ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
УЧРЕЖДЕНИЕ НАУКИ
МАТЕМАТИЧЕСКИЙ ИНСТИТУТ им. В. А. СТЕКЛОВА
РОССИЙСКОЙ АКАДЕМИИ НАУК

На правах рукописи
УДК 519.766.23

Вишникин Максим Евгеньевич

Категориальные грамматики с ограничением на количество присваиваемых типов

Специальность 1.1.5 — математическая логика, алгебра,
теория чисел и дискретная математика

ДИССЕРТАЦИЯ
на соискание учёной степени кандидата физико-математических наук

Научный руководитель:
доктор физико-математических наук
Кузнецов Степан Львович

МОСКВА — 2025

Содержание

Введение **3**

Глава первая. Иерархия классов языков **22**

1.1 AB-грамматики 22

1.2 Базовые категориальные грамматики 25

1.3 Преобразование БКГ в контекстно-свободную грамматику 27

1.4 Иерархия классов языков с присвоением k типов 29

Глава вторая. Инварианты и сведение посредством гомоморфизмов **37**

2.1 Свойства редукций 37

2.2 Незамкнутость класса $\mathcal{G}^1$ 42

2.3 Сведение к AB-грамматикам с однозначным присвоением типов посредством гомоморфизмов 44

2.4 Сведение к грамматикам из класса $\mathcal{G}^2$ посредством гомоморфизмов 71

Глава третья. Алгоритмические свойства **74**

3.1 Грамматики класса $\mathcal{G}^2$ 74

3.2 Грамматики класса $\mathcal{G}^1$ 75

3.3 AB-грамматики с однозначным присвоением типов 82

Заключение **85**

Литература **87**

Введение

Общая характеристика работы

Актуальность темы и степень её разработанности

Настоящая диссертация посвящена теории формальных языков, относящейся к пересечению математической логики и теории алгоритмов, теоретической информатики и алгебры. Сегодня эта теория представляет собой обширный раздел математики, в котором формируется целый спектр формализмов, восходящих к лингвистическим моделям.

Важнейшим источником для теории формальных языков является теория порождающих грамматик Н. Хомского [1]. В этой теории был предложен подход к описанию синтаксических структур естественного языка в виде формального исчисления, порождающего цепочки в некотором алфавите. Символами (буквами) алфавита являются слова естественного языка; используются также дополнительные символы, среди которых могут быть синтаксические типы, например: «группа подлежащего». С помощью такого формализма можно также естественно описывать синтаксис языков математической логики и языков программирования [2]. В диссертации мы используем термин «слово» как синонимичный термину «цепочка символов» в данном (конечном) алфавите. Хомским была предложена иерархия порождающих грамматик и, соответственно, порождаемых ими языков. Важнейшим является класс контекстно-свободных грамматик и порождаемых ими языков.

Другой подход, который можно охарактеризовать как «логический», заключается в том, чтобы приписать элементам алфавита Σ некоторые «типы» — формулы специального построенного логического языка. Предложение же считают «корректно заданным», если существует такой выбор типов для каждого отдельного слова, что в некотором исчислении выводится импликация из данной последовательности типов в фиксированный результирующий тип. Такое

описание синтаксиса хорошо согласуется с определением семантики по Монтегю [3], где «смысл» выражения определяется, исходя из семантики его частей (см. также [4]). Общее название для класса подобного рода формализмов — «категориальные грамматики».

Исторически первым вариантом категориальных грамматик были «AB-грамматики» (в честь К. Айдукевича и Й. Бар-Хиллела), восходящие к работе Айдукевича [5] 1935 года. В своей современной форме данное понятие появляется в работе Бар-Хиллела [6] 1953 года. В этих грамматиках типы построены из произвольного множества «примитивных типов» посредством двух операций — левого и правого деления. Позднее в работе [7] доказаны основные свойства формализма, наиболее интересным из которых является равенство класса порождаемых языков классу контекстно-свободных языков без пустого слова. Данный формализм играет ключевую роль, так как большинство других категориальных грамматик являются его расширением при помощи добавления новых операций и правил.

Приведём пример, как категориальные грамматики могут задавать множества корректных предложений. Рассмотрим фразу: «Иван встречает Марию». Синтаксическим единицам «Иван» и «Мария» присваиваются типы именной группы np. Глаголу «встречает» присваивается тип $(np\backslash(S/np))$, где S — тип правильно построенного предложения. Таким образом, для слов в данной фразе существует такое присвоение типов $np; (np\backslash(S/np)); np$, что выводится соответствующая импликация: $np; (np\backslash(S/np)); np \Rightarrow S$.

Дополнительные возможности в описании синтаксиса естественного языка дают категориальные грамматики, предложенные И. Ламбеком [8] [9]. Этот вид грамматик построен на основе AB-грамматик за счёт добавления правил введения делений справа и дополнительной операции — умножения. М.Р. Пентусом [10] доказано равенство класса языков, порождаемых категориальными грамматиками Ламбека, классу контекстно-свободных языков без пустого слова. Среди расширений AB-грамматик рассматривались также категориальные

грамматики зависимостей (CDG), комбина́торные категориальные грамматики (CCG) и конъюнктивные категориальные грамматики. Введенные М.И. Дехтярём совместно с А.Я. Диковским [11] [12], а впоследствии исследованные в работах [13] [14], CDG получены из AB-грамматик при помощи добавления валентности у типов. Комбина́торные категориальные грамматики, изучавшиеся М. Стидманом [15], расширяют AB-грамматики добавлением новых правил преобразования строк типов. Конъюнктивные категориальные грамматики, введённые в статье [16], расширяют AB-грамматики за счёт добавления конъюнкций в знаменателях типов.

В теории формальных языков важную роль играют алгоритмические и алгебраические свойства различных классов грамматик. Классические алгебраические свойства — совпадения классов языков для различных формализмов, замкнутость относительно естественных операций. К основным алгоритмическим задачам относятся: проверка включения языков, проверка эквивалентности, задача синтаксического разбора и многие другие.

Известно множество работ, посвящённых различным алгоритмическим задачам для контекстно-свободных грамматик. Для задачи синтаксического анализа, то есть проверки принадлежности слова некоторому контекстно-свободному языку, И. Сакаи [17] доказана разрешимость за полиномиальное время (обычно алгоритм называют алгоритмом Кока — Янгера — Касами, по более поздним работам). Для класса контекстно-свободных грамматик, для которых у каждого слова существует одно «левое» древо разбора, А.Л. Семёновым [18] доказана теорема о разрешимости равенства автоматному языку. Однако многие задачи алгоритмически неразрешимы, что доказывается в работах Ш. Грейбах [19], А.В. Гладкого [20], Бар-Хиллела [21] и других авторов. В настоящее время эти результаты стали классическими (см. [22] [23]).

Наряду с языками интерес представляют их гомоморфизмы. Одной из наиболее известных теорем о представлении посредством гомоморфизмов является теорема Хомского — Шюценберже [24], которая утверждает, что про-

извольный контекстно-свободный язык может быть получен как гомоморфный образ пересечения регулярного языка и языка правильных скобочных последовательностей, а также её различные усиленные варианты, например, полученный А.С. Охотиным [25] вариант для нестирающих гомоморфизмов.

Наряду с гомоморфными образами рассматривают и представления языков взятием прообраза при гомоморфизме. Классическим результатом для такого рода представлений является теорема Грейбах «о самом трудном языке» [26]. В работе Грейбах строится такой контекстно-свободный язык L_0, что для любого контекстно-свободного языка L существует гомоморфизм h, для которого выполнено: $L = h^{-1}(L_0)$. Подобного рода представление имеет место не для всех грамматических формализмов. В работе [27] Грейбах доказала, что для класса языков, описываемых $LR(1)$-грамматиками, не существует подобного «самого трудного языка». В дальнейшем Л. Боассон и М. Нива доказали [28] аналогичный отрицательный результат для класса линейных контекстно-свободных языков.

В большинстве вариантов категориальных грамматик одному элементу алфавита Σ может присваиваться несколько различных типов. С лингвистической точки зрения это свойство соответствует явлению омонимии: например, слово «течь» может быть как глаголом, так и существительным в различных контекстах.

Одной из предпосылок данной диссертации является работа А.Н. Сафиуллина [29], где доказывается, что для грамматик Ламбека ограничение на один присваиваемый тип одному символу не сужает класс порождаемых языков. Таким образом, упомянутый выше результат Пентуса переносится на случай однозначного присвоения типов. Основная идея в работе Сафиуллина заключается в том, чтобы имитировать множественность назначений типов одному символу при помощи одного типа. Позднее данный результат и общая конструкция были использованы в работах С.Л. Кузнецова [30] [31], в которых доказывается результат о сложности для исчисления Ламбека, пополненного дополнительной

унарной операцией — звёздочкой Клини.

Представляется естественным вопрос — как условие однозначности присвоения типов повлияет на класс AB-грамматик. В отличие от грамматик Ламбека, подобное ограничение в данном случае оказывается очень существенным. Даже язык a^+ всех непустых слов из одной буквы такие грамматики задать не способны.

Отметим, что AB-грамматики с ограничением на количество присваиваемых типов изучались ранее. Так, в работах В. Бушковского [32] и М. Канадзавы [33] рассмотрены "rigid" и "k-valued" грамматики, в которых одному символу присвоено не более чем один или не более чем k типов соответственно. В работе Канадзавы используются идеи, заложенные в работах Бушковского и Пенна [34] [35], и доказывается, что данные классы грамматик обучаемы в смысле Голда.

Данное свойство, формализующее понятие обучения языку по его конечным фрагментам, было введено в работе Э.М. Голда [36]. Класс языков обучаем по Голду, если для любой бесконечной последовательности слов $w_1, w_2, \ldots$, начиная с некоторого момента t_0, существует не более одного языка из данного класса, порождающего все слова $w_1, \ldots, w_{t_0}$. Таким образом, любой язык из данного класса можно «угадать», зная только его достаточно большой конечный фрагмент. Класс языков, порождаемых AB-грамматиками (т.е. класс контекстно-свободных языков без пустого слова), не обучаем по Голду. С другой стороны, в работе Канадзавы [33] доказана обучаемость класса языков, задаваемых AB-грамматиками с ограничением k на число присваиваемых типов для любого фиксированного k. Более того, процесс обучения можно сделать эффективным: существует алгоритм, принимающий на вход последовательность слов $w_1, w_2, \ldots$ и пытающийся «угадать» грамматику. Начиная с некоторого момента t_0, ответы, выдаваемые данным алгоритмом, стабилизируются на грамматике, правильно задающей данный язык.

Одним из часто встречающихся вариантов сужения AB-грамматик яв-

ляется рассмотрение подклассов, полученных ограничением на типы, которые могут быть использованы при присвоении. В частности, упомянутое выше соответствие между AB-грамматиками и контекстно-свободными выражается не только в классе порождаемых языков, но и в эффективном преобразовании данных грамматик друг в друга. А именно, в преобразовании произвольной контекстно-свободной грамматики без пустого слова в форму AB-грамматики, за счёт формы Грейбах [37] возможно добиться отсутствия деления на сложные типы. Более того, любой контекстно-свободный язык без пустого слова может быть порождён AB-грамматикой с типами трёх видов (p, q, r — произвольные примитивные типы): p, (p/q), $((p/q)/r)$. Таким образом, естественно рассматривать AB-грамматики с делениями только на примитивные типы, такие грамматики в данной работе носят название «базовые категориальные грамматики». Заметим, что подобного рода ограничения встречаются во многих формализмах, основанных на AB-грамматиках. Так, упомянутые выше категориальные грамматики зависимостей и конъюнктивные категориальные грамматики обладают именно таким ограничением. Стоит отметить, что базовыми категориальными грамматиками с однозначным присвоением типов посредством естественного гомоморфизма можно представить любой контекстно-свободный язык без пустого слова.

Хотя в работе Канадзавы исследуются AB-грамматики с однозначным присвоением типов, основной акцент делается на понятии обучаемости грамматике по Голду. При этом оставались открытыми вопросы, насколько данный подкласс AB-грамматик ограничен, а также насколько сложны связанные с ним алгоритмические задачи. Вопрос об алгоритмической разрешимости задачи проверки пустоты пересечения языков, заданных данными грамматиками, решен А. Форе [38]. Другие алгоритмические свойства данного класса, однако, оставались неизученными. Интуитивно, условие однозначности убирает весь возможный «недетерминизм» в грамматике, единственное, что остаётся, — это возможность различным образом редуцировать одну выбранную строку типов.

Следовательно, распознавание и другие задачи могли бы решаться проще, чем для контекстно-свободных языков.

В то время как ограничение на количество присвоенных типов одному символу для AB-грамматик было исследовано в упомянутой выше работе Канадзавы (в частности, получена теорема об иерархии по числу присвоений), для базовых категориальных грамматик данный результат ранее не был известен. Для данных грамматик мы ставим вопросы, аналогичные полученным для более широких классов.

Цели и задачи диссертации

В работе изучаются классы категориальных грамматик, полученные из класса AB-грамматик при помощи следующих ограничений: ограничение множества возможных присваиваемых типов и ограничение на количество типов, присваиваемых одному символу в грамматике. Основной задачей работы является изучение свойств, в частности алгоритмических, для указанных классов грамматик.

Методы исследования

В работе используются методы математической логики, теории формальных грамматик и теории алгоритмов.

Теоретическая и практическая ценность работы

Работа имеет теоретический характер. Разработанные методы могут быть использованы в дальнейших исследованиях. В частности, предполагается, что кодирование множественности выбора типов применимо при рассмотрении других формализмов. Некоторые результаты могут иметь практическое применение при автоматизированном анализе конструкций языков программирования.

Научная новизна и положения, выносимые на защиту

Все основные результаты диссертации являются новыми, получены автором самостоятельно и состоят в следующем.

- Доказательство строгости иерархии классов языков, порождаемых базовыми категориальными грамматиками с присвоением не более k типов одному символу ($\mathcal{L}^k$).

- Доказательство незамкнутости класса языков, порождаемых базовыми категориальными грамматиками с однозначным присвоением типов ($\mathcal{G}^1$), относительно операций объединения, пересечения, конкатенации, итерации Клини.

- Построение сведения посредством гомоморфизмов произвольного контекстно-свободного языка к языку, порождённому базовой категориальной грамматикой с двузначным присвоением типов (из класса $\mathcal{G}^2$). Как следствие, доказана алгоритмическая неразрешимость некоторых классических задач для категориальных грамматик из данного класса.

- Доказательство неразрешимости задачи проверки пустоты пересечения двух языков, порождаемых базовыми категориальными грамматиками с однозначным присвоением типов, над фиксированным алфавитом из 11 букв.

- Доказательство алгоритмической неразрешимости задачи проверки включения языков, заданных AB-грамматиками с однозначным присвоением типов (поставлена в статье Форе [38]).

Степень достоверности и апробация результатов

Достоверность результатов обеспечена строгими математическими доказательствами. Они опубликованы в рецензируемых научных журналах (см. работы [44] [45] [46]). Результаты диссертации докладывались на следующих научных семинарах и конференциях.

- Международная конференция «Мальцевские чтения» (Новосибирск, ИМ им. С. Л. Соболева СО РАН и НГУ, с 11.11.2024 по 15.11.2024).

- Семинар «Теория доказательств» (Logic Online Seminar) отдела математической логики МИАН (Москва, 01.03.2021, 28.10.2024).
- Семинар «Категориальные грамматики» (Москва, МГУ, 12.05.2024).
- Объединённый семинар «Логические проблемы информатики» и «Модальная и алгебраическая логика» (Москва, МГУ, 05.12.2019).
- Семинар «Теоретические основы информатики» (Тверь, ТвГУ, 24.12.2024).
- Однодневный семинар по математической логике (Москва, НИУ ВШЭ, 29.06.2023).
- Международная научная конференция студентов, аспирантов и молодых учёных «Ломоносов-2021» (Москва, МГУ, с 12.04.2021 по 23.04.2021).
- Международная научная конференция студентов, аспирантов и молодых учёных «Ломоносов-2025» (Москва, МГУ, с 11.04.2025 по 25.04.2025).

Структура и объём диссертации

Диссертация состоит из введения, трёх глав, заключения и библиографии. Объём работы составляет 92 страницы. Библиография включает 49 наименований.

Краткое содержание работы

Во введении даётся общая характеристика работы, формулируются постановки задач, цели работы и обосновывается её актуальность. Кроме того, приводится обзор результатов, полученных ранее другими авторами по теме диссертации и смежным направлениям исследований, а также кратко излагается содержание глав диссертации.

Первая глава вводная. Основной результат данной главы служит мотивировкой для рассмотрения условия ограниченного присвоения типов в базовых категориальных грамматиках. Сперва вводятся AB-грамматики и базовые

категориальные грамматики. Затем доказывается равенство класса языков, заданных вторым формализмом, и классического класса контекстно-свободных языков без пустого слова. В конце рассматриваются классы грамматик и, соответственно, заданных ими языков, полученных из данного формализма наложением ограничения на количество типов, присваиваемых одному символу. Перечислим основные положения данной главы.

В параграфе 1.1 вводится понятие контекстно-свободной грамматики и AB-грамматики.

Определение 1.1.3 (Бар-Хиллел и др. [7])**.** Пусть дано конечное множество $\mathrm{Pr} = \{p, q, r, \dots\}$ *примитивных типов*. Определим множество *типов* Tp индуктивно:

- если p — примитивный тип, то p — тип;
- если A и B — типы, то $(B\backslash A)$ и (A/B) — типы.

При записи строк типов типы разделяются точкой с запятой (;), а внешние скобки опускаются.

AB-грамматика есть упорядоченная четвёрка $(\Sigma, \mathrm{Pr}, S, \triangleright)$, где Σ — конечный алфавит, Pr — конечное множество примитивных типов, $S \in \mathrm{Tp}$ (S называется *результирующим типом*) и $\triangleright$ — произвольное конечное бинарное отношение $\triangleright \subseteq \Sigma \times \mathrm{Tp}$, *присваивающее* каждой букве алфавита некоторый набор типов.

Строки типов могут быть *преобразованы* при помощи следующих *правил редукции*, каждое из которых сокращает два рядом стоящих типа в один.

$$\frac{A/B \qquad B}{A} \qquad\qquad \frac{B \qquad B\backslash A}{A}$$

Соответственно, будем говорить, что строка типов α *редуцируется за один шаг* к строке типов β ($\alpha \Rightarrow \beta$), если для некоторых типов A и B, а также для некоторых строк типов η и θ выполнено, что: $\alpha = \eta; (A/B); B; \theta$ или

$\alpha = \eta; B; (B \backslash A); \theta$, а строка β равна $\eta; A; \theta$. Будем говорить, что строка типов α *редуцируется* к строке типов β ($\alpha \Rightarrow^* \beta$), если строку β можно получить из α при помощи нуля или нескольких применений указанных выше правил.

Слово $a_1 \dots a_n$ *порождается* AB-грамматикой $G = (\Sigma, \mathrm{Pr}, S, \triangleright)$, если существуют такие типы $B_1, \dots, B_n$, что для каждого i выполняется $a_i \triangleright B_i$, и строка $B_1; \dots; B_n$ может быть редуцирована к строке из единственного типа S.

Язык $L(G)$, *задаваемый* AB-грамматикой G, определяется как множество всех слов $a_1 \dots a_n$ в алфавите Σ, порождаемых грамматикой G.

Известно, что AB-грамматики задают все контекстно-свободные языки без пустого слова и только их. Затем даётся определение «k-значной» AB-грамматики, которое ранее было введено в работе [33], а также теорема, утверждающая, что «k-значные» грамматики при разных значениях параметра k задают строго вложенные классы языков.

Определение 1.1.5 (Канадзава [33]). AB-грамматика $(\Sigma, \mathrm{Pr}, S, \triangleright)$ называется *«k-значной»* AB-грамматикой, если для любого символа $a \in \Sigma$ выполнено, что $|\{A \mid a \triangleright A\}| \leqslant k$.

Теорема 1.1.6 (Канадзава [33]). Для произвольного натурального числа k существует такой язык L, что L задаётся «$(k+1)$-значной» AB-грамматикой, но не задаётся никакой «k-значной» AB-грамматикой.

В параграфе 1.2 вводится понятие базовых категориальных грамматик, подкласса AB-грамматик, в котором запрещены типы вида (A/B) и $(B \backslash A)$, в которых B — не примитивный тип.

В параграфе 1.3 определяется способ преобразования произвольной базовой категориальной грамматики в контекстно-свободную грамматику особенного вида.

В параграфе 1.4 вводится класс базовых категориальных грамматик с присвоением не более k типов.

Определение 1.4.1. Базовая категориальная грамматика $(\Sigma, \mathrm{Pr}, S, \triangleright)$ принадлежит классу $\mathcal{G}^k$, если для любого символа $a \in \Sigma$ выполнено, что $|\{A \mid a \triangleright A\}| \leqslant k$. Обозначим через $\mathcal{L}^k$ класс языков, задаваемых грамматиками из класса $\mathcal{G}^k$.

Дальше изучается соотношение классов $\mathcal{L}^k$ и $\mathcal{L}^{k+1}$ при произвольном k. Очевидно, что данные классы вложены, но появляется вопрос, а могло ли иметь место совпадение классов языков при некотором k. При конкретных небольших значениях k это оказывается достаточно легко проверить на примерах языков, лежащих в классе $\mathcal{L}^2$, но не лежащих в классе $\mathcal{L}^1$, а также лежащих в $\mathcal{L}^3$, но не в $\mathcal{L}^2$. В итоге параграф завершается доказательством теоремы об иерархии, аналог которой для случая AB-грамматик доказан в работе [33].

Теорема 1.4.7. Для произвольного натурального числа k существует такой язык L, что L задаётся грамматикой из класса $\mathcal{G}^{k+1}$, но не задаётся никакой грамматикой из класса $\mathcal{G}^k$.

Вторая глава содержит основные технические результаты данной работы. Сперва доказывается существование естественных инвариантов для сокращения произвольной строки типов. Далее доказывается, что класс $\mathcal{G}^1$ не замкнут относительно естественных операций на языках. В третьем параграфе доказываются теоремы о сведении посредством гомоморфизмов контекстно-свободных языков к языкам, заданным AB-грамматиками с однозначным присвоением типов. В последнем параграфе доказывается усиленный вариант аналогичной теоремы для языков, заданных грамматиками из класса $\mathcal{G}^2$.

В параграфе 2.1 вводится понятие счётчика по примитивному типу p.

Определение 2.1.1 (ван Бентем [39, гл. 6])**.** Пусть p — примитивный тип. Тогда значением *счётчика по* p для типа C, которое обозначается как $\#_p(C)$, называется целое число, определённое индукцией по структуре типа C следующим образом:

- если C – примитивный тип p, то $\#_p(C) = 1$;

- если C – примитивный тип q, не равный p, то $\#_p(C) = 0$;
- если C имеет вид A/B, то $\#_p(A/B) = \#_p(A) - \#_p(B)$;
- если C имеет вид $B\backslash A$, то $\#_p(B\backslash A) = \#_p(A) - \#_p(B)$.

Определение расширяется на строку типов $A_1; \ldots; A_n$:

$$\#_p(\varepsilon) = 0 \text{ и } \#_p(A_1; \ldots; A_n) = \#_p(A_1) + \ldots + \#_p(A_n).$$

Дальше доказывается, что счётчики не изменяются при редукции строк типов.

Лемма 2.1.2. Пусть строка типов u редуцируется к строке типов v. Тогда $\#_p(u) = \#_p(v)$ для любого примитивного типа p.

Далее вводится понятие интерпретации в свободной группе.

Определение 2.1.3. Для произвольного типа C его интерпретацию $[\![C]\!]$ в свободной группе, порождённой множеством примитивных типов, определим индукцией:

- если C – примитивный тип p, то $[\![C]\!] = p$;
- если C имеет вид A/B, то $[\![C]\!] = [\![A]\!][\![B]\!]^{-1}$;
- если C имеет вид $B\backslash A$, то $[\![C]\!] = [\![B]\!]^{-1}[\![A]\!]$.

Определение расширяется на строку типов $A_1; A_2; \ldots; A_n$:

$$[\![A_1; A_2; \ldots; A_n]\!] = [\![A_1]\!][\![A_2]\!] \ldots [\![A_n]\!].$$

Интерпретация в свободной группе также является инвариантом при редукциях.

Лемма 2.1.4. Если строка типов u редуцируется к строке типов v, то их интерпретации в свободной группе совпадают.

В конце вводится понятие числителя и знаменателя произвольной строки типов и доказываются технические свойства, требуемые в дальнейших построениях.

В параграфе 2.2 с помощью счётчикового инварианта для строк типов доказывается, что класс языков $\mathcal{L}^1$ не замкнут относительно большинства операций на языках.

Теорема 2.2.2. Для произвольного языка L из класса $\mathcal{L}^1$ язык $\Sigma^+ - L$ не принадлежит классу $\mathcal{L}^1$.

Теорема 2.2.3. Если язык L содержит хотя бы одно непустое слово, то язык L^+ не принадлежит классу $\mathcal{L}^1$.

Теоремы 2.2.4–6. Класс языков $\mathcal{L}^1$ не замкнут относительно конкатенации, объединения и пересечения.

Параграф 2.3 посвящён AB-грамматикам с однозначным присвоением типов. Основной целью данного параграфа является доказательство теоремы о том, что произвольный контекстно-свободный язык без пустого слова можно задать как гомоморфный прообраз некоторого языка, заданного AB-грамматикой с однозначным присвоением типов.

Параграф начинается с рассмотрения основной идеи доказательства финального результата. Идея заключается в имитации выбора из нескольких типов для одной буквы за счёт различных вариантов редукций некоторой строки типов. Для этого сперва для любых двух типов A и B нужно построить строку типов, которая редуцируется как к A, так и к B.

Однако данная цель не реализуема напрямую, что является следствием инвариантности относительно интерпретации в свободной группе. Таким образом, чтобы обойти данные инварианты, можно сделать преобразование, которое произвольные два типа преобразует в типы, интерпретации которых в свободной группе равны. Необходимое преобразование устроено следующим образом.

Определение 2.3.1. Определим функцию ϕ как отображение из множества типов во множество типов по индукции:

- если C – примитивный тип p, то $\phi(C) = p/p$;
- если C имеет вид A/B, то $\phi(C) = \phi(A)/\phi(B)$;
- если C имеет вид $B\backslash A$, то $\phi(C) = \phi(B)\backslash\phi(A)$.

Для произвольных типов A и B существует строка типов, которая редуцируется как к $\phi(A)$, так и к $\phi(B)$. Однако от этих строк нам требуется больше, чем просто возможность редукции к нескольким разным типам. Нам необходимо, чтобы составленная из конкатенации такого рода «блоков» строка, в случае редукции к некоторому типу C, всегда редуцировалась «правильным» образом. Чтобы каждый из «блоков» сперва редуцировался к единственному типу, а затем полученные типы редуцировались между собой. Иначе говоря, данные «блоки» не должны «перемешиваться» между собой. Для этого вместо ϕ используется модифицированное преобразование ψ.

Определение 2.3.7. Для произвольного типа A положим значение $\psi(A)$ равным $\ell\backslash(\phi(A)/r)$.

Уже для этого преобразования доказывается следующая лемма.

Лемма 2.3.8. Пусть A и B — различные типы вида p, p/q или $(p/q)/s$. Тогда существует строка типов $u_{A,B}$, такая что:

I. $u_{A,B}$ редуцируется к $\psi(A)$ и к $\psi(B)$;

II. никакой собственный суффикс $u_{A,B}$ не редуцируется к $\psi(A)$, и никакой собственный префикс $u_{A,B}$ не редуцируется к $\psi(B)$.

Далее при помощи строки $u_{A,B}$ строится финальная строка $w_{A_1,\ldots,A_n}$, ко-

торая обобщает выбор между двумя типами на множественный случай:

$$w_{A_1,\ldots,A_n} = x_1; u_{A_1,A_2}; x_2; u_{A_2,A_3}; x_3; \ldots; x_{n-1}; u_{A_{n-1},A_n}; x_n, \text{ где}$$
$$x_1 = \psi(A_1)/e_1; e_1/\psi(A_2),$$
$$x_i = \psi(A_{i-1})\backslash e_{2i-2}; (e_{2i-2}\backslash\psi(A_i))/e_{2i-1}; e_{2i-1}/\psi(A_{i+1}),$$
$$\text{для всех } i \in \{2,\ldots,n-1\},$$
$$x_n = \psi(A_{n-1})\backslash e_{2(n-1)}; e_{2(n-1)}\backslash\psi(A_n).$$

Таким образом, для построенной строки выполнена следующая лемма.

Лемма 2.3.11. Пусть $A_1, \ldots, A_n$, где $n \geqslant 1$ — произвольные попарно различные типы форм p, p/q или $(p/q)/s$. Тогда строка типов $w_{A_1,\ldots,A_n}$ редуцируется к каждому из типов $\psi(A_1)$, …, $\psi(A_n)$.

После этого фиксируется контекстно-свободная грамматика в нормальной форме Грейбах. Для каждого символа a рассматриваются все правила вида $X \to a$, $X \to aY$ и $X \to aYZ$, где данный символ стоит справа на первом месте. Нетерминальные символы далее будут использованы как примитивные типы. Затем a сопоставляется строка типов, которая имитирует выбор одного из этих правил. Итого, определяется гомоморфизм h, отображающий каждую букву a в соответствующую строку типов. Технические условия, которым удовлетворяла $u_{A,B}$, устроены так, чтобы была выполнена следующая ключевая лемма.

Лемма 2.3.13. Если $h(y) = h(a_1 \ldots a_n)$ редуцируется к типу $X/X = \phi(X)$, то существует строка типов формы $\phi(C_1); \ldots; \phi(C_n)$, которая редуцируется к X/X, где каждый тип C_i выбран из множества:

$$\{\, X \mid X \to a_i \in R \,\} \cup \{\, X/Y \mid X \to a_iY \in R \,\} \cup \{\, (X/Y)/Z \mid X \to a_iZY \in R \,\}.$$

С использованием данного свойства для гомоморфизма h доказывается ключевая лемма.

Лемма 2.3.12. Для произвольного слова $y \in \Sigma^+$ и произвольного нетерминального символа $X \in N$, слово y выводимо в грамматике G из нетерминального символа X тогда и только тогда, когда его образ $h(y)$ редуцируется к типу $\phi(X)$.

В итоге, используя понятие сводимости языков, доказывается основная теорема параграфа.

Определение 2.3.17. Язык $L_1 \in \Sigma^*$ *сводится посредством гомоморфизма* к языку $L_2 \in \Omega^*$ ($L_1 \leqslant L_2$), если существует гомоморфизм $g\colon \Sigma \to \Omega^*$ такой, что слово w принадлежит L_1 тогда и только тогда, когда $h(w)$ принадлежит L_2.

Теорема 2.3.18. Для произвольного контекстно-свободного языка $L \subseteq \Sigma^+$ существует AB-грамматика с однозначным присвоением типов G, такая, что $L \leqslant L(G)$.

Таким образом, устанавливается, что AB-грамматики с однозначным присвоением типов достаточно сильны, чтобы закодировать произвольный контекстно-свободный язык.

В параграфе 2.4 показывается, что произвольный контекстно-свободный язык эквивалентен по сводимости языку из класса $\mathcal{L}^2$.

Теорема 2.4.2. Для произвольного контекстно-свободного языка $L \subseteq \Sigma^+$ существует G — грамматика из класса $\mathcal{G}^2$ такая, что $L \leqslant L(G)$ и $L(G) \leqslant L$.

Третья глава посвящена алгоритмическим свойствам ранее введенных формализмов. Сперва рассматриваются грамматики из класса $\mathcal{G}^2$, затем из класса $\mathcal{G}^1$ и в конце AB-грамматики с однозначным присвоением типов.

В параграфе 3.1 показывается, как при помощи теоремы 2.4.2 доказать, что классические задачи для класса грамматик $\mathcal{G}^2$ алгоритмически неразрешимы.

Теорема 3.1.1. Следующие задачи алгоритмически неразрешимы для грам-

матик из класса $\mathcal{G}^2$:

- проверить по двум грамматикам из класса $\mathcal{G}^2$, равны ли заданные ими языки;
- по произвольной грамматике G из класса $\mathcal{G}^2$ проверить, является ли язык, заданный грамматикой G, регулярным.

В параграфе 3.2 доказываются следующие теоремы для класса $\mathcal{G}^1$.

Теорема 3.2.6. Задача определения для произвольной контекстно-свободной грамматики G того, задаётся ли язык $L(G)$ некоторой грамматикой из класса $\mathcal{G}^1$, алгоритмически неразрешима.

Теорема 3.2.7. Задача определения для произвольных грамматик G_1 и G_2 из класса $\mathcal{G}^1$ того, является ли пересечение языков $L(G_1) \cap L(G_2)$ бесконечным языком, алгоритмически неразрешима.

Теорема 3.2.9. Задача определения для произвольных грамматик G_1 и G_2 из класса $\mathcal{G}^1$, где количество символов в алфавитах обеих грамматик равно 11, того, является ли пересечение языков $L(G_1) \cap L(G_2)$ пустым, алгоритмически неразрешима.

Доказательство неразрешимости основано на сведении известной неразрешимой задачи. В данном случае в качестве такой задачи выступает проверка существования решения для произвольной системы соответствий Поста.

Теорема 3.2.9 усиливает ранее известный результат из работы Форе [38], где доказана аналогичная теорема для AB-грамматик, при этом требуется алфавит из существенно большего количества символов.

В параграфе 3.3, при помощи теоремы из параграфа 2.3, доказывается усиление классической теоремы Грейбах о самом трудном языке [26].

Следствие 3.3.1. Существуют алфавит Ω и язык L_0, заданный AB-грамматикой с однозначным присвоением типов, такие, что для произвольного контекст-

но-свободного языка $L \subseteq \Sigma^+$ верно $L \leqslant L_0$.

А также решается следующий вопрос, поставленный в статье Форе [38].

Теорема 3.3.2. Задача определения для произвольных AB-грамматик с однозначным присвоением типов G_1 и G_2 того, что $L(G_1) \subseteq L(G_2)$, алгоритмически неразрешима.

Благодарности

Автор выражает глубокую благодарность своему научному руководителю — доктору физико-математических наук Степану Львовичу Кузнецову за выбор темы, постановку задач и постоянную поддержку. Особенную признательность автор адресует доктору физико-математических наук, профессору Мати Рейновичу Пентусу за внимательное прочтение рукописей и ценнейшие замечания. Кроме того, автор приносит благодарность Ph.D., профессору Александру Сергеевичу Охотину за полезные обсуждения и плодотворную совместную деятельность, а также всем сотрудникам отдела математической логики за создание тёплой и дружелюбной атмосферы.

Глава первая. Иерархия классов языков

1.1. AB-грамматики

Классическим подходом в формализации понятия «синтаксически корректных» предложений в теории формальных языков является определение грамматики как набора правил переписывания строк. Тем самым появляется иерархия по видам правил, которые могут быть использованы. В настоящей работе интересен второй уровень данной иерархии (иерархии Хомского [1]), а именно *контекстно-свободные* грамматики, определение которых дано ниже. Также напомним следующие обозначения. Пусть Σ — некоторый конечный *алфавит*, элементы которого будем называть *символами* (или *буквами*). Множество всех строк (конечных последовательностей, в том числе пустых) над Σ обозначается Σ^*, а его элементы называются *словами*. Множество всех непустых строк обозначается Σ^+, а пустая строка — ε. Произвольное подмножество Σ^* называется (формальным) *языком*.

Определение 1.1.1. Контекстно-свободная грамматика есть четверка $G = (\Sigma, N, S, R)$, где:

- Σ — конечное множество *терминальных* символов (терминалов);
- N — конечное множество *нетерминальных* символов (нетерминалов), такое, что $N \cap \Sigma = \emptyset$;
- S — выбранный нетерминальный символ, т.е. $S \in N$;
- R — конечное подмножество $N \times (N \cup \Sigma)^*$ *правил* вида $X \to u$, где символ X — произвольный нетерминал, а u — строка составленная из терминалов и нетерминалов.

Определение 1.1.2. Язык, *задаваемый* контекстно-свободной грамматикой $G = (\Sigma, N, S, R)$, обозначается $L(G)$ и определяется как множество слов в Σ^*, полученных из выбранного символа S применением нескольких правил из грамматики G:

$$L(G) = \{w \in \Sigma^* \mid S \to^* w\}.$$

Здесь под $\to^*$ понимается рефлексивно-транзитивное замыкание отношения R.

Соответственно, слово $a_1 \dots a_n$ *порождается* контекстно-свободной грамматикой (Σ, N, S, R), если $S \to^* a_1 \dots a_n$.

Существуют и другие формы описания синтаксиса языков. Альтернативным является направление «логических» формализмов, где каждому символу присваивается один или несколько синтаксических типов. Слово же считается «синтаксически корректным», если символам в нём можно назначить типы так, чтобы соответствующая строка типов *редуцировалась* к некоторому выбранному типу. Одним из вариантов являются *AB-грамматики*.

Определение 1.1.3 (Бар-Хиллел и др. [7])**.** Пусть дано конечное множество $\mathrm{Pr} = \{p, q, r, \dots\}$ *примитивных типов*. Определим множество *типов* Tp индуктивно:

- если p — примитивный тип, то p — тип;
- если A и B — типы, то $(B \backslash A)$ и (A / B) — типы.

При записи строк типов типы разделяются точкой с запятой (;), а внешние скобки опускаются.

AB-грамматика есть упорядоченная четвёрка $(\Sigma, \mathrm{Pr}, S, \triangleright)$, где Σ — конечный алфавит, Pr — конечное множество примитивных типов, $S \in \mathrm{Tp}$ (S называется *результирующим типом*) и $\triangleright$ — произвольное конечное бинарное отношение $\triangleright \subseteq \Sigma \times \mathrm{Tp}$, *присваивающее* каждой букве алфавита некоторый набор типов.

Строки типов могут быть *преобразованы* при помощи следующих *правил редукции*, каждое из которых сокращает два рядом стоящих типа в один.

$$\frac{A/B \qquad B}{A} \qquad\qquad \frac{B \qquad B\backslash A}{A}$$

Соответственно, будем говорить, что строка типов α *редуцируется за один шаг* к строке типов β ($\alpha \Rightarrow \beta$), если для некоторых типов A и B, а также для некоторых строк типов η и θ выполнено, что: $\alpha = \eta; (A/B); B; \theta$ или $\alpha = \eta; B; (B\backslash A); \theta$, а строка β равна $\eta; A; \theta$. Будем говорить, что строка типов α *редуцируется* к строке типов β ($\alpha \Rightarrow^* \beta$), если строку β можно получить из α при помощи нуля или нескольких применений указанных выше правил.

Слово $a_1 \dots a_n$ *порождается* AB-грамматикой $G = (\Sigma, \mathrm{Pr}, S, \triangleright)$, если существуют такие типы $B_1, \dots, B_n$, что для каждого i выполняется $a_i \triangleright B_i$, и строка $B_1; \dots; B_n$ может быть редуцирована к строке из единственного типа S.

Язык $L(G)$, *задаваемый* AB-грамматикой G, определяется как множество всех слов $a_1 \dots a_n$ в алфавите Σ, порождаемых грамматикой G.

В общем случае AB-грамматики эквивалентны по выразительной силе контекстно-свободным грамматикам.

Теорема 1.1.4 (Бар-Хиллел и др. [7]). *Произвольный язык $L \subseteq \Sigma^+$ задаётся AB-грамматикой тогда и только тогда, когда он задаётся контекстно-свободной грамматикой.*

Для данного формализма ранее уже рассматривалось условие ограничения количества типов присваиваемых одному символу.

Определение 1.1.5 (Канадзава [33]). AB-грамматика $(\Sigma, \mathrm{Pr}, S, \triangleright)$ называется *«k-значной»* AB-грамматикой, если для любого символа $a \in \Sigma$ выполнено, что $|\{A \mid a \triangleright A\}| \leqslant k$.

Класс грамматик, для которых количество типов, присвоенных одному символу, ограничено единицей, будем называть *AB-грамматиками с однознач-*

ным присвоением типов. Классы «k-значных» AB-грамматик вложены. Оказывается, данное вложение строгое, то есть верна следующая теорема.

Теорема 1.1.6 (Канадзава [33])**.** *Для произвольного натурального числа k существует такой язык L, что L задаётся «$(k+1)$-значной» AB-грамматикой, но не задаётся никакой «k-значной» AB-грамматикой.*

Следующий пример показывает, что AB-грамматики с однозначным присвоением типов не могут задать даже самые простые языки.

Пример 1. Язык a^+ не может быть задан никакой AB-грамматикой с однозначным присвоением типов.

Доказательство. Так как слово a принадлежит языку, типом для символа a должен быть S. Но тогда последовательность типов $S; S$ не редуцируется ни к какому типу, а значит, никакое другое слово не может принадлежать языку. □

1.2. Базовые категориальные грамматики

В данном параграфе мы вводим базовые категориальные грамматики. В отличии от AB-грамматик, здесь разрешаются деления только на примитивные типы, а не на произвольные.

Определение 1.2.1. Пусть дано конечное множество $\mathrm{Pr} = \{p, q, r, \ldots\}$ *примитивных типов*. Определим множество *базовых типов* Bt индуктивно:

- если p — примитивный тип, то p — базовый тип;
- если A — базовый тип, а q — примитивный тип, то (A/q) и $(q\backslash A)$ — базовые типы.

Произвольную AB-грамматику $(\Sigma, \mathrm{Pr}, S, \triangleright)$ будем называть *базовой категориальной грамматикой* (БКГ), если $S \in \mathrm{Pr}$, а $\triangleright \subseteq \Sigma \times \mathrm{Bt}$.

Тем самым в базовых категориальных грамматиках могут быть использованы только базовые типы, в которых запрещаются деления на сложные типы. В силу указанного определения редукции в строках базовых типов могут иметь один из следующих видов.

$$\frac{A/q \qquad q}{A} \qquad\qquad \frac{q \qquad q\backslash A}{A}$$

Здесь A — произвольный базовый тип, а q — произвольный примитивный тип.

Отметим несколько допущений, позволяющих сильно упростить обозначения. Как и в случае AB-грамматик, внешние скобки в типах будем опускать, а в записи строки типов отдельные типы разделять точками с запятой.

Пусть G — произвольная базовая категориальная грамматика. Вместо того, чтобы выписывать все элементы множества $\triangleright$, будем писать все типы для одного символа в строчку через символ $|$, аналогично тому, как подобное делается для правил контекстно-свободной грамматики. Например, обозначение $a \triangleright r \mid r/p \mid p$ значит, что символу a присвоены типы r, r/p и p. Или, формально, что пары (a, r), $(a, r/p)$ и (a, p) принадлежат множеству присвоений $\triangleright$.

Везде, где не указано обратное, будем обозначать результирующий тип грамматики r. Тем самым, для БКГ будем писать лишь присвоения типов символам, а не всю грамматику целиком. Например, под $a \triangleright p \mid r \mid r/r$ будем понимать грамматику $(\{a\}, \{p, r\}, r, \{(a, p), (a, r), (a, (r/r))\})$.

Редукции строк типов могут быть представлены как деревьями разбора, так и последовательностями редукций.

Пример 2. Строка типов $q; q\backslash(p/p)/p; p; p$ редуцируется к типу p.

$$\dfrac{\dfrac{\dfrac{q \quad q\backslash(p/p)/p}{(p/p)/p} \qquad p}{p/p} \qquad p}{p} \qquad \text{или} \qquad q; q\backslash(p/p)/p; p; p \Rightarrow (p/p)/p; p; p \Rightarrow p/p; p \Rightarrow p$$

Очевидно, что строки типов могут быть редуцированы совершенно разным образом. Но может ли существовать строка, редуцируемая к двум различ-

ным типам? Следующий пример показывает, что ответ на этот вопрос положительный.

Пример 3. Строка типов $(p/p)/p; p; p; p\backslash(p\backslash p)$ редуцируется к p/p и к $p\backslash p$.

$$\dfrac{\dfrac{\dfrac{(p/p)/p \quad p}{p/p} \quad p}{p} \quad p\backslash(p\backslash p)}{p\backslash p} \qquad \dfrac{(p/p)/p \quad \dfrac{p \quad \dfrac{p \quad p\backslash(p\backslash p)}{p\backslash p}}{p}}{p/p}$$

1.3. Преобразование БКГ в контекстно-свободную грамматику

Наша цель в данном параграфе — построить преобразование из БКГ в контекстно-свободные грамматики. Каждому базовому типу сопоставим тип его числителя, а также строки левого и правого знаменателя.

Определение 1.3.1. Для произвольной базового типа C определим тип числителя $nu(C)$, а также строки левого $ld(C)$ и правого $rd(C)$ знаменателей индуктивно:

- для примитивного типа $C = p$: $nu(C) = p$, $ld(C) = \varepsilon$ и $rd(C) = \varepsilon$;
- для C вида $q\backslash B$: $nu(C) = nu(B)$, $ld(C) = ld(B); q$ и $rd(C) = rd(B)$;
- для C вида B/q: $nu(C) = nu(B)$, $ld(C) = ld(B)$ и $rd(C) = q; rd(B)$.

Произвольный контекстно-свободный язык без пустого слова может быть задан некоторой грамматикой в нормальной форме Грейбах [37], которая в свою очередь эквивалентна некоторой БКГ. Тем самым базовые категориальные грамматики задают все контекстно-свободные языки без пустого слова и только их.

В данном параграфе мы рассматриваем контекстно-свободные грамматики специального вида: где во всех правых частях стоит ровно один терминальный символ. Таким грамматикам естественным образом сопоставляются эквивалентные БКГ, минуя приведение к нормальной форме Грейбах [37].

Определение 1.3.2. Пусть дана БКГ $G = (\Sigma, \mathrm{Pr}, r, \triangleright)$. Определим контекстно-свободную грамматику $CF(G) = (\Sigma, \mathrm{Pr}, r, R)$, где отношение R получено из $\triangleright$ следующим образом: для каждого присвоения $a \triangleright A$ введем контекстно-свободное правило $nu(A) \to ld(A)\, a\, rd(A)$.

Теорема 1.3.3. *Языки, заданные БКГ G и контекстно-свободной грамматикой $CF(G)$, совпадают.*

Доказательство. Построим по контекстно-свободной грамматике $CF(G)$ новую контекстно-свободную грамматику $\widetilde{G}$ следующим образом.

Зафиксируем в $CF(G)$ правило $p \to q_k \dots q_1\, a\, p_1 \dots p_m$, где a — терминальный символ из Σ. Добавим в множество нетерминальных символов грамматики $CF(G)$ новые нетерминальные символы $\{h_{i,j} \mid i \in \{1, \dots, k\}$ и $j \in \{1, \dots, m\}\}$, ранее не встречавшиеся в грамматике. Заменим выбранное правило из $CF(G)$ на следующее множество правил:

$$\{p \to h_{k,m}\} \cup \{h_{i,j} \to h_{i,j-1}\, p_j \mid i \in \{1, \dots, k\}, j \in \{1, \dots, m\}\} \cup$$
$$\{h_{i,j} \to q_i\, h_{i-1,j} \mid i \in \{1, \dots, k\}, j \in \{1, \dots, m\}\} \cup \{h_{0,0} \to a\}.$$

Заметим, что если применялось первое правило $p \to h_{k,m}$, то к нетерминалу $h_{k,m}$ должны применяться правила из второй и третьей группы, пока не останется $h_{0,0}$, а затем последнее правило $h_{0,0} \to a$. Обратно, любой нетерминальный символ $h_{i,j}$ получен из нетерминала $h_{k,m}$, который в свою очередь получен из правила $p \to h_{k,m}$.

Итого, грамматики $CF(G)$ и $\widetilde{G}$ задают равные языки. Осталось отметить, что произвольное дерево разбора в грамматике $\widetilde{G}$ слова $a_1 \dots a_n$ совпадает с

деревом разбора некоторой строки типов $A_1; \ldots; A_n$, присвоенных данным символам в исходной грамматике G. $\square$

Отметим важное свойство указанного выше преобразования: если в некоторой грамматике G символу a было присвоено k типов, то в $CF(G)$ существует ровно k правил, где символ a встречается в правой части.

Приведём пример подобного преобразования для фиксированной БКГ.

Пример 4. Пусть G задана левым набором присвоений, а $CF(G)$ задан правым набором правил:

$$
\begin{array}{ll}
a \triangleright q\backslash(r/p) \mid r/r & r \to q\,a\,p \mid a\,r \mid q\,b\,p\,p \\
b \triangleright (q\backslash(r/p))/p \mid p & p \to b \\
c \triangleright q & q \to c
\end{array}
$$

Отметим, что язык, заданный БКГ, не меняется от порядка скобок в присвоенных типах, что является прямым следствием теоремы 1.3.3. Говоря точнее, для произвольных грамматик G_1 и G_2, которые отличаются лишь порядком скобок в присваиваемых типах, $CF(G_1)$ и $CF(G_2)$ — это контекстно-свободные грамматики с одинаковым набором правил. Таким образом, в дальнейшем, если не сказано обратное, все скобки будем опускать. Например, тип $q\backslash(r/p)$ будем писать как $q\backslash r/p$.

1.4. Иерархия классов языков с присвоением k типов

От рассмотрения произвольных БКГ перейдём к случаю, когда количество присвоений для одной буквы ограничено некоторым числом. Дадим определение класса грамматик $\mathcal{G}^k$ с не более k присвоениями.

Определение 1.4.1. Базовая категориальная грамматика $(\Sigma, \mathrm{Pr}, S, \triangleright)$ принадлежит классу $\mathcal{G}^k$, если для любого символа $a \in \Sigma$ выполнено, что $|\{A \mid a \triangleright A\}| \leqslant$

k. Обозначим через $\mathcal{L}^k$ класс языков, задаваемых грамматиками из класса $\mathcal{G}^k$.

Отметим простое следствие определения.

Предложение 1.4.2. *Для произвольного k, если грамматика $G \in \mathcal{G}^k$, то $G \in \mathcal{G}^{k+1}$. Аналогично, если язык $L \in \mathcal{L}^k$, то $L \in \mathcal{L}^{k+1}$.*

Как уже отмечалось, любой контекстно-свободный язык без пустого слова задаётся некоторой базовой категориальной грамматикой. Тем самым, любой контекстно-свободный язык без пустого слова принадлежит какому-то из классов $\mathcal{L}^k$.

Также, отметим, что произвольная грамматика из класса $\mathcal{G}^k$ является «k-значной» AB-грамматикой. Более того, данное вложение классов языков строгое, что демонстрируется следующим примером.

Пример 5. AB-грамматика $G = (\Sigma, \mathrm{Pr}, S, \triangleright)$, где $\Sigma = \{a, b, c\}$, $\mathrm{Pr} = \{p\}$, присвоения типов заданы как $a \triangleright p$, $b \triangleright (p/p)/p$ и $c \triangleright p/((p/p)/p)$, а результирующим типом является p, задаёт язык, не задающийся никакой грамматикой из класса $\mathcal{G}^1$.

Доказательство. Пусть язык $L(G)$ задаётся грамматикой G с результирующим типом r, где символам a, b и c присвоены базовые типы A, B и C соответственно. Так как данная грамматика порождает слова a, baa и cb, то каждая из строк типов A, $B; A; A$ и $C; B$ редуцируется к r. Следовательно, тип A равен результирующему типу r. Так как строка $B; r; r$ редуцируется к типу r, то тип B должен быть $(r/r)/r$. Но строка типов $C; (r/r)/r$ не редуцируется ни к какому базовому типу C. □

Рассмотрим тривиальные примеры языков из различных классов $\mathcal{L}^k$.

Пример 6. Язык $\{aa\}$ задаётся грамматикой из класса $\mathcal{G}^2$, но не задаётся никакой грамматикой из класса $\mathcal{G}^1$.

Доказательство. Чтобы была возможна хотя бы одна редукция типов в грамматике, букве a должен быть назначен хотя бы один примитивный тип. Если a назначен тип r, то грамматика должна порождать слово из одного символа a, что не так. Иначе, a назначен другой тип p, но тогда слово aa не будет порождаться грамматикой. Тем самым, данный язык не в классе $\mathcal{L}^1$, но грамматика $a \triangleright p \mid r/p$ из класса $\mathcal{G}^2$ задаёт данный язык. □

Следовательно, мы доказали строгое вложение $\mathcal{L}^1$ в $\mathcal{L}^2$. Следующий пример показывает, что $\mathcal{L}^2 \subsetneq \mathcal{L}^3$.

Пример 7. Язык $\{a, aa\}$ не задаётся никакой грамматикой из $\mathcal{G}^2$, но задаётся грамматикой $a \triangleright r \mid p \mid p\backslash r$ из $\mathcal{G}^3$.

Доказательство. Чтобы слово a порождалось грамматикой, в ней символу a должен быть присвоен тип r. Рассмотрим редукции типов для слова aa. Есть две возможности: первой a присвоен примитивный тип p, а второй a присвоен тип $p\backslash r$ или второй a присвоен примитивный тип p, а первой a — тип r/p. Тем самым, чтобы и слово a, и слово aa порождались грамматикой из $\mathcal{G}^2$, необходимо, чтобы символу a были присвоены типы r/r и r, или $r\backslash r$ и r. Но обе эти грамматики задают большие языки, в частности, слово aaa порождается обеими грамматиками. □

Итого, мы показали, что $\mathcal{L}^1 \subsetneq \mathcal{L}^2 \subsetneq \mathcal{L}^3$.

Наша ближайшая цель состоит в том, чтобы обобщить данные примеры на случай грамматик с k типами. Как мы видим, различия в классах $\mathcal{L}^1$, $\mathcal{L}^2$ и $\mathcal{L}^3$ появляются в достаточно простых языках. В действительности, конечных языков хватит, чтобы различить $\mathcal{L}^k$ и $\mathcal{L}^{k+1}$ при произвольном k. Для доказательства нам потребуется следующее техническое утверждение.

Предложение 1.4.3. *Пусть G — БКГ, задающая конечный язык. Тогда существуют эквивалентная G БКГ G' и терминальный символ a, которому присвоен примитивный тип p такой, что никакая строка из двух и более*

типов, встречающихся в присвоениях типов грамматики G', не редуцируется к p. Более того, если $G \in \mathcal{G}^k$, то $G' \in \mathcal{G}^k$.

Доказательство. Сперва устраним в грамматике присвоения типов, которые не могут участвовать ни в какой редукции, что делается стандартным образом, аналогично контекстно-свободным грамматикам.

Обозначим полученную грамматику G'. Рассмотрим слово $w = a_1 \dots a_n$ максимальной длины в языке $L(G')$. Следовательно, существуют такие присвоения типов символам a_i, что строка $A_1; \dots; A_n$ редуцируется к типу r.

Если $w = a_1$, то в грамматике G' символу a_1 присвоен тип r, и так как это слово максимальной длины, то никакие другие типы из присвоений грамматики G' не могут редуцироваться к r, тем самым получаем требуемое утверждение.

Теперь рассмотрим случай, когда w состоит из двух или более символов. Чтобы была возможна хотя бы одна редукция, один из типов A_i должен быть примитивным типом p. Отметим, что p не может быть результирующим типом (т.е. не равен r). Иначе слово

$$a_1 \dots a_{i-1} a_1 \dots a_{i-1} a_i a_{i+1} \dots a_n a_{i+1} \dots a_n$$

порождается грамматикой G', что демонстрируется следующей последовательностью редукций:

$$A_1 \dots A_{i-1}; A_1; \dots; A_{i-1}; A_i; A_{i+1}; \dots; A_n; A_{i+1}; \dots; A_n \Rightarrow$$
$$\Rightarrow A_1 \dots A_{i-1}; r; A_{i+1}; \dots; A_n \Rightarrow r.$$

Это противоречит максимальности длины слова w.

Теперь предположим, что какая-то из строк типов, встречающихся в G', может редуцироваться к p. Обозначим её α, а слово, из которого присвоениями получается α, обозначим u. Рассмотрим слово $a_1 \dots a_{i-1} u a_{i+1} \dots a_n$ и соответствующее присвоение типов его символам $A_1; \dots; A_{i-1}; \alpha; A_{i+1}; \dots; A_n$. Следующие редукции показывают, что строка типов $A_1; \dots; A_{i-1}; \alpha; A_{i+1}; \dots; A_n$ ре-

дуцируется к r:

$$A_1; \ldots; A_{i-1}; \alpha; A_{i+1}; \ldots; A_n \Rightarrow A_1; \ldots; A_{i-1}; p; A_{i+1}; \ldots; A_n \Rightarrow r.$$

Значит, слово $a_1 \ldots a_{i-1} u a_{i+1} \ldots a_n$ принадлежит языку грамматики G', что противоречит максимальности длины слова w. □

Нам потребуются следующие функции.

Определение 1.4.4. Для произвольного контекстно-свободного языка $L \subseteq \Sigma^+$ определим значение функции λ как такое минимальное число k, что L может быть задан грамматикой из $\mathcal{G}^k$.

Определение 1.4.5. Для произвольного конечного языка $L \subseteq \Sigma^+$ определим значение функции μ как такое минимальное число k, что существует конечный язык L' с условием $L \subseteq L'$, и L' может быть задан грамматикой из $\mathcal{G}^k$.

Предложение 1.4.6. *Функции λ и μ удовлетворяют следующим свойствам:*

- *пусть L — произвольный конечный язык, тогда $\mu(L) \leqslant \lambda(L)$;*
- *пусть L и $\tilde{L}$ — конечные языки такие, что $L \subseteq \tilde{L}$, тогда $\mu(L) \leqslant \mu(\tilde{L})$.*

Доказательство. Отметим, что $\mu(L)$ — это минимальное количество типов, необходимое, чтобы задать конечный язык, содержащий L, следовательно, $\mu(L)$ меньше или равно количеству типов, необходимых для задания L.

Для доказательства второго утверждения достаточно заметить, что минимум $\mu(L)$ берётся по большему множеству языков, чем $\mu(\tilde{L})$. □

Теперь с помощью функции λ сформулируем основную теорему данного параграфа и один из основных результатов диссертации.

Теорема 1.4.7. *Для произвольного натурального числа k существует такой язык L, что L задаётся грамматикой из класса $\mathcal{G}^{k+1}$, но не задаётся никакой грамматикой из класса $\mathcal{G}^k$, иначе говоря, $\lambda(L) = k$.*

Доказательство. Рассмотрим значение функции μ на языках вида $L_n = \{a^i \mid i \in \{1, \ldots, n\}\}$ при различных n.

Лемма 1.4.8. *Значения функции μ неограничены на множестве языков $\{L_n \mid n \in \mathbb{N}\}$.*

Доказательство. Доказательство от противного. Пусть l — наибольшее значение функции μ, которое достигается на языке $L_t = \{a^i \mid i \in \{1, \ldots, t\}\}$.

Рассмотрим язык $L_{(t+1)^{l+1}+1} = \{a^i \mid i \in \{1, \ldots, (t+1)^{l+1}+1\}\}$ и базовую категориальную грамматику G, задающую данный язык. Заметим, что $\mu(L_{(t+1)^{l+1}+1}) = l$ по предположению и второй части утверждения 1.4.6. По определению μ это значит, что существует конечный язык $\tilde{L}$, содержащий язык $L_{(t+1)^{l+1}+1}$, и $\lambda(\tilde{L}) = l$.

В языке $\tilde{L}$ точно содержатся $(t+1)^{l+1}+1$ различных слов, так как $\tilde{L}$ содержит язык $L_{(t+1)^{l+1}+1}$. Более того, язык $\tilde{L}$ конечен и задаётся грамматикой из $\mathcal{G}^l$, обозначим эту грамматику G, а типы, присвоенные символу a в этой грамматике, $A_1, \ldots, A_l$.

Рассмотрим $CF(G)$. Глубина деревьев вывода в данной грамматике ограничена $l+1$, иначе существовал бы примитивный тип, повторяющийся в одной ветви дерева вывода, а следовательно, повторением этой ветви можно было бы получить слово длины больше произвольной константы.

Предположим, что в каждом правиле грамматики $CF(G)$ с правой стороны не более чем t символов. Значит, на каждом шаге добавляется не более t новых вершин, а следовательно, в итоговом слове не более $(t+1)^l$ символов. Таким образом, существует такое правило $p \to u$, что в правой части минимум $t+1$ символ.

Правило $p \to u$ не может быть использовано в выводе слов длины меньше $t+1$. Следовательно, грамматика $CF(G)$ с удаленным правилом $p \to u$ обязана порождать все слова с длиной не более чем t. Без ограничения общности будем считать, что в изначальной грамматике G правило $p \to u$ соответствовало

присвоению $a \triangleright A_1$. Рассмотрим G' — грамматику, полученную из грамматики G удалением присвоения $a \triangleright A_1$. По построению $CF(G')$ отличается от $CF(G)$ только правилом $p \to u$, а следовательно, G' порождает все слова с длиной не более чем t, а значит, $\{a^i \mid i \in \{1, \ldots, t\}\} \subseteq L(G)$. Однако, в грамматике G' всего $l - 1$ присвоение, следовательно, $\mu(\{a^i \mid i \in \{1, \ldots, t\}\}) \leqslant l - 1$, что противоречит предположению. □

Лемма 1.4.9. *Если $k \geqslant 2$, то верно $\lambda(\{a^i \mid i \in \{1, \ldots, k+1\}\}) \leqslant \lambda(\{a^i \mid i \in \{1, \ldots, k\}\}) + 1$.*

Доказательство. Пусть грамматика G задаёт язык $\{a^i \mid i \in \{1, \ldots, k\}\}$. По утверждению 1.4.3 существует такая грамматика G', задающая тот же язык, и такой примитивный тип p, присвоенный символу a в G', что никакая строка из двух и более типов в данной грамматике не может редуцироваться к p. Обозначим присвоения в G' следующим образом: $a \triangleright A_1 \mid \ldots \mid A_n$, где $n = \lambda(\{a^i \mid i \in \{1, \ldots, k\}\})$.

Теперь рассмотрим грамматику G'' со следующими присвоениями: $a \triangleright r/p \ldots /p \mid A_1 \mid \ldots \mid A_n$, где количество делений на тип p в первом типе равно k. Так как эта грамматика содержит все типы из грамматики, задающей L_k, любое слово из L_k также принадлежит языку данной грамматики. Также заметим, что a^{k+1} принадлежит задаваемому языку, когда первой букве a присваивается тип $r/p/\ldots/p$, а всем остальным — тип p.

Осталось доказать, что никаких других слов данная грамматика породить не может. Для этого достаточно рассмотреть $CF(G'')$ и $CF(G')$. Данные грамматики отличаются только одним правилом, а именно $r \to a\,p \ldots p$, которое, таким образом, обязано быть использовано при выводе нового слова a^t, где $t \geqslant k+1$. Заменим все поддеревья в выводе слова a^t, начинающиеся с указанного выше правила, на дерево вывода слова a^k. Получим дерево вывода, в котором использовались только правила из $CF(G')$, и если правило $r \to a\,p \ldots p$ было использовано не в корне дерева вывода a^t, то получим слово длины больше k,

в выводе которого использовались только правила из $CF(G')$, что невозможно, так как $L(CF(G')) = \{a^i \mid i \in \{1, \ldots, k\}\}$. Итого, правило $r \to a\,p \ldots p$ было использовано в корне дерева вывода a^t. Осталось отметить, что, так как никакая строка нескольких типов в G'' не редуцируется к p, поддерево, соответствующее p, после применения указанного выше правила может состоять только из одного символа a, а это значит, что $t = k + 1$. □

Итого, доказано, что λ — неограниченная функция, так как μ неограничена, и на каждом шаге растёт не более чем на единицу. На языке $\{a, aa\}$ значение λ равно 2. Тем самым, λ принимает все натуральные значения, что завершает доказательство. □

Подводя конец параграфа и главы, отметим, что теорема 1.4.7 является аналогом теоремы 1.1.6 для случая базовых категориальных грамматик. Данный результат не следует из теоремы 1.1.6, так как для преобразования произвольной AB-грамматики к БКГ требуется сперва перейти к эквивалентной контекстно-свободной грамматике. Данное преобразование основано на использовании нормальной формы Грейбах, что приводит к существенному увеличению размера грамматики и нарушению условия присвоения k типов.

Глава вторая. Инварианты и сведение посредством гомоморфизмов

2.1. Свойства редукций

В данной главе нам потребуется набор несложных утверждений для строк типов. В частности, вводится понятие *счётчика* для произвольной строки.

Определение 2.1.1 (ван Бентем [39, гл. 6])**.** Пусть p — примитивный тип. Тогда значением *счётчика по* p для типа C, которое обозначается как $\#_p(C)$, называется целое число, определенное индукцией по структуре типа C следующим образом:

- если C – примитивный тип p, то $\#_p(C) = 1$;
- если C – примитивный тип q, не равный p, то $\#_p(C) = 0$;
- если C имеет вид A/B, то $\#_p(A/B) = \#_p(A) - \#_p(B)$;
- если C имеет вид $B\backslash A$, то $\#_p(B\backslash A) = \#_p(A) - \#_p(B)$.

Определение расширяется на строку типов $A_1; \ldots; A_n$:

$$\#_p(\varepsilon) = 0 \text{ и } \#_p(A_1; \ldots; A_n) = \#_p(A_1) + \ldots + \#_p(A_n).$$

Лемма 2.1.2. *Пусть строка типов u редуцируется к строке типов v. Тогда $\#_p(u) = \#_p(v)$ для любого примитивного типа p.*

Доказательство. Доказательство индукцией по количеству применений правил редукции строки u к строке v.

База индукции: строки u и v совпадают. Следовательно, $\#_p(u) = \#_p(v)$.

Шаг индукции: строка u редуцируется к строке v за n шагов. Рассмотрим первое применение правил редукции в строке u. Тогда $u = \alpha; B/C; C; \beta$ или

$u = \alpha; C; C \backslash B; \beta$ для некоторых типов B, C, а также строк α и β. В обоих случаях, $\#_p(u) = \#_p(\alpha) + (\#_p(B) - \#_p(C)) + \#_p(C) + \#_p(\beta) = \#_p(\alpha) + \#_p(B) + \#_p(\beta)$. По предположению индукции $\#_p(\alpha) + \#_p(B) + \#_p(\beta) = \#_p(\alpha; B; \beta) = \#_p(v)$, а значит в обоих случаях получается $\#_p(u) = \#_p(v)$. □

Как мы уже видели ранее, одна строка типов может редуцироваться к нескольким разным типам. В дальнейшем нам потребуется строить строки типов, которые могут редуцироваться к заранее фиксированным типам.

Счётчики дают необходимое, но не достаточное условие. Например, типы $(p/q)/p$ и $(p/p)/q$ имеют равные значения счётчиков по произвольному примитивному типу, но никакая строка не редуцируется к этим обоим типам. Это утверждение объясняется более тонким инвариантом, учитывающим некоммутативную природу типов, — *интерпретацией в свободной группе*. В работе [40] данное понятие введено для грамматик Ламбека, а также доказаны его основные свойства. На случай AB-грамматик определение переносится напрямую.

Определение 2.1.3. Для произвольного типа C его интерпретацию $[\![C]\!]$ в свободной группе, порождённой множеством примитивных типов, определим индукцией:

- если C – примитивный тип p, то $[\![C]\!] = p$;
- если C имеет вид A/B, то $[\![C]\!] = [\![A]\!][\![B]\!]^{-1}$;
- если C имеет вид $B \backslash A$, то $[\![C]\!] = [\![B]\!]^{-1}[\![A]\!]$.

Определение расширяется на строку типов $A_1; A_2; \ldots; A_n$:

$$[\![A_1; A_2; \ldots; A_n]\!] = [\![A_1]\!][\![A_2]\!] \ldots [\![A_n]\!].$$

Инвариантное свойство для строк типов выражено следующим утверждением.

Лемма 2.1.4. *Если строка типов u редуцируется к строке типов v, то их интерпретации в свободной группе совпадают.*

Доказательство. Доказательство индукцией по количеству применений правил редукции.

База индукции: строки u и v совпадают. Следовательно, $[\![u]\!] = [\![v]\!]$.

Шаг индукции: строка u редуцируется к строке v за n шагов. Рассмотрим первое применение правил редукции в строке u. Тогда $u = \alpha; B/C; C; \beta$ или $u = \alpha; C; C\backslash B; \beta$ для некоторых типов B, C, а также строк α и β. В первом случае $[\![u]\!] = [\![\alpha]\!][\![B]\!][\![C]\!]^{-1}[\![C]\!][\![\beta]\!] = [\![\alpha; B; \beta]\!]$. Во втором аналогично $[\![u]\!] = [\![\alpha]\!][\![C]\!][\![C]\!]^{-1}[\![B]\!][\![\beta]\!] = [\![\alpha; B; \beta]\!]$. По предположению индукции $[\![\alpha; B; \beta]\!] = [\![v]\!]$, а значит в обоих случаях получается $[\![u]\!] = [\![v]\!]$. □

Следовательно, любые два типа, полученные редукцией одной строки, должны иметь одинаковые интерпретации в свободной группе. В частности, не существует строки типов u, которая могла бы редуцироваться к двум различным типам p/q и $q\backslash p$.

Нам потребуется дополнительная терминология. Для любого типа $B\backslash A$ или A/B, тип A называется *числителем*, а B — *знаменателем*. Для типа C множества всех типов, встречающихся в нём в качестве числителей и знаменателей, обозначаются следующим образом.

Определение 2.1.5. Для произвольного типа C множество его числителей $num(C)$ и множество его знаменателей $den(C)$ определяются индукцией:

- для примитивного типа: $num(p) = \varnothing$ и $den(p) = \varnothing$;
- для составного типа:

$$num(B\backslash A) = num(A/B) = \{A\} \cup num(A),$$
$$den(B\backslash A) = den(A/B) = \{B\} \cup den(A).$$

Определение расширяется на строку типов $A_1; \ldots; A_n$:

- $num(A_1; \ldots; A_n) = \bigcup_{i=1}^{n} num(A_i)$;

- $den(A_1; \ldots; A_n) = \bigcup_{i=1}^{n} den(A_i)$.

Можно сформулировать следующее очень простое, но полезное свойство.

Предложение 2.1.6. *Если строка из нескольких типов w редуцируется к единственному типу A, то хотя бы один из типов в этой строке имеет A в множестве своих числителей.*

Доказательство. Доказательство проводится индукцией по длине строки.

База индукции: строка w состоит из двух типов $B; C$. Тогда для применения правила редукции возможен один из двух случаев $B = A/C$ или $C = B\backslash A$. В первом случае $A \in num(A) \cup \{A\} = num(B)$, во втором случае $A \in num(A) \cup \{A\} = num(C)$.

Шаг индукции: строка w состоит из трёх и более типов. Рассмотрим первый шаг редукции w к типу A. Таким образом, w имело вид $\alpha; B; B\backslash C; \beta$ или $\alpha; C/B; B; \beta$ и на первом шаге редуцировалась к строке $\widetilde{w} = \alpha; C; \beta$. Следовательно:

$$num(w) = num(\alpha) \cup num(B) \cup num(B\backslash C) \cup num(\beta),$$
$$num(\widetilde{w}) = num(\alpha) \cup num(C) \cup num(\beta).$$

Откуда следует, что $num(\widetilde{w}) \subseteq num(w)$. Поскольку $A \in num(\widetilde{w})$ по предположению индукции, то $A \in num(w)$. □

Также для строк типов верно следующее техническое свойство.

Предложение 2.1.7. *Пусть A, B, C и D — произвольные типы, а α, β и γ — произвольные строки типов. Пусть $w = \alpha; A/C; \beta; D\backslash B; \gamma$, и предположим, что:*

- $A/C \notin den(w)$;
- $D\backslash B \notin den(w)$;

- *никакой собственный суффикс β не редуцируется к D;*
- *никакой собственный префикс β не редуцируется к C.*

Тогда, если w редуцируется к единственному типу E, то хотя бы одна из строк $\alpha; A/C; C; D\backslash B; \gamma$ или $\alpha; A/C; D; D\backslash B; \gamma$ также редуцируется к E.

Доказательство. Рассмотрим шаг в редукции w к E, когда один из типов A/C или $D\backslash B$ впервые участвует в редукции. Поскольку ни один из этих типов не встречается в знаменателях, либо A/C редуцируется с C, либо D редуцируется с $D\backslash B$.

Если первой была редукция $A/C; C$ к A, то C получен из префикса $\beta; D\backslash B; \gamma$. Так как тип $D\backslash B$ ещё не участвовал в редукциях, тип C должен был быть получен из префикса β. По предположению, этим префиксом не может быть собственный префикс β, значит, весь β должен был быть редуцирован к C. Если какие-либо шаги редукции затрагивали γ или α, их можно отложить до завершения редукции β к C. В результате редукция начинается с редукции $\alpha; A/C; \beta; D\backslash B; \gamma$ к $\alpha; A/C; C; D\backslash B; \gamma$, которая затем редуцируется к E, как и утверждалось.

Случай редукции $D; D\backslash B$ к B, где D получена из суффикса $\alpha; A/C; \beta$, симметричен: A/C остаётся нетронутым, следовательно, D получен из суффикса β, который по предположению не может быть собственным. Таким образом, β редуцируется к D, и редукция $\alpha; A/C; \beta; D\backslash B; \gamma$ к E проходит через промежуточный этап $\alpha; A/C; D; D\backslash B; \gamma$. □

2.2. Незамкнутость класса $\mathcal{G}^1$

В данном параграфе рассматриваются грамматики класса $\mathcal{G}^1$. Принципиальным свойством, отличающим данные грамматики, является детерминированность выбора типов для произвольной строки букв.

Лемма 2.2.1. *Пусть G — произвольная грамматика из класса $\mathcal{G}^1$, а w — произвольное непустое слово. Тогда $w^k \notin L(G)$, для любого $k \geqslant 2$.*

Доказательство. Предположим противное. Тогда для некоторого непустого слова $w = a_1 \dots a_n$ и некоторого $k \geqslant 2$ слово $(a_1 \dots a_n)^k$ лежит в $L(G)$. Пусть в грамматике G символу a_i присвоен тип A_i. Тогда строка типов α, равная $A_1; \dots; A_n; A_1; \dots; A_n; \dots; A_1; \dots; A_n$, редуцируется к типу r. По лемме 2.1.2 значение счётчика по примитивному типу r для указанной строки типов равно $\#_r(r) = 1$. Итого, $\#_r(\alpha) = k(\#_r(A_1) + \dots + \#_r(A_n)) = 1$, что невозможно, так как 1 не делится на k. □

Никакая базовая категориальная грамматика не может породить пустое слово ε. Следовательно, если язык содержит пустое слово, данный язык не может задаваться БКГ. Значит, очевидно, что для произвольного языка L его *дополнение* $\Sigma^* - L$ не задаётся никакой БКГ. Следовательно, для класса языков задаваемых грамматиками из $\mathcal{G}^1$ естественно рассматривать дополнение языков до Σ^+, а не до Σ^*. Аналогично *итерация Клини* — L^* произвольного языка не задаётся никакой БКГ, а следовательно будем рассматривать *плюс Клини* для данного класса.

Оказывается, что дополнение до Σ^+ языка, задаваемого грамматикой из $\mathcal{G}^1$, никогда не может задаваться грамматикой из $\mathcal{G}^1$.

Теорема 2.2.2. *Для произвольного языка L из класса $\mathcal{L}^1$ язык $\Sigma^+ - L$ не принадлежит классу $\mathcal{L}^1$.*

Доказательство. Рассмотрим произвольное слово w из языка Σ^+. По лемме 2.2.1 слово ww не принадлежат языку $L(G)$. Следовательно, слово ww лежит в языке $\Sigma^+ - L(G)$, а значит по лемме 2.2.1 данный язык не задаётся никакой грамматикой из класса $\mathcal{G}^1$. □

Теорема 2.2.3. *Если язык L содержит хотя бы одно непустое слово, то язык L^+ не принадлежит классу $\mathcal{L}^1$.*

Доказательство. Пусть $w \in L$, тогда $ww \in L^+$. Значит по лемме 2.2.1 язык L^+ не задаётся никакой грамматикой из класса $\mathcal{G}^1$. □

Теорема 2.2.4. *Класс языков $\mathcal{L}^1$ не замкнут относительно конкатенации.*

Доказательство. Рассмотрим язык $L = \{a\}$, заданный грамматикой $a \triangleright r$. Тогда язык $L \cdot L = \{a^2\}$ не принадлежит классу $\mathcal{L}^1$ по лемме 2.2.1. □

Теорема 2.2.5. *Класс языков $\mathcal{L}^1$ не замкнут относительно объединения.*

Доказательство. Рассмотрим языки $L_1 = \{a\}$ и $L_2 = \{ab\}$. Язык L_1 задаётся грамматикой $a \triangleright r$, а L_2 грамматикой $a \triangleright r$, $b \triangleright (r \backslash r)$. Предположим, что $L_1 \cup L_2 = \{a, ab\}$ задаётся некоторой грамматикой G из класса $\mathcal{G}^1$. Тогда в силу того, что слово a порождается грамматикой, символу a в грамматике G должен быть присвоен примитивный тип r. Поскольку слово ab порождается грамматикой, символу b должен быть присвоен тип $r \backslash r$. Но так как строка типов $r; r \backslash r; r \backslash r$ редуцируется к типу r, то слово abb должно порождаться грамматикой G, что не так. □

Теорема 2.2.6. *Класс языков $\mathcal{L}^1$ не замкнут относительно пересечения.*

Доказательство. Примером будут являться языки $L_1 = \{a^n d b^n e c^m \mid n, m \geqslant 0\}$ и $L_2 = \{a^m d b^n e c^n \mid n, m \geqslant 0\}$. Рассмотрим грамматику G_1 из класса $\mathcal{G}^1$ с набором присвоений $a \triangleright (p/q)/p$, $b \triangleright q$, $c \triangleright r \backslash r$, $d \triangleright p$, $e \triangleright p \backslash r$. По теореме 1.3.3

данная грамматика эквивалентна контекстно-свободной грамматике со следующим набором правил:

$$\{p \to a\,p\,q\} \cup \{q \to b\} \cup \{r \to r\,c\} \cup \{p \to d\} \cup \{r \to p\,e\}.$$

Эквивалентная грамматика имеет правила:

$$\{p \to a\,p\,b\} \cup \{r \to r\,c\} \cup \{p \to d\} \cup \{r \to p\,e\}$$

и задаёт язык L_1.

Аналогично грамматика G_2 с набором присвоений $a \triangleright r/r$, $b \triangleright p/q$, $c \triangleright p\backslash q$, $d \triangleright r/p$, $e \triangleright p$ задаёт язык L_2.

Пересечение языков L_1 и L_2 — язык $\{a^n d b^n e c^n \mid n \geqslant 0\}$, который не является контекстно-свободным, а значит тем более не задаётся грамматикой из класса $\mathcal{G}^1$. □

Тем самым было доказанно, что класс $\mathcal{L}^1$ не замнут относительно большинства естественных операций на языках, что является одним из основных результатов данной диссертации.

Однако, класс $\mathcal{L}^1$ замнут относительно операции *обращения* языка. Прямым следствием теоремы 1.3.3 является следующее несложное утверждение, где под L^R понимается *обращение* языка L, то есть язык $\{a_n \dots a_1 \mid a_1 \dots a_n \in L\}$.

Теорема 2.2.7. *Класс языков $\mathcal{L}^1$ замкнут относительно обращения.*

Доказательство. По теореме 1.3.3 преобразуем произвольную грамматику G из класса $\mathcal{G}^1$, задающую язык L, в контекстно-свободную грамматику $CF(G)$. Перевернём все правила в грамматике $CF(G)$, а затем перейдём к эквивалентной грамматике из класса $\mathcal{G}^1$, которая задаёт язык L^R. □

2.3. Сведение к AB-грамматикам с однозначным присвоением типов посредством гомоморфизмов

Пусть $G = (\Sigma, N, S, R)$ — контекстно-свободная грамматика в **нормальной форме Грейбах**, где все правила имеют вид $X \to a$, $X \to aY$, $X \to aYZ$, где $X, Y, Z \in N$ и $a \in \Sigma$.

Для каждого символа $a \in \Sigma$ рассмотрим все правила, содержащие a в правой части. В конструкции, представленной в теореме 1.3.3, эквивалентная AB-грамматика имеет следующие присвоения:

- правило $X \to a$ соответствует присвоению $a \triangleright X$,
- правило $X \to aY$ соответствует присвоению $a \triangleright X/Y$,
- правило $X \to aYZ$ соответствует присвоению $a \triangleright (X/Z)/Y$.

Таким образом, в эквивалентной AB-грамматике одному символу a может быть присвоено несколько типов.

Текущей целью является построение единой строки типов, которая могла бы редуцироваться к каждому типу присвоенному символу a.

Однако по лемме 2.1.4 данное невозможно. Например, не существует строки типов, которая редуцируется как к примитивному типу p, так и к примитивному типу t. С другой стороны, интерпретация в свободной группе не препятствует построению строки типов, которая редуцируется к p/p и к t/t. Пример такой строки приведён ниже.

Пример 8. Пусть p и t — произвольные примитивные типы. Тогда строка типов $(p/p)/(t/t); t/t; t; t\backslash p; p\backslash(t/t)$ редуцируется к p/p, а также к t/t.

$$\dfrac{(p/p)/(t/t) \quad \dfrac{t/t \quad t}{t} \quad \dfrac{t\backslash p}{p} \quad \dfrac{p\backslash(t/t)}{t/t}}{p/p} \qquad \dfrac{\dfrac{(p/p)/(t/t) \quad t/t}{p/p} \quad \dfrac{t \quad t\backslash p}{p}}{p} \quad \dfrac{p\backslash(t/t)}{t/t}$$

Итак, обойдём инварианты, основанные на интерпретации в свободной группе, заменой каждого примитивного типа p на p/p. Например, тип $p/(s/r)$ преобразуем в $(p/p)/((s/s)/(r/r))$. Введём следующее обозначение.

Определение 2.3.1. Определим функцию ϕ как отображение из множества типов во множество типов по индукции:

- если C – примитивный тип p, то $\phi(C) = p/p$;
- если C имеет вид A/B, то $\phi(C) = \phi(A)/\phi(B)$;
- если C имеет вид $B\backslash A$, то $\phi(C) = \phi(B)\backslash\phi(A)$.

Теперь наша цель выглядит следующим образом: для любых фиксированных типов $A_1, \ldots, A_n$ необходимо построить строку типов, которая редуцируется ко всем типам $\phi(A_1), \ldots, \phi(A_n)$. Более того, достаточно это сделать для типов трёх видов: p, p/q и $(p/q)/s$, так как именно такие типы возникают при преобразовании контекстно-свободной грамматики в нормальной форме Грейбах в AB-грамматику.

Сперва разберёмся со случаем двух типов.

Лемма 2.3.2. *Пусть A и B — различные типы вида p, p/q или $(p/q)/s$. Тогда существует строка типов $z_{A,B}$, такая что:*

I. $z_{A,B}$ редуцируется к $\phi(A)$ и к $\phi(B)$;

II. никакой собственный суффикс $z_{A,B}$ не редуцируется к $\phi(A)$, и никакой собственный префикс $z_{A,B}$ не редуцируется к $\phi(B)$.

Доказательство. Требуемая строка типов $z_{A,B}$ строится в несколько этапов.

Сперва определим $x_{A,t}$ (где t — новый примитивный тип, то есть не встречающийся в типе A), которая редуцируется как к $\phi(A)$, так и к t/t. Поскольку A может иметь три возможных формы, $x_{A,t}$ определяется по-разному для каждой из них.

$$x_{p,t} = (p/p)/(t/t); t/t; t; t\backslash p; p\backslash(t/t)$$
$$x_{p/q,\,t} = ((p/p)/(q/q))/(t/t); t/t; t; t\backslash(q/q); q; q\backslash p; p\backslash(t/t)$$
$$x_{(p/q)/s,\,t} = (((p/p)/(q/q))/(s/s))/(t/t); t/t; t; t\backslash(s/s); s; s\backslash(q/q); q; q\backslash p; p\backslash(t/t)$$

Утверждение 2.3.3. *Для любого типа A строка $x_{A,t}$ редуцируется как к $\phi(A)$, так и к $\phi(t)$.*

Доказательство. Редукции для строки $x_{p,t}$ в точности совпадают с примером 8.

Приведём редукции для $x_{p/q,\,t}$:

$$\dfrac{((p/p)/(q/q))/(t/t) \quad \dfrac{t/t \quad t}{t} \quad \dfrac{t\backslash(q/q)}{q/q} \quad \dfrac{q}{q} \quad \dfrac{q\backslash p}{p} \quad \dfrac{p\backslash(t/t)}{t/t}}{(p/p)/(q/q)}$$

$$\dfrac{\dfrac{((p/p)/(q/q))/(t/t) \quad t/t}{(p/p)/(q/q)} \quad \dfrac{t \quad t\backslash(q/q)}{q/q} \quad \dfrac{q \quad q\backslash p}{p} \quad p\backslash(t/t)}{t/t}$$

Редукции для $x_{(p/q)/s,\,t}$ выглядят следующим образом:

$$\dfrac{(((p/p)/(q/q))/(s/s))/(t/t) \quad \dfrac{t/t \quad t}{t} \quad \dfrac{t\backslash(s/s)}{s/s} \quad \dfrac{s}{s} \quad \dfrac{s\backslash(q/q)}{q/q} \quad \dfrac{q}{q} \quad \dfrac{q\backslash p}{p} \quad \dfrac{p\backslash(t/t)}{t/t}}{((p/p)/(q/q))/(s/s)}$$

$$\dfrac{\dfrac{(((p/p)/(q/q))/(s/s))/(t/t) \quad t/t}{((p/p)/(q/q))/(s/s)} \quad \dfrac{t \quad t\backslash(s/s)}{s/s} \quad \dfrac{s \quad s\backslash(q/q)}{q/q} \quad \dfrac{q \quad q\backslash p}{p} \quad p\backslash(t/t)}{t/t}$$

□

Теперь определим другую строку $y_{t,B}$, которая редуцируется к t/t и $\phi(B)$. Хотя требуемое свойство точно соответствует свойству $x_{A,t}$, сама строка отличается от $x_{A,t}$. Это отличие понадобится позже для выполнения условия II леммы 2.3.2. Определим $y_{t,B}$ следующим образом.

$$
\begin{aligned}
y_{t,p} &= (t/t)/(p/p); p/p; p; p\backslash t; t\backslash(p/p) \\
y_{t,\,p/q} &= (t/t)/((p/p)/(q/q)); (p/p)/(q/q); q/q; q; q\backslash p; p\backslash t; t\backslash((p/p)/(q/q)) \\
y_{t,\,(p/q)/s} &= (t/t)/(((p/p)/(q/q))/(s/s)); ((p/p)/(q/q))/(s/s); s/s; s; \\
&\qquad s\backslash(q/q); q; q\backslash p; p\backslash t; t\backslash(((p/p)/(q/q))/(s/s))
\end{aligned}
$$

Утверждение 2.3.4. *Для любого типа B строка $y_{t,B}$ редуцируется как к $\phi(t)$, так и к $\phi(B)$.*

Доказательство. Приведём редукции для $y_{t,p}$:

$$
\dfrac{\dfrac{(t/t)/(p/p) \qquad \dfrac{\dfrac{p/p \quad p}{p} \quad p\backslash t}{t}}{t/t} \quad t\backslash(p/p)}{p/p}
$$

$$
\dfrac{\dfrac{\dfrac{(t/t)/(p/p) \quad p/p}{t/t} \quad \dfrac{p \quad p\backslash t}{t}}{t} \quad t\backslash(p/p)}{p/p}
$$

Редукции для $y_{t,\,p/q}$:

$$
\dfrac{\dfrac{(t/t)/((p/p)/(q/q)) \qquad \dfrac{\dfrac{\dfrac{(p/p)/(q/q) \quad q/q}{p/p} \quad \dfrac{q \quad q\backslash p}{p}}{p} \quad p\backslash t}{t}}{t/t} \quad t\backslash((p/p)/(q/q))}{(p/p)/(q/q)}
$$

$$\dfrac{\dfrac{(t/t)/((p/p)/(q/q)) \quad (p/p)/(q/q)}{t/t} \quad \dfrac{\dfrac{\dfrac{q/q \quad q}{q} \quad q\backslash p}{p} \quad p\backslash t}{t}}{\dfrac{t \qquad\qquad t\backslash((p/p)/(q/q))}{(p/p)/(q/q)}}$$

Строка $y_{t,\,(p/q)/s}$ слишком длинная, чтобы сразу выписывать её редукции. Сперва рассмотрим её подстроку.

$$u = ((p/p)/(q/q))/(s/s); s/s; s; s\backslash(q/q); q; q\backslash p; p\backslash t$$

Тогда строка $y_{t,\,(p/q)/s}$ имеет следующий вид.

$$(t/t)/(((p/p)/(q/q))/(s/s)); u; t\backslash(((p/p)/(q/q))/(s/s))$$

Строка u редуцируется к строке $((p/p)/(q/q))/(s/s); t$, а также к типу t:

$$\dfrac{\dfrac{\dfrac{((p/p)/(q/q))/(s/s) \quad s/s}{(p/p)/(q/q)} \quad \dfrac{s \quad s\backslash(q/q)}{q/q}}{p/p} \quad \dfrac{q \quad q\backslash p}{p}}{\dfrac{p \qquad\qquad p\backslash t}{t}}$$

$$((p/p)/(q/q))/(s/s) \quad \dfrac{\dfrac{\dfrac{\dfrac{s/s \quad s}{s} \quad s\backslash(q/q)}{q/q} \quad q}{q} \quad q\backslash p}{p} \quad \dfrac{p\backslash t}{t}$$

Значит, $y_{t,(p/q)/s}$ редуцируется к следующим строкам.

$$(t/t)/(((p/p)/(q/q))/(s/s)); t; t\backslash(((p/p)/(q/q))/(s/s))$$

$$(t/t)/(((p/p)/(q/q))/(s/s)); (((p/p)/(q/q))/(s/s); t; t\backslash(((p/p)/(q/q))/(s/s))$$

Первая строка редуцируется к t/t, а вторая к $((p/p)/(q/q))/(s/s)$. □

Требуемая строка типов $z_{A,B}$ является объединением $x_{A,t}$ и $y_{t,B}$, чтобы редуцироваться как к $\phi(A)$, так и к $\phi(B)$.

Пусть k_1, k_2, k_3 и k_4 — четыре новых примитивных типа. Определим строку типов $z_{A,B}$ следующим образом:

$$z_{A,B} = \phi(A)/k_1; k_1/(t/t); x_{A,t}; \phi(A)\backslash k_2; k_2\backslash((t/t)/k_3); k_3/\phi(B); y_{t,B}; \\ (t/t)\backslash k_4; k_4\backslash\phi(B)$$

Обозначим следующие подстроки $z_{A,B}$ через u и v (не путать со строкой u из доказательства утверждения 2.3.4):

$$u = \phi(A)/k_1; k_1/(t/t); x_{A,t}; \phi(A)\backslash k_2; k_2\backslash((t/t)/k_3); k_3/\phi(B); y_{t,B}$$
$$v = x_{A,t}; \phi(A)\backslash k_2; k_2\backslash((t/t)/k_3); k_3/\phi(B); y_{t,B}; (t/t)\backslash k_4; k_4\backslash\phi(B)$$

Тогда строку $z_{A,B}$ можно записать двумя образами:

$$z_{A,B} = u; (t/t)\backslash k_4; k_4\backslash\phi(B)$$
$$z_{A,B} = \phi(A)/k_1; k_1/(t/t); v$$

Следующие редукции показывают, что обе строки u и v редуцируются к t/t.

$$\dfrac{\dfrac{\dfrac{x_{A,t}}{\phi(A)} \quad \phi(A)\backslash k_2}{k_2} \quad k_2\backslash((t/t)/k_3)}{(t/t)/k_3} \quad \dfrac{k_3/\phi(B) \quad \dfrac{\dfrac{y_{t,B}}{t/t} \quad (t/t)\backslash k_4}{k_4} \quad k_4\backslash\phi(B)}{\phi(B)}}{k_3}}{t/t}$$

$$\dfrac{\dfrac{\dfrac{\phi(A)/k_1 \quad \dfrac{k_1/(t/t) \quad \dfrac{x_{A,t}}{t/t}}{k_1}}{\phi(A)} \quad \phi(A)\backslash k_2}{k_2} \quad k_2\backslash((t/t)/k_3)}{(t/t)/k_3} \quad \dfrac{k_3/\phi(B) \quad \dfrac{y_{t,B}}{\phi(B)}}{k_3}}{t/t}$$

Таким образом, строка $z_{A,B}$ редуцируется к $t/t; (t/t)\backslash k_4; k_4\backslash\phi(B)$, которая в свою очередь редуцируется к $\phi(B)$. В то же самое время строка $z_{A,B}$ редуцируется к $\phi(A)/k_1; k_1/(t/t); t/t$, которая редуцируется к $\phi(A)$.

Перейдём к доказательству части II леммы 2.3.2. Требуется доказать, что собственные префиксы $z_{A,B}$ не редуцируются к $\phi(B)$, а собственные суффиксы $z_{A,B}$ не редуцируются к $\phi(A)$. Доказательство проводится разбором всех собственных префиксов и суффиксов. Для префиксов и суффиксов, которые не разделяют $x_{A,t}$ и $y_{t,B}$ (т.е. содержат либо всю подстроку, либо ни одного из её типов), требуемое будет получено с использованием предложения 2.1.7. В ином случае доказательство опирается на следующие свойства $x_{A,t}$ и $y_{t,B}$.

Утверждение 2.3.5. *Для любого типа A:*

- *никакой собственный суффикс $x_{A,t}$ не редуцируется к $\phi(A)$;*
- *никакой собственный префикс $x_{A,t}$ не редуцируется к (t/t).*

Доказательство. В строке $x_{A,t}$ тип t/t присутствует во множестве числителей только последнего типа. Следовательно, никакой собственный префикс не может редуцироваться к t/t по предложению 2.1.6.

Также тип $\phi(A)$ в $x_{A,t}$ присутствует во множестве числителей только первого типа. Таким образом, никакой собственный суффикс не может быть редуцирован к $\phi(A)$ по предложению 2.1.6. □

Утверждение 2.3.6. *Для любого типа B:*

- *никакой собственный суффикс $y_{t,B}$ не редуцируется к (t/t);*
- *никакой собственный префикс $y_{t,B}$ не редуцируется к $\phi(B)$.*

Доказательство. В строке $y_{t,B}$ тип $\phi(B)$ присутствует в числителе только последнего типа. Следовательно, никакой собственный префикс не редуцируется к $\phi(B)$ по предложению 2.1.6.

Аналогично тип t/t присутствует в числителе только в первом типе $y_{t,B}$, поэтому никакой собственный суффикс не может быть редуцирован к t/t по предложению 2.1.6. □

Перейдём к рассмотрению строки $z_{A,B}$. Доказательство проводится отдельно для случая суффикса и префикса.

Случай суффикса. Сначала рассмотрим любой собственный суффикс $z_{A,B}$, который начинается не на границах $x_{A,t}$ и $y_{t,B}$ и не внутри любой из этих подстрок. Таким образом, суффикс может начинаться в одном из следующих четырёх мест.

$$\phi(A)/k_1 \overset{\downarrow}{;} k_1/(t/t); x_{A,t}; \phi(A)\backslash k_2 \overset{\downarrow}{;} k_2\backslash((t/t)/k_3) \overset{\downarrow}{;} k_3/\phi(B);$$
$$y_{t,B}; (t/t)\backslash k_4 \overset{\downarrow}{;} k_4\backslash\phi(B)$$

По обе стороны каждой из этих четырех стрелок находятся типы, содержащие новый примитивный тип (k_1, k_2, k_3 и k_4 соответственно). Если данные типы разделены, то суффикс, содержащий только один из них, не может быть редуцирован к $\phi(A)$, что делает эти случаи невозможными. Другим вариантом доказательства является рассмотрение интерпретации в свободной группе каждого суффикса при условии того, что для интерпретаций выполнено $[\![x_{A,t}]\!] = [\![y_{t,B}]\!]$, и равны единице свободной группы. Следовательно, по лемме 2.1.4, никакая из данных строк не может редуцироваться к $\phi(A)$, так как $[\![\phi(A)]\!]$ равна единице.

Следующий вариант состоит в том, что суффикс начинается сразу после $x_{A,t}$ или $y_{t,B}$.

$$\phi(A)/k_1; k_1/(t/t); x_{A,t} \overset{\downarrow}{;} \phi(A)\backslash k_2; k_2\backslash((t/t)/k_3); k_3/\phi(B); y_{t,B} \overset{\downarrow}{;} (t/t)\backslash k_4; k_4\backslash\phi(B)$$

В обоих случаях первый тип в суффиксе имеет деление слева и, следовательно, может редуцироваться только с некоторым большим типом справа. Однако ни $\phi(A)\backslash k_2$ ни $(t\backslash t)\backslash k_4$ не присутствуют во множестве знаменателей всей строки. Значит, данные типы не могут учавствовать ни в какой редукции.

Рассмотрим случай, когда суффикс начинается **внутри** $x_{A,t}$. Предположим, что некоторая строка такого вида редуцируется к единственному типу. Тогда на каком-то этапе редукции должен был быть получен примитивный тип k_2. Но чтобы получить k_2, суффикс $x_{A,t}$ должен был редуцироваться к $\phi(A)$, что невозможно согласно утверждению 2.3.5.

Пусть суффикс начнется **сразу перед** $x_{A,t}$. Предположим, что данная строка редуцируется к $\phi(A)$. Как и в предыдущем случае, на одном из этапов редукции должен был быть получен тип k_2, а значит подстрока $x_{A,t}$ редуцировалась к $\phi(A)$. А затем $\phi(A); \phi(A)\backslash k_2$ редуцировалась к k_2. Итого, следующая строка должна редуцироваться к $\phi(A)$.

$$k_2; k_2\backslash((t/t)/k_3); k_3/\phi(B); y_{t,B}; (t/t)\backslash k_4; k_4\backslash\phi(B)$$

Хотя $\phi(A)$ не присутствует в указанной строке, возможность её редукции к $\phi(A)$ нельзя исключить, воспользовавшись предложением 2.1.6, поскольку A может быть частью типа B.

Чтобы доказать невозможность такой редукции воспользуемся предложением 2.1.7 и свойствами $y_{t,B}$. Согласно утверждению 2.3.6:

- никакой собственный суффикс $y_{t,B}$ не редуцируется к t/t;
- никакой собственный префикс $y_{t,B}$ не редуцируется к $\phi(B)$.

Тогда, применяя предложение 2.1.7 где $A/C = k_3/\phi(B)$, $\beta = y_{t,B}$ и $D\backslash B = (t/t)\backslash k_4$, получаем, что если данная строка редуцируется к $\phi(A)$, то одна из следующих двух строк также должна редуцироваться к $\phi(A)$:

$$k_2; k_2\backslash((t/t)/k_3); k_3/\phi(B); \phi(B); (t/t)\backslash k_4; k_4\backslash\phi(B)$$

$$k_2; k_2\backslash((t/t)/k_3); k_3/\phi(B); t/t; (t/t)\backslash k_4; k_4\backslash\phi(B)$$

Однако первая строка редуцируется только к $\phi(B)$, а вторая — только к t/t.

Если суффикс начинается **непосредственно перед** $y_{t,B}$ и редуцируется к $\phi(A)$, то в процессе редукций должен был быть получен тип k_4, а значит, подстрока $y_{t,B}$ должна была быть редуцирована к t/t. Затем t/t и $(t/t)\backslash k_4$ редуцированны к k_4, и, наконец, k_4 и $k_4\backslash\phi(B)$ — к $\phi(B)$. Таким образом, единственный возможный тип, к которому данная строка редуцируется, — $\phi(B)$.

Наконец, пусть суффикс начинается **внутри** $y_{t,B}$, и предположим, что вся строка редуцируется к одному типу. Тогда при редукции должен был быть получен тип k_4, но никакой собственный суффикс $y_{t,B}$ не редуцируется к t/t. Тем самым доказательство для случая суффикса завершено.

Случай префикса: Доказательство аналогично, но не полностью симметрично случаю суффикса.

Снова выбранный префикс не может заканчиваться на границах $x_{A,t}$ и $y_{t,B}$, так как это разделило бы новые примитивные типы.

Более того, выбранный префикс не может заканчиваться непосредственно перед $x_{A,t}$ и $y_{t,B}$, поскольку последние типы таких строк не могут участвовать в дальнейших редукциях.

Кроме того, выбранный префикс не может заканчиваться внутри $x_{A,t}$, так как в таких строках тип $k_1/(t/t)$ не может быть редуцирован. А также не может заканчиваться внутри $y_{t,B}$, так как в таких строках тип $k_3/\phi(B)$ не может быть редуцирован.

Таким образом, префикс может заканчиваться либо непосредственно после $x_{A,t}$, либо непосредственно после $y_{t,B}$.

Первый случай легко разбирается, поскольку для редукции типа $k_1/(t/t)$ строка $x_{A,t}$ должна была быть редуцирована к t/t. Следовательно, такой префикс редуцируется только к $\phi(A)$.

Теперь предположим, что префикс заканчивается непосредственно после $y_{t,B}$. Следовательно, $y_{t,B}$ должна была быть редуцирована к $\phi(B)$ для получе-

ния k_3. И следующая строка должна быть редуцирована к $\phi(B)$.

$$\phi(A)/k_1; k_1/(t/t); x_{A,t}; \phi(A)\backslash k_2; k_2\backslash((t/t)/k_3); k_3$$

По утверждению 2.3.5, никакой собственный префикс $x_{A,t}$ не редуцируется к t/t, и никакой собственный суффикс $x_{t,B}$ не редуцируется к $\phi(A)$. Тогда, по предложению 2.1.7, где $A/C = k_1/(t/t)$, $\beta = x_{A,t}$ и $C\backslash D = \phi(A)\backslash k_2$, одна из следующих строк должна быть редуцирована к $\phi(B)$:

$$\phi(A)/k_1; k_1/(t/t); t/t; \phi(A)\backslash k_2; k_2\backslash((t/t)/k_3); k_3$$
$$\phi(A)/k_1; k_1/(t/t); \phi(A); \phi(A)\backslash k_2; k_2\backslash((t/t)/k_3); k_3$$

Однако это не так: первая строка редуцируется только к t/t, а вторая — только к $\phi(A)$. □

Подстрока $z_{A,B}$, которая редуцируется к $\phi(A)$ и $\phi(B)$, имитирует присвоение двух типов — A и B. Теперь общая идея заключается в использовании этой подстроки вместо символа, когда этому символу требуется присвоить два типа A и B. Таким образом, строка исходных символов заменяется строкой, составленной из подстрок вида $z_{A,B}$. Однако, свойства $z_{A,B}$ не гарантируют, что не возникнет редукций между соседними подстроками $z_{A,B}$ и $z_{C,D}$, в результате чего $y_{t,B}$ из первой подстроки может редуцироваться с $x_{C,t}$ из второй подстроки, что приведёт к непредсказуемым результатам.

Чтобы предотвратить это, строка $z_{A,B}$ будет расширена путём обёртывания новыми примитивными типами. Это наложит ограничение на возможный порядок редукций и, в конечном итоге, устранит указанную возможность. Первый шаг — применение деления на новые *сторожевые типы* ℓ и r с обеих сторон каждой строки, представляющей два типа A и B. Тем самым задаётся строка $u_{A,B}$, которая редуцируется к $\ell\backslash(\phi(A)/r)$ и $\ell\backslash(\phi(B)/r)$.

Обозначим через $\psi(A)$ тип $\phi(A)$, делённый на r справа, а затем на ℓ слева.

Определение 2.3.7. Для произвольного типа A положим значение $\psi(A)$ равным $\ell\backslash(\phi(A)/r)$.

Теперь план заключается в том, чтобы заменить ϕ-кодирование типов на новое ψ-кодирование. Оно имеет аналогичные свойства, что и ранее доказанные для ϕ-кодирования.

Лемма 2.3.8. *Пусть A и B — различные типы вида p, p/q или $(p/q)/s$. Тогда существует строка типов $u_{A,B}$ такая, что:*

I. $u_{A,B}$ редуцируется к $\psi(A)$ и к $\psi(B)$;

II. никакой собственный суффикс $u_{A,B}$ не редуцируется к $\psi(A)$, и никакой собственный префикс $u_{A,B}$ не редуцируется к $\psi(B)$.

Доказательство. Возьмём новые примитивные типы o_1, o_2, o_3, o_4, и построим финальную строку в два этапа. Сперва применяем деления на r справа и получаем следующую строку:

$$z'_{A,B} = (\phi(A)/r)/(\phi(B)/r); \phi(A)/o_1; o_1/\phi(B); z_{A,B}; \\ \phi(A)\backslash o_2; o_2\backslash(\phi(B)/r); r; \phi(A)\backslash(\phi(B)/r),$$

деление на ℓ будет добавлено позднее.

Утверждение 2.3.9. *Для произвольных двух различных типов A и B строка $z'_{A,B}$ редуцируется как к $\phi(A)/r$, так и к $\phi(B)/r$.*

Доказательство. Обозначим через u и v следующие подстроки строки $z'_{A,B}$:

$$u = \phi(A)/o_1; o_1/\phi(B); z_{A,B}; \phi(A)\backslash o_2; o_2\backslash(\phi(B)/r); r; \phi(A)\backslash(\phi(B)/r)$$
$$v = (\phi(A)/r)/(\phi(B)/r); \phi(A)/o_1; o_1/\phi(B); z_{A,B}; \phi(A)\backslash o_2; o_2\backslash(\phi(B)/r); r.$$

Таким образом, строку $z'_{A,B}$ можно записать, как $z'_{A,B} = (\phi(A)/r)/(\phi(B)/r); u$, и как $z'_{A,B} = v; \phi(A)\backslash(\phi(B)/r)$. Следующие редукции показывают, что u реду-

цируется к $\phi(B)/r$, а v редуцируется к $\phi(A)$.

$$\dfrac{\dfrac{\phi(A)/o_1 \quad \dfrac{o_1/\phi(B) \quad \dfrac{\dfrac{z_{A,B}}{\phi(A)} \quad \phi(A)\backslash o_2}{o_2}}{o_1}}{\phi(A)} \quad \phi(A)\backslash(\phi(B)/r)}{\phi(B)/r}$$

$$\dfrac{\dfrac{(\phi(A)/r)/(\phi(B)/r) \quad \dfrac{\phi(A)/o_1 \quad \dfrac{o_1/\phi(B) \quad \dfrac{z_{A,B}}{\phi(B)}}{o_1}}{\phi(A)} \quad \dfrac{\phi(A)\backslash o_2}{o_2}}{\phi(B)/r} \quad o_2\backslash(\phi(B)/r)}{\phi(A)/r} \quad r}{\phi(A)}$$

Таким образом, $z'_{A,B}$ редуцируется к $\phi(A)/r$ и к $\phi(B)/r$. □

Утверждение 2.3.10. *Для произвольных различных двух типов A и B никакой собственный суффикс $z'_{A,B}$ не редуцируется к $\phi(A)/r$, и никакой собственный префикс $z'_{A,B}$ не редуцируется к $\phi(B)/r$.*

Доказательство. **Случай суффикса.** Тип $\phi(A)/r$ присутствует в числителе только первого типа строки $z'_{A,B}$. Следовательно, согласно предложению 2.1.6, никакой собственный суффикс $z'_{A,B}$ не может быть редуцирован к $\phi(A)/r$.

Случай префикса. Тип $\phi(B)/r$ встречается в числителе $z'_{A,B}$ дважды: в $o_2\backslash(\phi(B)/r)$ и в $\phi(A)\backslash(\phi(B)/r)$. Если некоторый собственный префикс редуцируется к $\phi(B)/r$, то он должен как минимум содержать $o_2\backslash(\phi(B)/r)$. Таким образом, имеются два варианта: выбранный префикс может заканчиваться перед r или на r.

Если префикс заканчивается на r, то интерпретация в свободной группе данной строки равна единице, а $[\![\phi(B)/r]\!] = r^{-1}$. Значит, по лемме 2.1.4, он не может быть редуцирован к $\phi(B)/r$.

Тогда единственная возможность для префикса — закончиться на типе $o_2\backslash(\phi(B)/r)$. Поскольку никакой собственный префикс x не редуцируется к $\phi(B)$, и никакой собственный суффикс x не редуцируется к $\phi(A)$ (согласно лемме 2.3.2), мы можем применить предложение 2.1.7, где $A/C = o_1/\phi(B)$, $\beta = z_{A,B}$, $D\backslash B = \phi(A)\backslash o_2$.

Тем самым для редукции выбранного префикса к $\phi(B)/r$ одна из следующих строк должна редуцироваться к $\phi(B)/r$:

$$(\phi(A)/r)/(\phi(B)/r); \phi(A)/o_1; o_1/\phi(B); \phi(A); \phi(A)\backslash o_2; o_2\backslash(\phi(B)/r)$$
$$(\phi(A)/r)/(\phi(B)/r); \phi(A)/o_1; o_1/\phi(B); \phi(B); \phi(A)\backslash o_2; o_2\backslash(\phi(B)/r)$$

Каждая из этих строк имеет только одно дерево редукции: первая не редуцируется ни к какому типу, а вторая редуцируется только к $\phi(A)/r$. □

Продолжая доказательство леммы 2.3.8, строка типов $u_{A,B}$ получается из промежуточной строки $z'_{A,B}$ похожим преобразованием, которое добавляет деление на ℓ слева.

$$u_{A,B} = (\ell\backslash(\phi(A)/r))/(\phi(B)/r); \ell; (\ell\backslash(\phi(A)/r))/o_3; o_3/(\phi(B)/r); z'_{A,B};$$
$$(\phi(A)/r)\backslash o_4; o_4\backslash(\phi(B)/r); (\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r))$$

Для доказательства части I леммы 2.3.8 обозначим через u и v следующие подстроки строки $u_{A,B}$.

$$u = (\ell\backslash(\phi(A)/r))/o_3; o_3/(\phi(B)/r); z'_{A,B}; (\phi(A)/r)\backslash o_4; o_4\backslash(\phi(B)/r)$$
$$v = \ell; (\ell\backslash(\phi(A)/r))/o_3; o_3/(\phi(B)/r); z'_{A,B}; (\phi(A)/r)\backslash o_4; o_4\backslash(\phi(B)/r)$$

Таким образом, вся строка $u_{A,B}$ может быть представлена как:

$$(\ell\backslash(\phi(A)/r))/(\phi(B)/r); \ell; u; (\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r)).$$

И в то же самое время как:

$$(\ell\backslash(\phi(A)/r))/(\phi(B)/r); v; (\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r)).$$

Следующие редукции показывают, что u редуцируется к $\psi(A)$, а v редуцируется к $\phi(B)/r$.

$$\dfrac{\dfrac{(\ell\backslash(\phi(A)/r))/o_3 \quad \dfrac{o_3/(\phi(B)/r) \quad \dfrac{\dfrac{z'_{A,B}}{\phi(A)/r} \quad \dfrac{(\phi(A)/r)\backslash o_4}{o_4} \quad \dfrac{o_4\backslash(\phi(B)/r)}{\phi(B)/r}}{o_3}}{\ell\backslash(\phi(A)/r)}}{\ell}$$

$$\dfrac{\dfrac{\dfrac{(\ell\backslash(\phi(A)/r))/o_3 \quad \dfrac{o_3/(\phi(B)/r) \quad \dfrac{z'_{A,B}}{\phi(B)/r}}{o_3}}{\ell\backslash(\phi(A)/r)}}{\phi(A)/r} \quad \dfrac{\dfrac{(\phi(A)/r)\backslash o_4 \quad o_4\backslash(\phi(B)/r)}{o_4}}{\phi(B)/r}}{\ell}$$

Следующие редукции показывают, что u редуцируется к $\psi(A)$ и $\psi(B)$.

$$\dfrac{\dfrac{(\ell\backslash(\phi(A)/r))/(\phi(B)/r) \quad \dfrac{\ell \quad \dfrac{\dfrac{v}{\ell\backslash(\phi(A)/r)} \quad \dfrac{(\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r))}{\ell\backslash(\phi(B)/r)}}{\phi(B)/r}}{\ell\backslash(\phi(A)/r)}}{\ell}$$

$$\dfrac{\dfrac{(\ell\backslash(\phi(A)/r))/(\phi(B)/r) \quad \dfrac{v}{\phi(B)/r}}{\ell\backslash(\phi(A)/r)} \quad \dfrac{(\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r))}{\ell\backslash(\phi(B)/r)}}{\ell}$$

Это завершает доказательство части I леммы 2.3.8,

Переходим к доказательству части II леммы 2.3.8 для случая **префикса**. Требуемый тип $\ell\backslash(\phi(B)/r)$ встречается в числителе только в последнем типе строки $u_{A,B}$. Следовательно, по предложению 2.1.6, никакой собственный префикс $u_{A,B}$ не может быть редуцирован к $\ell\backslash(\phi(B)/r)$.

В случае **суффикса** требуемый тип $\ell\backslash(\phi(A)/r)$ присутствует в числителе только у типов $(\ell\backslash(\phi(A)/r))/(\phi(B)/r)$ и $(\ell\backslash(\phi(A)/r))/o_3$, поэтому, по предложению 2.1.6, суффикс должен содержать хотя бы один из них.

Если суффикс начинается с r, то интерпретация в свободной группе данной строки равна r^{-1}, а $[\![\ell\backslash(\phi(A)/r)]\!] = \ell^{-1}r^{-1}$. Значит, по лемме 2.1.4, он не может быть редуцирован к $\ell\backslash(\phi(B)/r)$. Таким образом, единственная возможность для выбранного суффикса — начинаться с $(\ell\backslash(\phi(A)/r))/o_3$.

Чтобы показать, что такая строка не редуцируется к $\ell\backslash(\phi(A)/r)$, необходимо применить предложение 2.1.7, где $A/C = o_3/(\phi(B)/r)$, $\beta = z'_{A,B}$ и $C\backslash D = (\phi(A)/r)\backslash o_4$. Следовательно, одна из следующих строк должна редуцироваться к $\ell\backslash(\phi(A)/r)$:

$$(\ell\backslash(\phi(A)/r))/o_1; o_1/(\phi(B)/r); \phi(B)/r; \\ (\phi(A)/r)\backslash o_2; o_2\backslash(\phi(B)/r); (\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r))$$

$$(\ell\backslash(\phi(A)/r))/o_1; o_1/(\phi(B)/r); \phi(A)/r; \\ (\phi(A)/r)\backslash o_2; o_2\backslash(\phi(B)/r); (\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r))$$

Первая строка редуцируется только к:

$$(\ell\backslash(\phi(A)/r)); (\phi(A)/r)\backslash o_2; o_2\backslash(\phi(B)/r); (\ell\backslash(\phi(A)/r))\backslash(\ell\backslash(\phi(B)/r)).$$

Вторая строка редуцируется только к $\ell\backslash(\phi(B)/r)$. □

Следующий шаг — обобщить $u_{A,B}$ на случай трёх и более типов, а также добавить дополнительный слой делений, которые окончательно устранят перекрёстное редуцирование между соседними кодировками.

Для любых типов $A_1, \ldots, A_n$ построим строку $w_{A_1,\ldots,A_n}$, которая редуцируется к $\psi(A_1)$, …, $\psi(A_n)$. Пусть $e_1, \ldots, e_{2n-2}$ — новые примитивные типы, не

встречающиеся ни в одном типе $u_{A_i,A_{i+1}}$. Тогда строка $w_{A_1,\ldots,A_n}$ имеет вид:

$$x_1; u_{A_1,A_2}; x_2; u_{A_2,A_3}; x_3; \ldots; x_{n-1}; u_{A_{n-1},A_n}; x_n, \text{ где}$$
$$x_1 = \psi(A_1)/e_1; e_1/\psi(A_2),$$
$$x_i = \psi(A_{i-1})\backslash e_{2i-2}; (e_{2i-2}\backslash\psi(A_i))/e_{2i-1}; e_{2i-1}/\psi(A_{i+1}),$$
$$\text{для всех } i \in \{2,\ldots,n-1\},$$
$$x_n = \psi(A_{n-1})\backslash e_{2(n-1)}; e_{2(n-1)}\backslash\psi(A_n).$$

Если $n = 1$, т.е. имеется только один тип в списке, то w_{A_1} определим как строку из одного типа $\psi(A_1)$.

Лемма 2.3.11. *Пусть $A_1,\ldots,A_n$, где $n \geqslant 1$ — произвольные попарно различные типы форм p, p/q или $(p/q)/s$. Тогда строка типов $w_{A_1,\ldots,A_n}$ редуцируется к каждому из типов $\psi(A_1),\ldots,\psi(A_n)$.*

Доказательство. Случай, когда $n = 1$, ясен, для $n \geqslant 2$, доказательство индукцией по n.

База: $n = 2$. Тогда строка w_{A_1,A_2} редуцируется как к $\psi(A_1)$, так и к $\psi(A_2)$, исходя из следующего:

$$\dfrac{\dfrac{\psi(A_1)/e_1 \quad \dfrac{e_1/\psi(A_2) \quad \dfrac{\dfrac{u_{A_1,A_2}}{\psi(A_1)} \quad \psi(A_1)\backslash e_2}{e_2}}{e_1}}{\psi(A_1)} \quad e_2\backslash\psi(A_2)}{\psi(A_2)}$$

$$\dfrac{\dfrac{\psi(A_1)/e_1 \quad \dfrac{e_1/\psi(A_2) \quad \dfrac{u_{A_1,A_2}}{\psi(A_2)}}{e_1}}{\psi(A_1)} \quad \dfrac{\psi(A_1)\backslash e_2}{e_2}}{\psi(A_2)} \quad e_2\backslash\psi(A_2)$$

Шаг индукции: Следующие редукции строк типов показывают, что суффикс $u_{A_n,A_{n+1}}; x_{n+1}$ строки $w_{A_1,\ldots,A_{n+1}}$ редуцируется к $\psi(A_{n+1})$, а также, что

$x_n; \psi(A_{n+1})$ редуцируется к $\psi(A_{n-1})\backslash e_{2n-2}; e_{2n-2}\backslash\psi(A_n)$:

$$\dfrac{\dfrac{u_{A_n,A_{n+1}}}{\psi(A_n)} \quad \psi(A_n)\backslash e_{2n}}{e_{2n}} \quad \dfrac{e_{2n}\backslash\psi(A_{n+1})}{\psi(A_{n+1})}$$

$$\psi(A_{n-1})\backslash e_{2n-2} \quad \dfrac{(e_{2n-2}\backslash\psi(A_n))/e_{2n-1} \quad \dfrac{e_{2n-1}/\psi(A_{n+1}) \quad \psi(A_{n+1})}{e_{2n-1}}}{e_{2n-2}\backslash\psi(A_n)}$$

Таким образом, строка $w_{A_1,\ldots,A_{n+1}}$, имеющая вид $\ldots; x_n; u_{A_n,A_{n+1}}; x_{n+1}$, редуцируется к строке вида $\ldots; u_{A_{n-1},A_n}; x_n; \psi(A_{n+1})$, а после этого уже эта строка к $\ldots; u_{A_n,A_{n+1}}; \psi(A_{n-1})\backslash e_{2n-2}; e_{2n-2}\backslash\psi(A_{n-1})$. Последняя в точности совпадает с $w_{A_1,\ldots,A_n}$. Тогда, по индукционному предположению, $w_{A_1,\ldots,A_{n+1}}$ редуцируется к $\psi(A_i)$, где $i \in \{1,\ldots,n\}$.

В то же время строка $w_{A_1,\ldots,A_{n+1}} = \ldots; x_n; u_{A_n,A_{n+1}}; x_{n+1}$ редуцируется к строке $\ldots; x_n; \psi(A_{n+1}); x_{n+1}$ (путём редукции $u_{A_n,A_{n+1}}$ к $\psi(A_{n+1})$), а такая строка, в свою очередь, редуцируется к $w_{A_1,\ldots,A_n}; x_{n+1}$. Далее, по индукционному предположению, вся $w_{A_1,\ldots,A_n}$ редуцируется к $\psi(A_n); x_{n+1}$. Последняя в точности совпадает со строкой, которая в первой редукции выше была редуцирована к $\psi(A_{n+1})$. Следовательно, строка $w_{A_1,\ldots,A_{n+1}}$ редуцируется к $\psi(A_{n+1})$. □

Пусть $G = (\Sigma, N, S, R)$ – произвольная контекстно-свободная грамматика в нормальной форме Грейбах. Обозначим язык, определяемый каждым нетерминалом $X \in N$, через $L_G(X)$. Для каждого символа $a \in \Sigma$ рассмотрим все правила с a в правой части, имеющие вид: $X \to a$, $X \to aY$ и $X \to aZY$. Тогда гомоморфный образ a определяется как:

$$h(a) = \ell; w_{A_1,\ldots,A_m}; r,$$
$$\text{где } \{A_1,\ldots,A_m\} = \{\, X \mid X \to a \in R \,\} \cup \{\, X/Y \mid X \to aY \in R \,\} \cup$$
$$\cup \{\, (X/Y)/Z \mid X \to aZY \in R \,\}.$$

Согласно лемме 2.3.11, строка типов $h(a)$ может быть редуцируема к каждому из типов $\phi(A_1), \ldots, \phi(A_m)$, то есть к кодировкам упомянутых типов, каждый из которых соответствует одному из правил грамматики с a. Или, иначе говоря, одной из возможных синтаксических ролей a.

Тогда для любой строки $w = a_1 \ldots a_n$ её образ $h(a_1 \ldots a_n)$ редуцируется к строке типов $B_1 \ldots B_n$, где каждый B_i соответствует одной из синтаксических ролей символа a_i. Эти типы, условно говоря, формируют дерево разбора w, если такое дерево существует. Следующая лемма формализует это утверждение, а также устанавливает обратное: редукция к единственному типу влечёт существование разбора в грамматике G.

Лемма 2.3.12 (Основная лемма)**.** *Для произвольной строки $y \in \Sigma^+$ и произвольного нетерминала $X \in N$, строка y принадлежит $L_G(X)$ тогда и только тогда, когда её образ $h(y)$ редуцируется к типу $\phi(X)$.*

Доказательство проводится отдельно в две стороны, причём направление «слева направо» более просто.

Доказательство простого направления. Пусть строка $y \in \Sigma^+$ принадлежит $L_G(X)$ для некоторого $X \in N$. Требуется построить редукцию $h(y)$ к типу $\phi(X)$. Доказательство проводится индукцией по длине y.

База индукции: $y = a \in \Sigma$. Тогда в G существует правило $X \to a$, и, соответственно, тип X присутствует в образе a, то есть $h(a) = \ell; w_{\ldots,X,\ldots}; r$. По лемме 2.3.11, строка $w_{\ldots,X,\ldots}$ редуцируется к $\psi(X) = \ell \backslash (\phi(X)/r)$. Следовательно, $h(a)$ редуцируется к $\phi(X)$.

Шаг индукции: $|y| \geqslant 2$. Тогда y получена по правилу вида $X \to aY$ или $X \to aYZ$.

В первом случае пусть $y = ay'$, где y' определяется нетерминалом $Y \in N$. По предположению индукции $h(y')$ редуцируется к $\phi(Y)$. Кроме того, так как правило $X \to aY$ присутствует в G, тип X/Y включён в образ $h(a)$, то есть $h(a) = \ell; w_{\ldots,X/Y,\ldots}; r$. По лемме 2.3.11, $h(a)$ редуцируется к строке типов

$\ell; \ell\backslash(\phi(X/Y)/r); r = \ell; \ell\backslash((\phi(X)/\phi(Y))/r); r$, а следовательно и к $\phi(X)/\phi(Y)$. Таким образом, $h(y)$ редуцируется к $\phi(X)/\phi(Y); \phi(Y)$, которая в свою очередь редуцируется к $\phi(X)$.

Во втором случае $(X \to aYZ)$ строка y представляется как $ay'z'$, где $y' \in L_G(Y)$ и $z' \in L_G(Z)$. По предположению индукции, $h(y')$ редуцируется к $\phi(Y)$, а $h(z')$ — к $\phi(Z)$. Так как правило $X \to aYZ$ присутствует в $h(a)$, то данная строка редуцируется к $\phi((X/Z)/Y)$, по лемме 2.3.11. Последний тип равен $(\phi(X)/\phi(Z))/\phi(Y)$. Тогда вся строка $h(y)$ редуцируется к строке типов $(\phi(X)/\phi(Z))/\phi(Y); \phi(Y); \phi(Z)$, а последняя к $\phi(X)$. □

Теперь осталось доказать «сложное направление» основной леммы: если гомоморфный образ $h(y)$ строки y редуцируется к типу, соответствующему нетерминалу исходной грамматики, то y принадлежит языку, порожденному этим нетерминалом. Первая часть этого доказательства — следующая лемма, утверждающая, что если $h(y)$ редуцируется к X/X, то существует редукция, осуществляющая это упорядоченным образом, т.е. путём пошаговой редукции образа каждого символа к типу, соответствующему одной из его синтаксических ролей.

Лемма 2.3.13. *Если $h(y) = h(a_1 \dots a_n)$ редуцируется к типу $X/X = \phi(X)$, тогда существует строка формы $\phi(C_1); \dots; \phi(C_n)$, которая редуцируется к X/X, где каждый тип C_i выбран из множества:*

$$\{\, X \mid X \to a_i \in R \,\} \cup \{\, X/Y \mid X \to a_i Y \in R \,\} \cup \{\, (X/Y)/Z \mid X \to a_i ZY \in R \,\}.$$

Доказательство. Пусть $h(y) = h(a_1 \dots a_n)$ – строка типов, редуцируемая к X/X. Рассмотрим подстроку $h(a_1) = \ell; w_{A_1,\dots,A_m}; r = \ell; x_1; u_{A_1,A_2}; x_2; \dots; x_m; r$ в $h(a_1 \dots a_n)$, где $A_1, \dots, A_m$ – все типы, кодирующие правила грамматики с символом a_1, как описанно в построении гомоморфизма h. Таким образом,

$h(a_1 \dots a_n)$ имеет вид:

$$\ell; \underbrace{\psi(A_1)/e_1; e_1/\psi(A_2)}_{x_1}; u_{A_1,A_2}; \underbrace{\psi(A_1)\backslash e_2; (e_2\backslash\psi(A_2))/e_3; e_3/\psi(A_3)}_{x_2}; u_{A_2,A_3}; x_3; \dots; x_m; r; h(a_2 \dots a_n)$$

По лемме 2.3.8, никакой собственный префикс u_{A_1,A_2} не редуцируется к типу $\psi(A_2)$, и никакой собственный суффикс u_{A_1,A_2} не редуцируется к $\psi(A_1)$, а также $e_1/\psi(A_2)$ и $\psi(A_1)\backslash e_2$ не входят в множество знаменателей всей строки (по построению $w_{A_1 \dots A_m}$). Тогда к $h(a_1 \dots a_n)$ применимо предложение 2.1.7 со следующим разбиением на $\alpha; A/C; \beta; D\backslash B; \gamma$:

$$\underbrace{\ell; \psi(A_1)/e_1}_{\alpha}; \underbrace{e_1/\psi(A_2)}_{A/C}; \underbrace{u_{A_1,A_2}}_{\beta}; \underbrace{\psi(A_1)\backslash e_2}_{D\backslash B}; \underbrace{(e_2\backslash\psi(A_2))/e_3; e_3/\psi(A_3); u_{A_2,A_3}; x_3; \dots; x_m; r; h(a_2 \dots a_n)}_{\gamma}.$$

Согласно предложению 2.1.7, для случая $A/C = e_1/\psi(A_2)$, $\beta = u_{A_1,A_2}$ и $D\backslash B = \psi(A_1)\backslash e_2$, одна из строк, полученных заменой u_{A_1,A_2} на $\psi(A_1)$ или $\psi(A_2)$, также редуцируется к X/X:

$$\psi(A_1)/e_1; e_1/\psi(A_2); \underline{\psi(A_1)}; \psi(A_1)\backslash e_2; (e_2\backslash\psi(A_2))/e_3; e_3/\psi(A_3); \dots; x_m; r; \dots$$
$$\psi(A_1)/e_1; e_1/\psi(A_2); \underline{\psi(A_2)}; \psi(A_1)\backslash e_2; (e_2\backslash\psi(A_2))/e_3; e_3/\psi(A_3); \dots; x_m; r; \dots$$

Пусть $j_1 \in \{1, 2\}$ – индекс, при котором строка с заменой u_{A_1,A_2} на $\psi(A_{j_1})$ редуцируется к X/X. Рассмотрим подстроку u_{A_2,A_3} в этой строке. Применим предложение 2.1.7 к следующему разбиению:

$$\underbrace{\ell; \psi(A_1)/e_1; e_1/\psi(A_2); \psi(A_{j_1}); \psi(A_i)\backslash e_2; (e_2\backslash\psi(A_2))/e_3}_{\alpha}; \underbrace{e_3/\psi(A_3)}_{A/C}; \underbrace{u_{A_2,A_3}}_{\beta}; \underbrace{\psi(A_2)\backslash e_4}_{D\backslash B}; \underbrace{\dots; x_m; r; h(a_2 \dots a_n)}_{\gamma}$$

Согласно предложению 2.1.7, одна из двух новых строк (с заменой u_{A_2,A_3} на $\psi(A_2)$ или $\psi(A_3)$) редуцируется к X/X. Пусть $j_2 \in \{2, 3\}$ – индекс, при котором строка с заменой на $\psi(A_{j_2})$ редуцируется к X/X.

Аналогичные рассуждения повторяются для каждой подстроки $u_{A_k,A_{k+1}}$ в $h(a_1)$ (всего $m-1$ раз), пока $h(a_1 \dots a_n)$ не будет заменена на строку вида $\ell; \alpha_1; r; h(a_2); \dots; h(a_n)$, где α_1 получена из $w_{A_1,\dots A_n}$ заменой каждого $u_{A_i,A_{i+1}}$ на $\psi(A_i)$ или $\psi(A_{i+1})$.

Теперь рассмотрим подстроку $h(a_2)$ в строке $\ell; \alpha_1; r; h(a_2); \dots; h(a_n)$ и повторим для неё те же шаги. Проделаем аналогичные замены для каждого $h(a_i)$. В итоге получим строку вида $\ell; \alpha_1; r; \dots; \ell; \alpha_n; r$, редуцируемую к X/X, где каждая подстрока α_i получена заменой всех строк вида $u_{C,D}$ в $h(a_i)$ на один из типов $\psi(C)$ или $\psi(D)$.

Рассмотрим произвольную α_i в полученной строке. Эта подстрока состоит из типов двух видов. Первый вид – типы, полученные редукцией $u_{D,E}$ к $\psi(D)$ или $\psi(E)$. Поскольку каждый $u_{D,E}$ является подстрокой $h(a_i)$, в силу определения $h(a_i)$, эти типы имеют вид $\psi(C)$, где C принадлежит следующему множеству:

$$\{\, X \mid X \to a_i \in R \,\} \cup \{\, X/Y \mid X \to a_i Y \in R \,\} \cup \{\, (X/Y)/Z \mid X \to a_i Z Y \in R \,\}.$$

Второй вид типов в α_i – оставшиеся типы из исходной строки $h(a_i) = \ell; w_{A_1,\dots,A_n}; r$, сохранившиеся в $\ell; \alpha_i; r$. Каждый из них имеет одну из форм:

$$\psi(C)/e_k, e_k/\psi(C), \psi(C)\backslash e_k, e_k\backslash\psi(C), (e_{k-1}\backslash\psi(C))/e_k,$$

где C принадлежит указанному выше множеству (кодирующему правила с a_i), а e_{k-1} и e_k — примитивные типы, введённые в определении $w_{A_1,\dots,A_n}$. Все типы в α_i обладают следующими общими свойствами.

Утверждение 2.3.14. *Интерпретация в свободной группе произвольного типа из α_i содержит равные степени ℓ и r. Единственные типы в множестве числителей α_i, редуцируемые с ℓ — типы $\psi(C)$, где C принадлежит указанному выше множеству. Аналогично, единственные типы, редуцируемые с типом r, — $\phi(C)/r$ для C из того же множества.*

Доказательство проводится проверкой всех перечисленных форм типов. Следовательно, интерпретация в свободной группе любого типа, полученного редукцией в некоторой подстроке α_i, общая степень (сумма всех степеней вхождения данной образующей в элемент) образующей r должна совпадать с общей степенью образующей ℓ.

Утверждение 2.3.15. *В редукции строки $\ell;\alpha_1;r;\ldots;\ell;\alpha_n;r$ до X/X не могли встречаться типы ни одного из следующих видов:*

$$D/\ell, r\backslash D, D/(\phi(C)/r), (\phi(C)/r)\backslash D,$$

где C и D – произвольные типы.

Это следует из того, что такие типы не входят в множество числителей всей строки $\ell;\alpha_1;r;\ldots;\ell;\alpha_n;r$, что видно при анализе всех форм типов в числителях α_i.

Теперь утверждается, что при редукции $\ell;\alpha_1;r;\ldots;\ell;\alpha_n;r$ к X/X каждая подстрока $\ell;\alpha_i;r$ должна быть сначала редуцирована к строке вида $\phi(C)/r;\widetilde{x};r$.

Рассмотрим последний шаг редукции, когда типы ℓ и r из подстроки $\ell;\alpha_i;r$ ещё не затронуты. Пусть $\ell;x;r$ — текущая подстрока, полученная из $\ell;\alpha_i;r$. На следующем шаге один из типов ℓ или r редуцируется с некоторым типом B. Во-первых, B должен находиться внутри подстроки x, так как в противном случае B должен быть вида D/ℓ или $r\backslash D$, что невозможно по утверждению 2.3.15. Следовательно, B — результат редукции типов из α_i. По утверждению 2.3.14, общая степень образующей ℓ интерпретации B в свободной группе совпадает с общей степенью образующей r. Кроме того, B должен быть редуцируем с ℓ или r, откуда $B = \ell\backslash(\phi(C)/r)$ или $B = \phi(C)/r$ для некоторого C. Второй случай невозможен, так как $[\![\phi(C)/r]\!] = r^{-1}$. Таким образом, подстрока $\ell;\alpha_i;r$ редуцируется к $\ell;\ell\backslash(\phi(C)/r);\widetilde{x};r$, где $x = \ell\backslash(\phi(C)/r);\widetilde{x}$, а затем — к $\phi(C)/r;\widetilde{x};r$.

Утверждение 2.3.16. *Строка $\widetilde{x}$ обязана быть пустой, следовательно, $\ell;\alpha_i;r$*

редуцируется к $\phi(C)/r; r$.

Предположим противное: $\widetilde{x}$ непуста. Рассмотрим первую редукцию с участием r или $\phi(C)/r$ из подстроки $\phi(C)/r; \widetilde{x}; r$. Эти типы не могут быть редуцированы с типами вне $\widetilde{x}$, так как для этого потребовались бы типы вида $r \backslash D$ или $D/(\phi(C)/r)$, что невозможно по утверждению 2.3.15.

Следовательно, r или $\phi(C)/r$ были редуцированы с типом E, полученным из $\widetilde{x}$. По утверждению 2.3.14, общие степени образующих r и ℓ в типе E совпадают. Кроме того, E может быть редуцирован с r только если $E = \phi(C)/r$, но это невозможно, так как $[\![\phi(C)/r]\!] = r^{-1}$. Аналогично, для редукции с $\phi(C)/r$ тип E должен быть вида $(\phi(C)/r) \backslash D$ или r, что также невозможно по утверждению 2.3.15.

Таким образом, при редукции $\ell; \alpha_1; r; \ldots; \ell; \alpha_n; r$ к X/X каждый блок $\ell; \alpha_i; r$ редуцируется к строке вида $\phi(C_i)/r; r$, где каждый тип C_i принадлежит множеству:

$$\{ X \mid X \to a_i \in R \} \cup \{ X/Y \mid X \to a_i Y \in R \} \cup \{ (X/Y)/Z \mid X \to a_i Z Y \in R \}.$$

Аналогично построению из первой части доказательства, отсюда следует, что строка $\phi(C_1)/r; r; \ldots; \phi(C_n)/r; r$ редуцируется к X/X.

Наконец, в произвольной редукции этой строки к X/X необходимо, чтобы каждый блок $\phi(C_i)/r; r$ редуцировался к $\phi(C_i)$, так как типы r не могут быть редуцированны иначе. Следовательно, $\phi(C_1); \ldots; \phi(C_n)$ редуцируется к X/X, что и утверждалось. □

Таким образом, доказано, что редукция $h(y)$ к X/X должна быть определенным образом структурирована, что будет использовано для доказательства леммы 2.3.12.

Доказательство леммы 2.3.12 в сложную сторону. Пусть $y = a_1 \ldots a_n \in \Sigma^+$, и предположим, что $h(y)$ редуцируется к типу $\phi(X) = X/X$ для некоторого

нетерминала X. Утверждается, что строка y обязана принадлежать $L_G(X)$. Доказательство проводится индукцией по длине y.

База индукции: $y = a \in \Sigma$. По условию, $h(a)$ редуцируется к типу X/X. Согласно лемме 2.3.13, существует строка типов $\phi(C)$ длины 1, где $C \in \{\, X \mid X \to a \in R \,\} \cup \{\, X/Y \mid X \to aY \in R \,\} \cup \{\, (X/Y)/Z \mid X \to aZY \in R \,\}$, которая редуцируется к X/X. Но так как X/X – уже один тип, эта редукция возможна только если $\phi(C) = X/X$, откуда $X = C$. Следовательно, существует правило $X \to a \in R$, и потому $a \in L_G(X)$, что и требовалось.

Шаг индукции: $y = a_1 \dots a_m$, где $m \geqslant 2$. Как и в базе индукции, так как $h(y)$ редуцируется к X/X, по лемме 2.3.13, существует строка типов $\phi(C_1) \dots \phi(C_n)$, редуцируемая к X/X, причём каждый C_i принадлежит соответствующему множеству:

$$\{\, X \mid X \to a_i \in R \,\} \cup \{\, X/Y \mid X \to a_iY \in R \,\} \cup \{\, (X/Y)/Z \mid X \to a_iZY \in R \,\}.$$

Из-за формы каждого типа C_i (в нём только правые деления), все редукции строки $\phi(C_1) \dots \phi(C_n)$ имеют вид $E/F; F \Rightarrow E$. В частности, итоговый тип $X/X = \phi(X)$ должен находиться в числителе $\phi(C_1)$, что даёт два случая: либо $\phi(C_1) = \phi(X)/\phi(Y)$, либо $\phi(C_1) = (\phi(X)/\phi(Y))/\phi(Z)$.

В первом случае вся строка $\phi(C_2) \dots \phi(C_n)$ редуцируется к $\phi(Y)$, и по предположению индукции $a_2 \dots a_m \in L_G(Y)$. Так как $\phi(C_1) = \phi(X)/\phi(Y) = \phi(X/Y)$, то $C_1 = X/Y$. Следовательно, исходная грамматика содержит правило $X \to a_1Y$, откуда $a_1 \dots a_m \in L_G(X)$.

Во втором случае если $\phi(C_1) = (\phi(X)/\phi(Y))/\phi(Z)$, редукция всей строки $\phi(C_1); \dots; \phi(C_m)$ к $\phi(X)$ завершается следующими двумя шагами:

$$\dfrac{\dfrac{(\phi(X)/\phi(Y))/\phi(Z) \quad \phi(Z)}{\phi(X)/\phi(Y)} \qquad \phi(Y)}{\phi(X)}$$

Соответственно, строка $\phi(C_2) \dots \phi(C_n)$ разбивается на две подстроки: где первая – $\phi(C_2) \dots \phi(C_i)$ редуцируется к $\phi(Z)$, а вторая – $\phi(C_{i+1}) \dots \phi(C_n)$ к $\phi(Y)$.

По предположению индукции, $a_2 \dots a_i \in L_G(Z)$ и $a_{i+1} \dots a_m \in L_G(Y)$. Так как $\phi(C_1) = (\phi(X)/\phi(Y))/\phi(Z) = \phi((X/Y)/Z)$, то $C_1 = (X/Y)/Z$. Значит, исходная грамматика содержит правило $X \to a_1 ZY$, откуда $a_1 \dots a_m \in L_G(X)$. □

Следующую теорему удобнее всего сформулировать используя понятие *сильной сводимости*, с которого мы и начнём.

Определение 2.3.17. Язык $L_1 \in \Sigma^*$ *сводится посредством гомоморфизма* к языку $L_2 \in \Omega^*$ ($L_1 \leqslant L_2$), если существует гомоморфизм $h\colon \Sigma \to \Omega^*$ такой, что слово w принадлежит L_1 тогда и только тогда, когда $g(w)$ принадлежит L_2.

В итоге, всё готово для доказательства следующей теоремы.

Теорема 2.3.18. *Для произвольного контекстно-свободного языка $L \subseteq \Sigma^+$ существует AB-грамматика с однозначным присвоением типов G, такая, что $L \leqslant L(G)$.*

Доказательство. Для контекстно-свободного языка L рассмотрим контекстно-свободную грамматику в форме Грейбах (Σ, N, S, R), которая задаёт данный язык. Язык L не содержит пустого слова, следовательно, будем считать, что в данной грамматике нет ε-правил.

Для каждого терминального символа $a \in \Sigma$, рассмотрим строку типов $h(a) = B_1; \dots; B_n$, и для каждого типа B_i в данной строке в грамматику G добавим символ a_i, а в грамматику G добавим правило $a_i \triangleright B_i$. В грамматике G выберем тип S/S в качестве результирующего. Определим значение гомоморфизма g на символе a как строку $a_1 \dots a_n$. Таким образом, для произвольных двух различных символов a и b из Σ, строки $g(a)$ и $g(b)$ состоят из разных букв. Полученная грамматика является AB-грамматикой с однозначным присвоением типов. По лемме 2.3.12, строка y принадлежит языку L тогда и только тогда, когда строка типов $h(y)$ редуцируется к типу $\phi(S) = S/S$, а это и значит, что грамматика G порождает слово $g(y)$. Это, в свою очередь, значит, что $L \leqslant L(G)$. □

2.4. Сведение к грамматикам из класса $\mathcal{G}^2$ посредством гомоморфизмов

Перейдём к грамматикам из класса $\mathcal{G}^2$, то есть базовым категориальным грамматикам, где допускается присвоение символу не более двух типов. Оказывается, в данном случае можно получить усиленный аналог теоремы 2.3.18.

Для начала разберём ранее встречавшийся пример языка $\{a, aa\}$. Грамматика с присвоениями $a \triangleright r \mid p \mid r/p$ задаёт данный язык.

Пример 9. Базовая категориальная грамматика с правилами:

$$a_1 \triangleright r/t_3/t_2 \mid t_1$$
$$a_2 \triangleright t_1 \backslash p/t_3 \mid t_2$$
$$a_3 \triangleright t_2 \backslash t_1 \backslash r/p \mid t_3$$

задаёт язык $\{a_1a_2a_3, a_1a_2a_3a_1a_2a_3\}$.

Каждая строка $a_1a_2a_3$ «кодирует» в себе все три типа, присвоенные букве a в исходной грамматике. Иначе говоря, строка $a_1a_2a_3$ редуцируется ко всем типам буквы a в исходной грамматике. Аналогично тому, как данное сделано в конструкции доказательства теоремы 2.3.18. Более того, никаких слов, кроме полученных заменой a на $a_1a_2a_3$ в исходном языке, данная грамматика не порождает. Отметим, что в теореме 2.3.18 подобное не утверждалось, и в общем случае построения неверно.

Зафиксируем произвольную базовую категориальную грамматику G и обобщим кодировку грамматики из примера.

Определение 2.4.1. Построим грамматику G' по грамматике G, заменив все

присвоения одному символу $a \triangleright A_1 \mid \ldots \mid A_n$ на следующий набор правил.

$$
\begin{aligned}
&a_1 \triangleright A_1/t_n^a/t_{n-1}^a/\ldots/t_2^a \mid t_1^a \\
&a_2 \triangleright t_1^a\backslash A_2/t_n^a/t_{n-1}^a/\ldots/t_3^a \mid t_2^a \\
&\ldots \\
&a_{n-1} \triangleright t_{n-2}^a\backslash\ldots\backslash t_1^a\backslash A_{n-1}/t_n^a \mid t_{n-1}^a \\
&a_n \triangleright t_{n-1}^a\backslash\ldots\backslash t_1^a\backslash A_n \mid t_n^a
\end{aligned}
$$

Здесь, t_i^a — новый примитивный тип, не встречающийся в типах изначальной грамматики. Обозначим A_i' и A_i'' соответственно типы, присвоенные символу a_i. Для символа a определим образ гомоморфизма $g_1(a)$ как строку $a_1 a_2 \ldots a_n$.

Обратно, для символа a_1 определим образ гомоморфизма $g_2(a_1) = a$ и для всех остальных символов a_i (где $i \neq 1$) положим $g_2(a_i) = \varepsilon$.

По теореме 1.3.3 перейдём к эквивалентным контекстно-свободным грамматикам для G и G'. Таким образом, грамматика $CF(G)$ определяется следующим набором правил (выпишем правила с терминалом a в правой части):

$$
\begin{aligned}
&nu(A_1) \;\to\; ld(A_1)\, a\, rd(A_1) \\
&nu(A_2) \;\to\; ld(A_2)\, a\, rd(A_2) \\
&\ldots \\
&nu(A_n) \;\to\; ld(A_n)\, a\, rd(A_n)
\end{aligned}
$$

По определению $nu(A_i) = nu(A_i')$, а строки левых и правых знаменателей следующие:

$$
\begin{aligned}
&ld(A_1') = ld(A_1) && rd(A_1') = t_2^a\, t_3^a \ldots t_n^a\, rd(A_1) \\
&ld(A_2') = ld(A_2)\, t_1^a && rd(A_1') = t_3^a \ldots t_n^a\, rd(A_2) \\
&\ldots && \ldots \\
&ld(A_n') = ld(A_n)\, t_1^a\, t_2^a\, t_3^a \ldots t_{n-1}^a && rd(A_n') = rd(A_n)
\end{aligned}
$$

Для A_i'' положим $nu(A_i'') = t_i^a$, а строки левых и правых знаменателей, соответственно, пустыми.

Итого, грамматика G' эквивалентна контекстно-свободной грамматике с правилами:

$$\begin{array}{ll}
nu(A_1) \to ld(A_1)\, a_1\, t_2^a\, t_3^a \ldots t_{n-1}^a\, t_n^a\, rd(A_1) & t_1^a \to a_1 \\
nu(A_2) \to ld(A_2)\, t_1^a\, a_2\, t_3^a \ldots t_{n-1}^a\, t_n^a\, rd(A_2) & t_2^a \to a_2 \\
\ldots & \ldots \\
nu(A_n) \to ld(A_n)\, t_1^a\, t_2^a\, t_3^a \ldots t_{n-1}^a\, a_n\, rd(A_n) & t_n^a \to a_n
\end{array}$$

Теперь заметим, что каждый нетерминал t_i^a встречается в левой части грамматики ровно в одном правиле, а именно в i-м из правых n. Тем самым, можно везде в грамматике заменить t_i^a на a_i и получить эквивалентную грамматику.

Таким образом, получили грамматику (выписываем правила только полученные из присвоений для символа a):

$$\begin{array}{l}
nu(A_1) \to ld(A_1)\, a_1\, a_2\, a_3 \ldots a_{n-1}\, a_n\, rd(A_1) \\
nu(A_2) \to ld(A_2)\, a_1\, a_2\, a_3 \ldots a_{n-1}\, a_n\, rd(A_2) \\
\ldots \\
nu(A_n) \to ld(A_n)\, a_1\, a_2\, a_3 \ldots a_{n-1}\, a_n\, rd(A_n)
\end{array}$$

которая отличается от правил грамматики G заменой терминала a на строку терминалов $a_1 \ldots a_n$.

Тем самым доказана следующая теорема.

Теорема 2.4.2. *Для произвольного контекстно-свободного языка $L \subseteq \Sigma^+$ существует G — грамматика из класса $\mathcal{G}^2$ такая, что $L \leqslant L(G)$ и $L(G) \leqslant L$.*

Глава третья. Алгоритмические свойства

3.1. Грамматики класса $\mathcal{G}^2$

В конце предыдущей главы в параграфе 2.4 мы доказали теорему 2.4.2. Эта теорема говорит, что произвольный контекстно-свободный язык можно закодировать при некотором гомоморфизме языком, заданным грамматикой из класса $\mathcal{G}^2$. При этом имеет место сведение в обе стороны. Следовательно, большинство алгоритмических свойств класса языков, заданных грамматиками из $\mathcal{G}^2$, совпадают с алгоритмическими свойствами контекстно-свободных языков. Следующая теорема является одним из основных результатов диссертации.

Теорема 3.1.1. *Следующие задачи для грамматик из класса $\mathcal{G}^2$ алгоритмически неразрешимы:*

- *проверить по двум грамматикам из класса $\mathcal{G}^2$, равны ли задаваемые ими языки;*
- *по произвольной грамматике G из класса $\mathcal{G}^2$ проверить, является ли язык, задаваемый грамматикой G, регулярным.*

Доказательство. Во втором случае свойства языков сохраняются под действием гомоморфизмов, таким образом, утверждаемое является следствием классической теоремы для контекстно-свободных грамматик.

Для доказательства первого утверждения заметим, что в произвольных двух грамматиках G_1 и G_2 можно считать, что выбранному символу a в первой и во второй грамматике присвоено одинаковое количество типов. Иначе некоторые типы можно продублировать. Построим по G_1 и G_2 грамматики G_1' и G_2' соответственно, как это делалось в доказательстве теоремы 2.4.2. Тогда для произвольного символа a верно, что $h_1(a) = g_1(a)$, где h_1 и g_1, соответ-

ствующие гомоморфизмы. Значит, $L(G'_1) = L(G'_2)$ тогда и только тогда, когда $L(G_1) = L(G_2)$. □

Данное следствие не исчерпывает весь список утверждений, которые могут быть доказаны при помощи теоремы 2.4.2. Общая структура доказательств остаётся аналогичной. Например, доказательство неразрешимости задачи включения языков порожденных грамматиками из класса $\mathcal{G}^2$ полностью повторяет доказательство первого утверждения из теоремы 3.1.1.

Завершая данный параграф, отметим, что приведённые выше теоремы могут быть обобщены на произвольный класс грамматик $\mathcal{G}^k$, где $k > 1$. Более того, предложенная конструкция демонстрирует, что даже грамматики из класса $\mathcal{G}^2$ способны закодировать произвольный контекстно-свободный язык «в обе стороны». Следовательно, неформально можно сделать вывод, что этот класс по своим выразительным возможностям находится «близко» к классу контекстно-свободных грамматик.

3.2. Грамматики класса $\mathcal{G}^1$

Перейдём к рассмотрению грамматик из класса $\mathcal{G}^1$. Сейчас мы докажем, что для грамматик из $\mathcal{G}^1$ некоторые задачи алгоритмически неразрешимы. Как уже было доказано ранее, класс языков, задаваемых данными грамматиками, не замкнут относительно большинства естественных операций. Это делает доказательства в данном случае трудными и требует особенных способов кодирования.

Стандартным образом доказательство неразрешимости основано на сведении известной неразрешимой задачи к выбранной. В качестве известной неразрешимой задачи в данном случае будет выступать проблема проверки существования решений для систем соответствий Поста.

Напомним, что *системой соответствий Поста* называется пара набо-

ров слов $\vec{x} = (x_1, \ldots, x_n)$ и $\vec{y} = (y_1, \ldots, y_n)$ над алфавитом из двух букв $\{a, b\}$. Число n будем называть *количеством компонент* системы соответствий Поста $(\vec{x}, \vec{y})$. *Решением* данной системы Поста называется непустая последовательность чисел $(i_1, \ldots, i_k)$, для которой выполняется условие $x_{i_1} \ldots x_{i_k} = y_{i_1} \ldots y_{i_k}$. Известно, что задача выяснения существования решения для произвольной системы Поста алгоритмически неразрешима. Иначе говоря, верна следующая теорема.

Теорема 3.2.1 (Пост [41]). *Задача определения для произвольной системы соответствий Поста $(\vec{x}, \vec{y})$, существует ли решение данной системы, алгоритмически неразрешима.*

Более того, известно что при фиксированном количестве компонент — $n = 5$, данная задача также неразрешима.

Теорема 3.2.2 (Нири [42]). *Задача определения для произвольной системы соответствий Поста $(\vec{x}, \vec{y})$, где количество компонент равно 5, существует ли решение данной системы, алгоритмически неразрешима.*

Нам потребуется следующее простое утверждение.

Предложение 3.2.3. *Если d — символ, не принадлежащий алфавиту Σ, то язык $\{d\} \cdot \Sigma^*$ принадлежит классу $\mathcal{L}^1$.*

Доказательство. Новой букве d присвоим результирующий тип r. Всем символам из Σ присвоим одинаковые типы $r \backslash r$. Любое слово из языка $\{d\} \cdot \Sigma^*$ порождается данной грамматикой, поскольку строка $r; r \backslash r; r \backslash r; \ldots; r \backslash r$ редуцируется к типу r. Осталось отметить, что в любой строке, составленной из типов $r \backslash r$ и r, которая редуцируется к r, тип r может встречаться лишь один раз и стоять в начале. □

Определение 3.2.4. Пусть $\vec{x} = (x_1, \ldots, x_n)$, где $x_i \in \{a, b\}^*$ при произвольном i. Обозначим через $G_{\vec{x}}$ контекстно-свободную грамматику над алфавитом

$\{a, b, c\}$ со следующим множеством правил:

$$\{S \to b\, a^i\, S\, x_i \mid i \in \{1, \ldots, n\}\} \cup \{S \to b\, a^i\, c\, x_i \mid i \in \{1, \ldots, n\}\}.$$

Обозначим за $L_{\vec{x}}$ язык $L(G_{\vec{x}})$, за $M_{\vec{x}\vec{y}}$ язык $\{a, b, c\}^* - (L_{\vec{x}} \cap L_{\vec{y}})$, и за $M'_{\vec{x}\vec{y}}$ язык $\{d\} \cdot M_{\vec{x}\vec{y}}$.

Нетрудно доказать, что язык $M_{\vec{x}\vec{y}}$ является контекстно-свободным (см., например, [43, лемма 15.12]). Следовательно, язык $M'_{\vec{x}\vec{y}}$ также является контекстно-свободным.

Предложение 3.2.5. *Пусть $(\vec{x}, \vec{y})$ — произвольная система соответствий Поста, тогда следующие утверждения равносильны:*

1. *$(\vec{x}, \vec{y})$ не имеет решений;*
2. *язык $M'_{\vec{x}\vec{y}}$ задаётся некоторой грамматикой из класса $\mathcal{G}^1$;*
3. *язык $L_{\vec{x}} \cap L_{\vec{y}}$ пуст.*

Доказательство. Для доказательства равносильности первого и второго утверждения см. [43, лемма 15.13].

Докажем равносильность второго и третьего утверждений. Доказательство проводится отдельно в каждую из сторон.

Справа налево. Если пересечение языков $L_{\vec{x}}$ и $L_{\vec{y}}$ пусто, то $M'_{\vec{x}\vec{y}} = \{d\} \cdot \{a, b, c\}^*$, а следовательно, по предложению 3.2.3, данный язык задаётся грамматикой из класса $\mathcal{G}^1$.

Слева направо. Пусть язык $M'_{\vec{x}\vec{y}}$ задаётся некоторой грамматикой G из класса $\mathcal{G}^1$. В $M'_{\vec{x}\vec{y}}$ есть слово, состоящее из одного символа d. Значит, d в грамматике G должен быть присвоен результирующий примитивный тип r. Также в языке $M'_{\vec{x}\vec{y}}$ есть слова da, db и dc. Значит, каждой из букв a, b и c должны быть присвоены одинаковые типы $r \backslash r$. Следовательно, любая строка du, где $u \in \{a, b, c\}^*$, должна принадлежать языку $L(G)$. Получаем, что $L(G) = \{d\} \cdot \{a, b, c\}^*$, значит, пересечение языков $L_{\vec{x}}$ и $L_{\vec{y}}$ должно быть пусто. □

Таким образом, доказана следующая теорема.

Теорема 3.2.6. *Задача определения для произвольной контекстно-свободной грамматики G того, задаётся ли язык $L(G)$ некоторой грамматикой из класса $\mathcal{G}^1$, алгоритмически неразрешима.*

Перейдём к случаю, когда имеется две грамматики из $\mathcal{G}^1$.

Теорема 3.2.7. *Задача определения для произвольных грамматик G_1 и G_2 из класса $\mathcal{G}^1$ того, является ли пересечение языков $L(G_1) \cap L(G_2)$ бесконечным языком, алгоритмически неразрешима.*

Доказательство. Зафиксируем некоторую систему соответствий Поста $(\vec{x}, \vec{y})$, где количество компонент равно n. Переименуем буквы в компонентах $\vec{x}$, дописав к каждой букве индекс 1. В компонентах $\vec{y}$ поставим везде индекс 2. Получается, что x_i — строка в алфавите $\{a_1, b_1\}$, а y_i — строка в алфавите $\{a_2, b_2\}$.

Теперь построим грамматики G_1 и G_2, пересечение языков которых будет кодировать решение фиксированной системы соответствий Поста. Алфавиты обеих грамматик совпадают и равны $\Sigma = \{c_1, \ldots, c_n, a_1, b_1, a_2, b_2, d\}$. Пусть p_1, q_1, p_2, q_2, r — примитивные типы, положим результирующий тип в обеих грамматиках равным r.

Нам потребуются гомоморфизмы $\alpha_1 \colon \Sigma^* \to \{p_1, q_1\}^*$, $\alpha_2 \colon \Sigma^* \to \{p_2, q_2\}^*$ и $\gamma \colon \Sigma^* \to \{a_1, b_1\}^*$ определенные на буквах Σ следующим образом.

$$\begin{array}{lll} \alpha_1(a_1) = p_1 & \alpha_2(a_2) = p_2 & \gamma(a_2) = a_1 \\ \alpha_1(b_1) = q_1 & \alpha_2(b_2) = q_2 & \gamma(b_2) = b_1 \\ \alpha_1(z) = \varepsilon, z \notin \{a_1, b_1\} & \alpha_2(z) = \varepsilon, z \notin \{a_2, b_2\} & \gamma(z) = \varepsilon, z \notin \{a_2, b_2\} \end{array}$$

Пусть строка x_i имеет вид $t_1 \ldots t_k$, а строка y_i — $\widetilde{t_1} \ldots \widetilde{t_m}$, где t_i и $\widetilde{t_j}$ — соответствующие буквы. Обозначим через A_{x_i,y_i} тип:

$$\alpha_2(\widetilde{t_m}) \backslash \ldots \backslash \alpha_2(\widetilde{t_1}) \backslash r \backslash \alpha_1(t_1) \backslash \ldots \backslash \alpha_1(t_k) \backslash r.$$

Грамматику G_1 зададим следующим множеством присвоений:

$$\{c_i \triangleright A_{x_i,y_i} \mid i \in \{1,\dots,n\}\} \cup \{a_1 \triangleright p_1, b_1 \triangleright q_1\} \cup \{a_2 \triangleright p_2, b_2 \triangleright q_2\} \cup \{d \triangleright r\}.$$

По теореме 1.3.3 грамматика G_1 эквивалентна контекстно-свободной грамматике (под $(\alpha_1(x_i))^R$ понимается перевёрнутая строка $\alpha_1(x_i)$):

$$\{r \to (\alpha_1(x_i))^R\, r\, \alpha_2(y_i)c_i \mid i \in \{1,\dots,n\}\} \cup \{p_1 \to a_1, q_1 \to b_1\} \cup \{p_2 \to a_2, q_2 \to b_2\} \cup \{r \to d\}.$$

Эквивалентная грамматика будет задаваться следующим множеством правил:

$$\{r \to x_i^R\, r\, y_i c_i \mid i \in \{1,\dots,n\}\} \cup \{r \to d\}.$$

Соответственно, $L(G_1) = \{x_{i_k}^R \dots x_{i_1}^R d y_{i_1} c_{i_1} \dots y_{i_k} c_{i_k} \mid k \in \mathbb{N},\ i_j \in \{1,\dots,n\}\}$.

Зададим грамматику G_2 следующими присвоениями:

$$\{c_i \triangleright r \backslash r \mid i \in \{1,\dots,n\}\} \cup \{d \triangleright r\} \cup \{a_1 \triangleright r/p_1,\ b_1 \triangleright r/p_2,\ a_2 \triangleright r \backslash p_1\, b_2 \triangleright r \backslash p_2\}.$$

По теореме 1.3.3 эквивалентная контекстно-свободная грамматика имеет следующий набор правил:

$$\{r \to r c_i \mid i \in \{1,\dots,n\}\} \cup \{r \to d\} \cup \{r \to a_1 r a_2,\ r \to b_1 r b_2\}.$$

Данная грамматика задаёт язык:

$$L(G_2) = \{udv \mid u \in \{a_1, b_1\}^*,\ v \in (\{a_2, b_2\} \cup \{c_i \mid i \in \{1,\dots,n\}\})^*,\ \gamma(v) = u^R\}.$$

Вектор $(i_1, \dots, i_k)$ является решением изначальной системы соответствий Поста $(\vec{x}, \vec{y})$ тогда и только тогда, когда выполняется следующая серия равенств: $\gamma(y_{i_1} c_{i_1} \dots y_{i_k} c_{i_k}) = (x_{i_k}^R \dots x_{i_1}^R)^R = x_{i_1} \dots x_{i_k}$. Таким образом, если решение существует, то пересечение языков $L(G_1)$ и $L(G_2)$ бесконечно, а если решений нет, то $L(G_1) \cap L(G_2) = \{d\}$.

Следовательно, задача определения для произвольной системы соответствий Поста $(\vec{x}, \vec{y})$ существования решения, сводится к задаче проверки бесконечности пересечения языков, заданных грамматиками из класса $\mathcal{G}^1$. $\square$

Следующая задача была решена в статье Форе [38] для случая AB-грамматик с однозначным присвоением типов.

Теорема 3.2.8 (Форе [38]). *Задача определения для произвольных AB-грамматик с однозначным присвоением типов G_1 и G_2, того, является ли пересечение языков $L(G_1) \cap L(G_2)$ пустым, алгоритмически неразрешима.*

Если внимательно изучить доказательство, можно заметить, что рассматриваемые AB-грамматики являются грамматиками из класса $\mathcal{G}^1$, следовательно, это простое замечание позволяет доказать аналогичную теорему для данного формализма прямой ссылкой.

Однако конструкция в статье Форе будет иметь место для фиксированного алфавита из $2n(n+1)+3$ символов. Здесь n – число, при котором задача проверки существования решения по произвольной системе соответствий Поста с количеством компонент n остаётся неразрешимой. По теореме 3.2.2 данная задача неразрешима при $n = 5$, следовательно, теорема 3.2.8 остаётся верной при фиксированном алфавите из 63 символов.

В сравнении, представленная в доказательстве теоремы 3.2.9 конструкция для случая фиксированного алфавита будет верна при $n+6$ символах, т.е. данная теорема верна при алфавите из 11 символов. Это усиливает теорему 3.2.8, так как произвольная грамматика из класса $\mathcal{G}^1$ является AB-грамматикой с однозначным присвоением типов.

Теорема 3.2.9. *Задача определения для произвольных грамматик G_1 и G_2 из класса $\mathcal{G}^1$, где количество символов в алфавитах обеих грамматик равно 11, того, является ли пересечение языков $L(G_1) \cap L(G_2)$ пустым, алгоритмически неразрешима.*

Доказательство. Основная идея заключается в том, чтобы немного изменить язык $L(G_1)$ из доказательства теоремы 3.2.7. Однако так как G_1 должна быть из класса $\mathcal{G}^1$, выбрасыванием части слов сделать это не получится.

Зафиксируем произвольную систему соответствий Поста $(\vec{x}, \vec{y})$, где количество компонент равно 5. Рассмотрим задачу о проверке существования решения вида $(1, i_1, \ldots, i_k)$ данной системы соответствий Поста. Данная задача также алгоритмически неразрешима, более того, для сведения в [42, теорема 11] произвольное решение имеет именно такой вид.

Гомоморфизмы α_1, α_2, γ возьмём из доказательства теоремы 3.2.7 и обозначим через Σ_1 алфавит $\{c_0, c_1, c_2, c_3, c_4, c_5, a_1, b_1, a_2, b_2, d\}$.

Пусть строка x_1 равна $t_1 \ldots t_k$, а строка y_1 равна $\widetilde{t_1} \ldots \widetilde{t_m}$. Грамматику G_1 зададим в алфавите Σ_1 с результирующим типом r и следующим набором присвоений:

$$\{c_0 \triangleright \alpha_2(\widetilde{t_m}) \backslash \ldots \backslash \alpha_2(\widetilde{t_1}) \backslash r_0 \backslash \alpha_1(t_1) \backslash \ldots \backslash \alpha_1(t_k) \backslash r\} \cup$$
$$\{c_i \triangleright A_{x_i,y_i} \mid i \in \{1, \ldots, 5\}\} \cup \{a_1 \triangleright p_1, b_1 \triangleright q_1\} \cup$$
$$\{a_2 \triangleright p_2, b_2 \triangleright q_2\} \cup \{d \triangleright r_0\}.$$

Здесь мы расширяем грамматику G_1 из доказательства теоремы 3.2.7, добавляя правило для c_0. Неформально, чтобы строка типов редуцировалась к r, сначала тип, присвоенный c_0, должен редуцироваться со строкой $\alpha_2(x_1); r_0; \alpha_1(y_1)$ к типу r. Далее, все редукции должны быть устроены так же, как в грамматике G_1. Иначе говоря, последовательно каждый тип вида A_{x_i,y_i} должен редуцироваться со строкой типов $\alpha_2(x_i); r; \alpha_1(y_i)$ к типу r. Преобразуя G_1 в контекстно-свободную форму, получим:

$$\{r \to (\alpha_1(x_1))^R r_0 (\alpha_2(y_1)) c_0\} \cup \{r \to (\alpha_1(x_i))^R r (\alpha_2(y_i)) c_i \mid i \in \{1, \ldots, 5\}\} \cup$$
$$\{p_1 \to a_1, q_1 \to b_1\} \cup \{p_2 \to a_2, q_2 \to b_2\} \cup \{r_0 \to d\}.$$

Как в доказательстве теоремы 3.2.7, легко показать, что порожденный язык $L(G_1)$ равен $\left\{x_{i_k}^R \ldots x_{i_1}^R x_1^R d y_1 c_0 y_{i_1} c_{i_1} \ldots y_{i_k} c_{i_k} \mid k \in \mathbb{N},\ i_j \in \{1, \ldots, 5\}\right\}$.

Грамматику G_2 зададим в алфавите Σ_1 с результирующим типом r:

$$\{c_i \triangleright r \backslash r \mid i \in \{0, \ldots, 5\}\} \cup \{d \triangleright r\} \cup \{a_1 \triangleright r/p_1,\ b_1 \triangleright r/p_2,\ a_2 \triangleright r \backslash p_1,\ b_2 \triangleright r \backslash p_2\}.$$

Данная грамматика задаёт язык $L(G_2)$ равный:

$$\{udv \mid u \in \{a_1, b_1\}^*,\ v \in (\{a_2, b_2\} \cup \{c_i \mid i \in \{0, \ldots, n\}\})^*,\ \gamma(v) = u^R\}.$$

Пересечение $L(G_1) \cap L(G_2)$ непусто тогда и только тогда, когда существует решение системы соответствий Поста $(1, i_1, \ldots, i_k)$. □

Этот результат является одним из основных в диссертации.

3.3. AB-грамматики с однозначным присвоением типов

В данном параграфе мы рассматриваем алгоритмические свойства AB-грамматик с однозначным присвоением типов. Ранее мы доказали в теореме 2.3.18, что данный формализм может закодировать посредством гомоморфизмов произвольный контекстно-свободный язык. Однако в отличии от теоремы 2.4.2 для случая грамматик из класса $\mathcal{G}^2$, кодирование AB-грамматик с однозначным присвоением типов происходит «в одну сторону».

Тем не менее, часть открытых задач данное кодирование позволяет решить. Одним из непосредственных следствий теоремы 2.3.18 является усиленный вариант теоремы Грейбах [26] о самом трудном языке.

Следствие 3.3.1. *Существуют алфавит Ω и язык L_0, заданный AB-грамматикой с однозначным присвоением типов, такие, что для произвольного контекстно-свободного языка $L \subseteq \Sigma^+$ верно $L \leqslant L_0$.*

Теорема 2.3.18 позволяет решать и другие задачи. Следующая теорема отвечает на вопрос, поставленный в статье [38], и является одним из основных результатов диссертации.

Теорема 3.3.2. *Задача определения для произвольных AB-грамматик с однозначным присвоением типов G_1 и G_2 того, что $L(G_1) \subseteq L(G_2)$, алгоритмически неразрешима.*

Доказательство. Зафиксируем язык $M'_{\vec{x}\vec{y}} = \{d\} \cdot (\{a, b, c\}^* - (L_{\vec{x}} \cap L_{\vec{y}}))$ из определения 3.2.4. Пусть g и $G_2 = (\Sigma, \Pr, S, \triangleright)$ — соответственно гомоморфизм и грамматика, полученные для данного языка по теореме 2.3.18. В алфавите грамматики G_2 четыре вида букв: входящие в строку $g(a) = a_1 \dots a_k$, в $g(b) = b_1 \dots b_m$, и $g(c) = c_1 \dots c_n$, и в $g(d) = d_l \dots d_1$. Отметим, что, по построению в теореме 2.3.18, можно считать, что в образе каждой буквы не менее трёх символов, иначе в исходной грамматике, задающей $M'_{\vec{x}\vec{y}}$, каждой из букв можно присвоить дополнительные примитивные типы.

В качестве грамматики G_1 положим грамматику с результирующим примитивным типом r и следующими наборами присвоений (p_i, q_i, t_i, r_i — примитивные типы):

$$\{a_1 \triangleright r \backslash p_1\} \cup \{a_i \triangleright p_{i-1} \backslash p_i \mid i \in \{2, \dots, k-1\}\} \cup \{a_k \triangleright p_{k-1} \backslash r\}$$
$$\{b_1 \triangleright r \backslash q_1\} \cup \{b_i \triangleright q_{i-1} \backslash q_i \mid i \in \{2, \dots, m-1\}\} \cup \{b_m \triangleright q_{m-1} \backslash r\}$$
$$\{c_1 \triangleright r \backslash t_1\} \cup \{c_i \triangleright t_{i-1} \backslash t_i \mid i \in \{2, \dots, n-1\}\} \cup \{c_n \triangleright t_{k-1} \backslash r\}$$
$$\{d_l \triangleright r / r_{l-1}\} \cup \{d_i \triangleright r_i / r_{i-1} \mid i \in \{2, \dots, l-1\}\} \cup \{d_1 \triangleright r_1\}.$$

Заметим, что данная AB-грамматика с однозначным присвоением типов является базовой категориальной грамматикой. По теореме 1.3.3 эквивалентная контекстно-свободная грамматика имеет следующий набор правил:

$$\{p_1 \to r\, a_1\} \cup \{p_i \to p_{i-1}\, a_i \mid i \in \{2, \dots, k-1\}\} \cup \{r \to p_{k-1}\, a_k\}$$
$$\{q_1 \to r\, b_1\} \cup \{q_i \to q_{i-1}\, b_i \mid i \in \{2, \dots, m-1\}\} \cup \{r \to q_{m-1}\, b_m\}$$
$$\{t_1 \to r\, c_1\} \cup \{t_i \to t_{i-1}\, c_i \mid i \in \{2, \dots, n-1\}\} \cup \{r \to t_{k-1}\, c_n\}$$
$$\{r \to d_l r_{l-1}\} \cup \{r_i \to d_i r_{i-1} \mid i \in \{2, \dots, l-1\}\} \cup \{r_1 \to d_1\}.$$

Эквивалентная грамматика имеет следующий вид:

$$r \to r\, a_1 \dots a_k \mid r\, b_1 \dots b_m \mid r\, c_1 \dots c_n \mid d_l \dots d_1.$$

Значит исходная грамматика G_1 задаёт язык:

$$(d_l \dots d_1) \cdot (a_1 \dots a_k + b_1 \dots b_m + c_1 \dots c_n)^*.$$

Итого, $L(G_1) \subseteq L(G_2)$ равносильно тому, что:

$$(d_l \dots d_1) \cdot (a_1 \dots a_k + b_1 \dots b_m + c_1 \dots c_n)^* \subseteq L(G_2).$$

По построению грамматики G_2 слово $g(dw)$, где $w \in \{a, b, c\}^*$, принадлежит $L(G_2)$ тогда и только тогда, когда слово $dw \in M'_{\vec{x}\vec{y}}$, то есть тогда и только тогда, когда $w \in \{a, b, c\}^* - (L_{\vec{x}} \cap L_{\vec{y}})$. Следовательно, $L(G_1) \subseteq L(G_2)$ тогда и только тогда, когда $\{a, b, c\}^* \subseteq \{a, b, c\}^* - (L_{\vec{x}} \cap L_{\vec{y}})$. Иначе говоря, $L_{\vec{x}} \cap L_{\vec{y}} = \emptyset$, то есть тогда и только тогда, когда система соответствий Поста для пары $(\vec{x}, \vec{y})$ не имеет решения. $\square$

Заключение

В первой главе работы введены базовые категориальные грамматики с условием присвоения не более k типов одному символу. Доказана теорема об иерархии по параметру k для классов задаваемых языков.

Во второй главе доказана незамкнутость класса $\mathcal{G}^1$ относительно большинства естественных операций на языках. Затем рассмотрены AB-грамматики с однозначным присвоением типов. Доказана возможность сведения любого контекстно-свободного языка к языку, заданному данным формализмом. В частности, разработана техника кодирования множественного присвоения типов за счёт единственной строки. Однако, техника нетривиальна, и пока неизвестно, существует ли более простой способ. Более того, имеются основания полагать, что, в силу крайне ограниченных возможностей, подобные конструкции будут настолько же технически сложными. Доказано, что произвольный контекстно-свободный язык эквивалентен по сводимости некоторому языку из класса $\mathcal{L}^2$.

В третьей главе рассматриваются алгоритмические свойства грамматических формализмов, введённых в первой главе. Доказано, что для грамматик из класса $\mathcal{G}^2$ неразрешимы классические алгоритмические задачи. Дальше доказано, что задача проверки пустоты пересечения языков, заданных грамматиками из класса $\mathcal{G}^1$, неразрешима. Более того, последнее верно при фиксированном алфавите из 11 символов. В третьем параграфе рассматриваются AB-грамматики с однозначным присвоением типов. Используя теорему из второй главы, доказывается, что задача проверки вложения языков, заданных грамматиками из данного класса, неразрешима.

Стоит отметить, что, хотя и доказан ряд теорем для класса $\mathcal{G}^1$, до сих пор остаются открытыми многие вопросы, касающиеся данного формализма. Например, неизвестно, разрешима ли задача проверки включения языков, порожденных данными грамматиками. Предполагается, что техника, разработан-

ная в данной диссертации, поможет решить эту задачу.

Методы, разработанные во второй главе, применимы в большинстве грамматических формализмов, основанных на AB-грамматиках. Следовательно, последующие работы могут использовать полученный подход для исследований. Так, результаты о сложности в более богатых исчислениях могут быть доказаны напрямую, используя разработанную конструкцию кодирования.

Список литературы

[1] Chomsky N. Three models for the description of language // IRE Transactions on Information Theory. — 1956. — Vol. 2, N. 3. — P. 113–124.

[2] Aho A., Sethi R., Ullman J. Compilers: Principles, Techniques and Tools. — Reading, Mass.: Addison-Wesley, 1985.

[3] Carpenter B. Type-Logical Semantics. — MIT Press, 1998.

[4] Мельчук И.А. Опыт теории лингвистических моделей «Смысл ⇔ Текст» — М.: Изд-во «Наука», Главная редакция восточной литературы, 1974. — 314 с.

[5] Ajdukiewicz K. Die syntaktische Konnexität // Studia Philosophica. — 1935. — Bd. 1 — S. 1–27.

[6] Bar-Hillel Y. A quasi-arithmetical notation for syntactic description // Language. — 1953. — Vol. 29, N. 1. — P. 47–58.

[7] Bar-Hillel Y., Gaifman H., Shamir E. On categorial and phrase structure grammars // Bulletin of the Research Council of Israel. — 1960. — Vol. 9F — P. 155–166.

[8] Lambek J. The mathematics of sentence structure // The American Mathematical Monthly. — 1958. — Vol. 65, N. 3. — P. 154–170.

[9] Lambek J. On the calculus of syntactic types // In: R.Jakobson (Ed.), Structure of Language and Its Mathematical Aspects. / Proceedings of Symposia in Applied Mathematics — Providence, R.I. : AMS, 1961. — Vol. 12 — P. 166–178.

[10] Pentus M. Lambek grammars are context free // In: Proceedings Eighth Annual IEEE Symposium on Logic in Computer Science. – IEEE, 1993. – P. 429–433.

[11] Dikovsky A. Dependencies as categories // In: Proceedings of the Workshop on Recent Advances in Dependency Grammar. — 2004. — P. 82–89.

[12] Dekhtyar M., Dikovsky A. Categorial dependency grammars // In: M. Moortgat, V. Prince (Eds.), Proceedings of the International Conference on Categorial Grammars, Montpellier, 2004. — P. 76–91.

[13] Dekhtyar M., Dikovsky A. Generalized categorial dependency grammars // Pillars of Computer Science: Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday. – Berlin, Heidelberg : Springer Berlin Heidelberg, 2008. — P. 230–255.

[14] Dekhtyar M., Dikovsky A., Karlov B. Categorial dependency grammars // Theoretical Computer Science. — 2015. — Vol. 579. — P. 33–63.

[15] Steedman M. The Syntactic Process. — MIT Press, 2001. — 348 pp.

[16] Kuznetsov S., Okhotin A. Conjunctive categorial grammars // Proceedings of the 15th Meeting on the Mathematics of Language. — ACL, 2017. — P. 140–151.

[17] Sakai I. Syntax in universal translation // In: Proceedings of the International Conference on Machine Translation and Applied Language Analysis. Her Majesty's Stationery Office, London, 1962. — P. 593–608.

[18] Семёнов А.Л. Алгоритмические проблемы для степенных рядов и контекстно-свободных грамматик // Доклады Академии наук СССР. — 1973. — Т. 212, № 1. — С. 50–52.

[19] Greibach S. A note on undecidable properties of formal languages // Mathematical Systems Theory. — 1968. — Vol. 2, N. 1. — P. 1–6.

[20] Гладкий А.В. Алгоритмическая нераспознаваемость существенной неопределенности контекстно-свободных языков // Алгебра и логика. — 1965. — Т. 4, № 4. — С. 53–63.

[21] Bar-Hillel Y., Perles M., Shamir E. On formal properties of simple phrase structure grammars // In: Language and Information: Selected Essays on their Theory and Application, 1964. — P. 116–150.

[22] Гладкий А.В. Формальные грамматики и языки. — М.: Изд-во «Наука» Главная редакция физико-математической литературы, 1973. — 368 с.

[23] Kozen D.C. Automata and Computability. — Springer Science & Business Media, 1997.

[24] Chomsky N., Schützenberger M. The algebraic theory of context-free languages // Computer Programming and Formal Systems. — 1963. — Vol. 35. — P. 118–161.

[25] Okhotin A. Non-erasing variants of the Chomsky–Schützenberger theorem // In: International Conference on Developments in Language Theory. – Berlin, Heidelberg : Springer Berlin Heidelberg, 2012. – P. 121–129.

[26] Greibach S. The hardest context-free language // SIAM Journal on Computing. — 1973. — Vol. 2, N. 4. — P. 304–310.

[27] Greibach S. Jump PDA's and hierarchies of deterministic context-free languages // SIAM Journal on Computing. — 1974. — Vol. 3, N. 2. — P. 111–127.

[28] Boasson L., Nivat M. Le cylindre des langages linéaires // Mathematical Systems Theory. — 1977. — Vol. 11. — P. 147–155.

[29] Сафиуллин А. Н. Выводимость допустимых правил с простыми посылками в исчислении Ламбека // Вестник Московского университета. Серия 1: Математика. Механика. — 2007. — № 4. — С. 72–76.

[30] Kuznetsov S. The Lambek calculus with iteration: two variants // International Workshop on Logic, Language, Information, and Computation. – Berlin, Heidelberg : Springer Berlin Heidelberg, 2017. — P. 182–198.

[31] Kuznetsov S. Complexity of the infinitary Lambek calculus with Kleene star // The Review of Symbolic Logic. — 2021. — Vol. 14, N. 4. — P. 946–972.

[32] Buszkowski W. Solvable problems for classical categorial grammars // Bulletin of the Polish Academy of Sciences: Mathematics. — 1987. — Vol. 35. — P. 373–382.

[33] Kanazawa M. Indentification in the Limit of Categorial Grammars // Journal of Logic, Language and Information. — 1996. — Vol. 5, N. 2. — P. 115–155.

[34] Buszkowski W. Discovery procedures for categorial grammars // In: J. van Benthem and E. Klein (Eds.), Categories, Polymorphism and Unification — Universiteit van Amsterdam, 1987. — С. 35–64.

[35] Buszkowski W., Penn G. Categorial grammars determined from linguistic data by unification // Studia Logica. — 1990. — Vol. 49. — P. 431–454.

[36] Gold E.M. Language identification in the limit // Information and Control. — 1967. — Vol. 10, N. 5. — P. 447–474.

[37] Greibach S. A new normal-form theorem for context-free phrase structure grammars // Journal of the ACM. — 1965. — Vol. 12, N. 1. — P. 42–52.

[38] Foret A. The emptiness of intersection problem for languages of k-valued categorial grammars (classical and Lambek) is undecidable // Electronic Notes in Theoretical Computer Science. — 2004. — Vol. 53. — P. 81–93.

[39] van Benthem J. Language in Action. Categories, Lambdas and Dynamic Logic. — Amsterdam: North-Holland, 1991. — (Studies in Logic and the Foundations of Mathematics; Vol. 130).

[40] Pentus M. Equivalent Types in Lambek Calculus and Linear Logic // Preprint No. 2 of the Department of Mathematical Logic of the Math. V.A. Steklov Institute of RAS. Series: Mathematical Logic and Theoretical Informatics. — 1992. — 21 pp.

[41] Post E.L. A variant of a recursively unsolvable problem // Bulletin of the American Mathematical Society. — 1946. — Vol. 52, N. 4. — P. 264–269.

[42] Neary T. Undecidability in binary tag systems and the Post correspondence problem for five pairs of words // In: Proceedings of the 32nd International Symposium on Theoretical Aspects of Computer Science (STACS 2015). – Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2015. – Vol. 30. – P. 649–661.

[43] Пентус А.Е., Пентус М.Р. Теория формальных языков: Учебное пособие. — М.: Изд-во ЦПИ при мех.-мат. ф-те МГУ, 2004. — 80 с.

Публикации автора по теме диссертации

[44] Вишникин М.Е. Базовые категориальные грамматики с однозначным присвоением типов // Вестник Московского университета. Серия 1: Математика. Механика. — 2022. — № 2. — С. 64–67.

[45] Вишникин М.Е. Базовые категориальные грамматики с ограничением на количество присваиваемых категорий // Сибирские электронные математические известия. — 2025. — Т. 22, № 1. — С. 343–353.

[46] Вишникин М.Е. Алгоритмические свойства базовых категориальных грамматик с однозначным присвоением категорий // Доклады Россий-

ской академии наук. Математика, информатика, процессы управления. — 2025. — Т. 523.

Тезисы докладов

[47] Вишникин М.Е., Охотин А.С. Категориальные грамматики с однозначным присвоением категорий // тез. докл. в эл. сборнике Международной конференции «Мальцевские чтения», Новосибирск, 2024.

[48] Вишникин М.Е. Базовые категориальные грамматики с однозначным присвоением типов // тез. докл. в сборнике Материалы Международного молодежного научного форума «Ломоносов-2021», М.: МАКС Пресс, 2021.

[49] Вишникин М.Е. Базовые категориальные грамматики с ограничением присваиваемые категории // тез. докл. в сборнике Материалы Международного молодежного научного форума «Ломоносов-2025», М.: МАКС Пресс, 2025.