Normal Forms for Characteristic Functions of Classes / E. V. Djukova, V. Yu. Nefedov // Pattern Recognition and Image Analysis. – 2009. – Vol. 19, № 3. – P. 435–440.

[59] Dragunov, N. A. One approach to monotone logical function decoding / N. A. Dragunov, E. V. Djukova // Automation and Remote Control. – 2022. – № 10. – P. 1600–1607.

[60] Dragunov, N. A. Supervised classification and finding frequent elements in data / N. A. Dragunov, E. V. Djukova, A. P. Djukova // Proc. 8th Int. Conf. Information Technology and Nanotechnology. – 2022. – P. 5.

[61] Dragunov, N. A. Logical classification based on finding regular representative elementary classifiers / N. A. Dragunov, E. V. Djukova, A. P. Djukova // J. Comput. Syst. Sci. Int. – 2024. – Vol. 63. – P. 634–641.

[62] Elbassioni, K. Algorithms for dualization over products of partially ordered sets / K. Elbassioni // SIAM J. Discrete Math. – 2009. – Vol. 23, № 1. – P. 487–510.

[63] Elbassioni, K. On finding minimal infrequent elements in multidimensional data defined over partially ordered sets / K. Elbassioni // arXiv preprint arXiv:1411.2275. – 2014.

[64] Fredman, M. L. On the complexity of dualization of monotone disjunctive normal forms / M. L. Fredman, L. Khachiyan // J. Algorithms. – 1996. – Vol. 21. – P. 618–628.

[65] Gnatyshak, D. V. Triadic formal concept analysis and triclustering: searching for optimal patterns / D. V. Gnatyshak, D. I. Ignatov, S. O. Kuznetsov // Machine Learning. – 2015. – Vol. 101. – P. 271–302.

[66] Hammer, P. L. Partially defined Boolean functions and cause-effect relationships / P. L. Hammer // Lecture at the Int. Conf. Multi-Attribute Decision-Making via OR-Based Expert Systems. – Passau, 1986.

[67] Han, J. Mining frequent patterns without candidate generation / J. Han, J. Pei, Y. Yin // Proc. ACM SIGMOD Int. Conf. on Management of Data. – 2000. – P. 1–12.

[68] Ignatov, D. I. Introduction to Formal Concept Analysis and Its Applications in Information Retrieval and Related Fields / D. I. Ignatov // arXiv preprint arXiv:1703.02819. – 2017.

[69] Johnson, D. S. On generating all maximal independent sets / D. S. Johnson, M. Yannakakis, C. H. Papadimitriou // Inform. Process. Lett. – 1988. – Vol. 27, № 3. – P. 119–123.

[70] Kovshov, N. V. Algorithms for finding logical regularities in pattern recognition / N. V. Kovshov, V. L. Moiseev, V. V. Ryazanov // Comput. Math. Math. Phys. – 2008. – Vol. 48, № 2. – P. 314–328.

[71] Murakami, K. Efficient algorithms for dualizing large-scale hypergraphs / K. Murakami, T. Uno // Discrete Applied Mathematics. – 2014. – Vol. 170. – P. 83–94.

[72] UC Irvine Machine Learning Repository [Electronic resource]. – Access mode: https://archive.ics.uci.edu/

[73] Wille, R. Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts / R. Wille // In: Rival I. (ed.) Ordered Sets. – NATO Adv. Study Inst. Ser. Springer Netherlands, 1982. – Vol. 83. – P. 445–470.

[74] Zaki, M. J. Efficiently mining maximal frequent itemsets / M. J. Zaki // Proc. Int. Conf. Data Mining. – 2001. – P. 163–170.

[75] Zaki, M. J. CHARM: an efficient algorithm for closed itemset mining / M. J. Zaki, C.-J. Hsiao // Proc. SIAM Int. Conf. Data Mining. – 2002. – P. 457–473.