Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Томский государственный университет систем управления и радиоэлектроники»
(ТУСУР)

На правах рукописи

Кручинин Дмитрий Владимирович

МЕТОДЫ, АЛГОРИТМЫ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ НА ОСНОВЕ ПРОИЗВОДЯЩИХ ФУНКЦИЙ МНОГИХ ПЕРЕМЕННЫХ ДЛЯ КОМПЛЕКСНОГО ИССЛЕДОВАНИЯ ИНФОРМАЦИОННЫХ ОБЪЕКТОВ

Специальность 05.13.17 — «Теоретические основы информатики»

Диссертация на соискание учёной степени доктора технических наук

Научный консультант:
доктор технических наук, доцент
Рулевский Виктор Михайлович

Томск — 2022

Оглавление

Введение

Актуальность темы. Цифровизация является глобальным вызовом для современного мира. От успешного решения этой проблемы зависит развитие стран и глобальных объединений. Для развития этого направления в Российской Федерации принята национальная программа «Цифровая экономика Российской Федерации», утвержденная протоколом заседания президиума Совета при Президенте Российской Федерации по стратегическому развитию и национальным проектам от 4 июня 2019 г. В рамках данной программы имеются проекты «Цифровые технологии» и «Искусственный интеллект». Следовательно, развитие информационных технологий является важной и насущной задачей поступательного движения Российской Федерации в области цифровизации.

Развитие информационных технологий имеет тенденцию к существенному его усложнению, увеличению сроков разработки и стоимости. Поэтому особое значение принимает направление развития информационных технологий, связанное с построением технологий разработки программного обеспечения и инструментальных средств. Для разработки таких технологий необходимо развитие математических основ, методов построения алгоритмов и баз знаний. Более того, интенсивное развитие информационных технологий на фоне формирования цифровой экономики приводит к экспоненциальному росту объемов различного рода данных, которые носят распределенный характер. Таким образом, возникает потребность в разработке эффективных систем хранения, передачи и обработки больших объемов данных. Для оперативной работы с такими данными требуется алгоритмический инструментарий, который позволит структурировать и систематизировать информацию о различных объектах.

Информационным объектом будем считать описание реального объекта, явления, процесса, события в виде совокупности логически связанных характеристик (информационных элементов). Существуют количественные и качественные характеристики информационного объекта. Особое значение для построения и анализа методов представления информационных объектов имеет теория производящих функций, поскольку производящие функции являются описательной характеристикой информационных объектов, заключающейся в подсчете числа объектов, принадлежащих некоторому семейству конечных множеств. Основная идея производящих функций заключается в отображении исследуемых множеств

информационных объектов в множество степенных рядов и последующей работе с ними при помощи развитого аппарата функционального анализа.

Поскольку многим информационным объектам, в том числе характеризующимся большим объемом данных, свойственна иерархическая или рекурсивная природа их описания, то существует альтернативный способ представления множества таких информационных объектов в форме комбинаторного множества. Общие и универсальные методы, направленные на исследование таких представлений и разработку соответствующих алгоритмов комбинаторной генерации, представлены в работ таких ученых, как Э.М. Рейнгольд, Д.Л. Крехер, Е. Баркуччи, С. Баччелли, А. Дель Лунго, В. Вайновски, Ф. Флажоле, К. Мартинес, К. Мулинеро, Б.Я. Рябко, Ю.С. Медведева и другие. Если рассмотреть последовательность значений функции мощности для заданного комбинаторного множества, то данная последовательность может быть представлена с помощью выражения производящей функции. Таким образом, возникает ситуация, когда для функции мощности комбинаторного множества не известно явное выражение, но имеется представление в виде производящей функции. Тогда, чтобы определить явное выражение для функции мощности комбинаторного множества, необходимо получить явное выражение для коэффициентов соответствующей ей производящей функции. Однако существует огромное количество комбинаторных множеств, которые определяются более чем одним параметром. В таком случае для получения явных выражений функций мощности комбинаторных множеств необходимо уже оперировать производящими функциями многих переменных.

Кроме того, исследования производящих функций находят свое применение в рамках решения следующих значимых и актуальных научных задач: развитие методов индексации и поиска сложных информационных объектов, оптимизация на сложных дискретных структурах, создание новых способов представления сложных дискретных структур (например, в области биоинформатики и хемоинформатики), развитие методов представления полиномов и другие. В настоящее время наблюдается значительный прогресс в области исследований производящих функций, в первую очередь связанный с исследованиями свойств полиномов, заданных производящими функциями. Можно выделить работы следующих ученых: А. Эрдели, Р.П. Боас и Р.К. Бак, С. Роман, Э.Б. Макбрайд, Х.М. Шривастава, Х.Л. Маноча, Т. Ким, Й. Шимшек, Я.Л. Геронимус, Н.Н. Лебедев, П.К. Суетин, В.В. Прасолов.

Математический аппарат производящих функций активно применяется при решении задач из области дискретной математики и информатики, так как производящие функции позволяют получить компактное представление дискретных структур, а также предоставляют широкий набор функциональных возможностей при работе с такими структурами. Согласно Р.П. Стенли, самый полезный метод численного представления функции $f(n)$, считающей число элементов в конечном множестве, — это метод производящих функций. Метод производящих функций позволяет не только посчитать количество элементов в конечном множестве, но и получить различные комбинаторные и практические приложения за счет выполнения операций над производящими функциями. В свою очередь Р.Л. Грэхем, Д.Э. Кнут и О. Паташник в своей книге «Конкретная математика. Математические основы информатики» в качестве ключевой идеи книги выделяют именно понятие производящих функций. Также стоит выделить работы следующих ученых: Г. Эндрюс, Д. Риордан, Ф. Флажоле, К. Краттенталер, Д. Фоата, Г.С. Уилф, Л. Комте, И.М. Гессель, М. Дрмота. В отечественной литературе значительное внимание теория производящих функций получила в работах Н.Я. Виленкина, К.А. Рыбникова, С.А. Ландо, В.Н. Сачкова, Г.П. Егорычева, О.В. Кузьмина и других ученых.

Основные разработки в данной области исследования касаются класса производящих функций одной переменной. Однако существует достаточно большое количество объектов, описываемых производящими функциями двух и более переменных (например, это могут быть как классические объекты — числа Стирлинга или биномиальные коэффициенты, так и новые — вероятность наличия направленного вверх ребра в графе для ацтекского бриллианта размерности n). Для производящих функций многих переменных соответствующий математический аппарат развит слабо, а в общем виде применение коэффициентов степеней производящих функций многих переменных исследовано недостаточно. Поэтому разработка новых методов и программного обеспечения, основанных на производящих функциях многих переменных, существенно расширит возможности использования производящих функций как для развития самой теории производящих функций, так и для решения прикладных задач.

Цели и задачи исследования. Целью диссертационной работы является повышение эффективности методов преобразования информации в данные и знания за счет применения аппарата производящих функций многих переменных и их реализации в программных средствах автоматизации.

Для достижения поставленной цели были сформулированы следующие задачи:

1. Провести аналитический обзор современного состояния исследований в области теории производящих функций и методов на основе их применения, в том числе для задач описания информационных объектов и построения алгоритмов комбинаторной генерации;

2. Разработать методы оперирования коэффициентами степеней производящих функций многих переменных и применить их для получения методов вычисления явных выражений функции мощности множеств, решения функциональных уравнений, описания полиномов, числовых треугольников и решеточных путей;

3. Модифицировать метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ с учетом описания функции мощности множества производящей функцией многих переменных, а также с применением приближенных вычислений и двоичного поиска для поиска выбранного сына ИЛИ-узла;

4. Разработать алгоритмы комбинаторной генерации для формирования информационных объектов, представленных следующими комбинаторными множествами: множество сочетаний из n по m с применением колексикографического порядка; множество самонепересекающихся решеточных путей на плоскости; множество помеченных путей Дика длины $2n$ с m подъемами на возвратных шагах; множество путей Дика с пиками; множество последовательностей вариантов ответа на тест с вопросами закрытого типа; множество исходов турнира на выбывание; множество частей круга, полученных при его разрезе прямыми линиями; множество правильных скобочных последовательностей, разряженных нулями; множество разбиений множества;

5. Создать базу знаний производящих функций двух переменных и реализовать ее в виде автоматизированной электронной энциклопедии числовых пирамид;

6. Разработать метод построения критериев простоты числа на основе методов оперирования коэффициентами степеней производящих функций;

7. На основе разработанных методов и алгоритмов создать программное обеспечение для определения коэффициентов степеней производящих функций и для генерации по рангу элементов комбинаторных множеств в виде библиотек к математическим пакетам Maxima и Mathematica;

8. Внедрить полученные методы, алгоритмы и программное обеспечение для решения задач, связанных с построением проверочных выборок для тестирования систем различного класса.

Объект исследования. Объектом исследования является процесс формирования информационных объектов на основе производящих функций многих переменных.

Предмет исследования. Предметом исследования являются методы и алгоритмы формирования информационных объектов на основе производящих функций многих переменных.

Методы исследования. В диссертационной работе применялись методы оперирования производящими функциями одной переменной, построения алгоритмов комбинаторной генерации, анализа алгоритмов и объектно-ориентированного программирования.

Научная новизна полученных результатов:

1. Предложен комплексный метод формирования информационных объектов, основанный на k-й степени производящих функций, отличающийся наличием правил преобразования коэффициентов степеней взаимных, обратных и композиции производящих функций многих переменных;

2. Предложена модификация метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, которая отличается применением предложенного комплексного метода для нахождения выражения функции мощности комбинаторного множества, а также применением приближенных вычислений и двоичного поиска для определения выбранного сына ИЛИ-узла для задач генерации информационных объектов;

3. Разработаны новые алгоритмы ранжирования и генерации по рангу для множества информационных объектов, обладающие меньшей вычислительной сложностью;

4. Сформулирован подход к созданию базы знаний производящих функций двух переменных и реализован в виде электронной энциклопедии, обеспечивающей автоматизированный поиск и манипулирование матричными представлениями соответствующих функций;

5. Сформулирован подход к созданию программных систем компьютерной алгебры и систем тестирования, отличающийся применением коэффициентов степеней производящих функций, представленных в явном или матричном виде.

Теоретическая значимость работы. Теоретическая значимость результатов диссертационной работы заключается в развитии теории производящих функций за счет разработки комплексного метода формирования информационных объектов, основанного на k-й степени производящих функций и их коэффициентов. Развиты методы построения алгоритмов комбинаторной генерации за счет использования разработанных методов оперирования производящими функциями и предварительного определения ветви дерева И/ИЛИ. Предложенный на основе применения коэффициентов степеней производящих функций подход к созданию программных систем компьютерной алгебры и систем тестирования является теоретической основой для развития новых технологий проектирования программного обеспечения и решения задач индексации и поиска сложных информационных объектов, для организации новых способов хранения информации и модернизации принципов работы систем баз данных, для создания новых способов представления сложных дискретных структур в конкретных прикладных задачах (например, в области биоинформатики и хемоинформатики).

Практическая значимость работы. Практическая значимость работы заключается в создании методов и программного обеспечения, ускоряющего процесс формирования входных последовательностей для тестирования сложных информационных и программных объектов. Разработанное программное обеспечение в виде библиотек для систем компьютерной алгебры Maxima и Mathematica позволяет решать задачи, отсутствующие в перечне стандартных функций математических пакетов. Применение разработанного программного обеспечения ускоряет процесс вычислений при работе с производящими функциями. Созданная база знаний и ее реализация в виде электронной энциклопедии расширяют возможности проведения исследований числовых пирамид.

Практическая значимость результатов диссертационной работы подтверждается их внедрением в деятельность научно-производственных предприятий. При внедрении в деятельность «НИИ АЭМ ТУСУР» и в деятельность АО «Информационные спутниковые системы» имени академика М. Ф. Решетнёва» использование разработанных алгоритмов и программного обеспечения позволило снизить время формирования выходных характеристик имитаторов энергопреобразующей аппаратуры на 43%, что привело к сокращению времени тестирования систем энергообеспечения. При внедрении в деятельность ООО «ПлантаПлюс» использование разработанных алгоритмов позволило сократить объем базы данных на 9% за счет уменьшения количества дублируемой информации и повысить скорость поиска и

обработки данных на 5%. При внедрении в деятельность ООО «Эль Контент» использование разработанных алгоритмов и программного обеспечения позволило уменьшить временные затраты на 50% во время тестирования систем тестового оценивания. При внедрении в учебный процесс ФГБОУ ВО «ТУСУР» в рамках создания автоматизированной системы обучения математическим дисциплинам использование разработанного программного обеспечения позволило сократить временные затраты на создание и проверку контрольных и домашних работ за счет автоматизации данного процесса.

Основные этапы диссертационного исследования выполнены в рамках государственных заданий, а также грантов РФФИ и РНФ. Получено четыре свидетельства о регистрации программ для ЭВМ. Получены акты о внедрении результатов исследования в учебный процесс и в деятельность коммерческих организаций, специализирующихся на создании и эксплуатации программного обеспечения.

Положения, выносимые на защиту:

1. Разработанный комплексный метод формирования информационных объектов, основанный на k-й степени производящих функций, отличающийся от существующих наличием правил преобразования коэффициентов степеней взаимных, обратных и композиции производящих функций многих переменных, позволяет получить их явные представления, обладающие меньшей вычислительной сложностью.

Соответствует п. 3 паспорта специальности 05.13.17: Исследование методов и разработка средств кодирования информации в виде данных. Принципы создания языков описания данных, языков манипулирования данными, языков запросов. Разработка и исследование моделей данных и новых принципов их проектирования;

2. Модифицированный метод построения алгоритмов комбинаторной генерации, отличающийся применением приближенных вычислений и двоичного поиска для определения выбранного сына ИЛИ-узла для задач генерации информационных объектов и применением предложенного комплексного метода для задач поиска функций мощности, позволяет строить алгоритмы комбинаторной генерации с меньшей вычислительной сложностью, в том числе для более сложных информационных объектов, описываемых производящими функциями многих переменных.

Соответствует п. 3 паспорта специальности 05.13.17: Исследование методов и разработка средств кодирования информации в виде данных. Принципы создания языков описания данных, языков манипулирования данными, языков запросов. Разработка и исследование моделей данных и новых принципов их проектирования;

3. Разработанные алгоритмы комбинаторной генерации для множеств информационных объектов, описываемых производящими функциями, позволяют генерировать информационные объекты с меньшей вычислительной сложностью.
Соответствует п. 3 паспорта специальности 05.13.17: Исследование методов и разработка средств кодирования информации в виде данных. Принципы создания языков описания данных, языков манипулирования данными, языков запросов. Разработка и исследование моделей данных и новых принципов их проектирования;

4. Построенная база знаний производящих функций двух переменных, основанная на фреймовой модели, позволяет автоматизировать процесс поиска и оперирования коэффициентами степеней производящих функций двух переменных.
Соответствует п. 4 паспорта специальности 05.13.17: Исследование и разработка средств представления знаний. Принципы создания языков представления знаний, в том числе для плохо структурированных предметных областей и слабоструктурированных задач; разработка интегрированных средств представления знаний, средств представления знаний, отражающих динамику процессов, концептуальных и семиотических моделей предметных областей;

5. Предложен подход к созданию программных систем компьютерной алгебры и систем тестирования. Отличительная особенность предложенного подхода заключается в применении коэффициентов степеней производящих функций, что позволяет решать следующие задачи: находить явные выражения коэффициентов композиции производящих функций, строить алгоритмы вычисления матричных представлений обратных и взаимных производящих функций, строить критерии простоты числа и формировать тестовые выборки.
Соответствует п. 14 паспорта специальности 05.13.17: Разработка теоретических основ создания программных систем для новых информационных технологий.

Достоверность результатов. Достоверность результатов диссертационной работы обеспечивается корректностью применения математических методов, осно-

ванных на теории производящих функций, сравнением разработанных алгоритмов с известными алгоритмами, полученными исследователями других научных групп, проверкой теоретических положений вычислительными экспериментами, положительным эффектом от внедрения полученных результатов.

Внедрение результатов работы. Результаты диссертационной работы использованы в ходе выполнения следующих научно-исследовательских работ:

– грант «Российского фонда фундаментальных исследований» (проект № 12-01-09350 «Метод получения выражений полиномов на основе композиции производящих функций» на 2012 г. — Руководитель);

– государственное задание Министерства образования и науки РФ (проект № 01201274654 «Разработка и исследование методов и технологий информационной безопасности в технических и высокопроизводительных вычислительных системах» на 2012–2014 гг. — Исполнитель);

– государственное задание Министерства образования и науки РФ (проект № 114030440055 «Фундаментальные основы проектирования информационно-безопасных систем» на 2014 г. — Исполнитель);

– стипендия Президента РФ для молодых ученых и аспирантов, осуществляющих перспективные научные исследования и разработки по приоритетным направлениям модернизации российской экономики по направлению «Стратегические информационные технологии, включая вопросы создания суперкомпьютеров и разработки программного обеспечения» (научно-исследовательская работа по теме «Разработка математического и алгоритмического обеспечения систем криптографической защиты информации на основе аппарата производящих функций» на 2015–2017 гг. — Руководитель);

– государственное задание Министерства образования и науки РФ (проект № АААА-А15-115120110047-7 «Разработка и исследование методов построения информационно-безопасных систем» на 2015–2016 гг. — Исполнитель);

– Федеральная целевая программа Министерства образования и науки РФ (проект № 114112170068 «Создание программно-аппаратного комплекса для управления стеганографической информацией для мультимедиа потоков в телевидении и интернет-вещании» на 2014–2016 гг. — Исполнитель);

– грант «Российского фонда фундаментальных исследований» (проект № 16-31-50010 «Разработка метода получения тождеств и свойств специальных полиномов на основе использования q-интегралов на кольце целых p-адических чисел и операции композиции производящих функций» на 2016 г. — Исполнитель);

– стипендия Президента РФ для молодых ученых и аспирантов, осуществляющих перспективные научные исследования и разработки по приоритетным направлениям модернизации российской экономики по направлению «Стратегические информационные технологии, включая вопросы создания суперкомпьютеров и разработки программного обеспечения» (научно-исследовательская работа по теме «Разработка алгоритмов и программного обеспечения на основе новых методов комбинаторной генерации» на 2018–2020 гг. — Руководитель);

– базовая часть государственного задания Министерства науки и высшего образования РФ (проект № 2.8172.2017/8.9 «Метод и модели определения уровня защищенности информационных систем» на 2017–2019 гг. — Исполнитель);

– грант «Российского научного фонда» (проект № 18-71-00059 «Разработка алгоритмов и программного обеспечения индексирования больших объемов данных на основе новых методов комбинаторной генерации» на 2018–2020 гг. — Руководитель);

– грант «Российского фонда фундаментальных исследований» (проект № 18-41-703006 «Исследование свойств полиномов на основе степеней производящих функций» на 2018–2019 гг. — Руководитель);

– грант «Российского фонда фундаментальных исследований» (проект № 20-31-70037 «Исследование коэффициентов степеней производящих функций многих переменных» на 2019–2021 гг. — Руководитель);

– грант «Фонда содействия инновациям» (проект № 77ГУЦЭС8-D3/56679 «Разработка системы адаптивного обучения с элементами искусственного интеллекта» на 2019–2021 гг. — Руководитель);

– государственное задание Министерства науки и высшего образования (проект № FEWM-2020-0046 «Фундаментальные основы и методология создания высокоэффективного энергопреобразования для систем космического и морского назначения на базе интеллектуальных силовых модулей сверхвысокой степени интеграции» на 2020–2022 гг. — Исполнитель);

– стипендия Президента РФ для молодых ученых и аспирантов, осуществляющих перспективные научные исследования и разработки по приоритетным направлениям модернизации российской экономики по направлению «Стратегические информационные технологии, включая вопросы создания суперкомпьютеров и разработки программного обеспечения» (научно-исследовательская работа по теме «Математическое, алгоритмическое и программное обеспечение для индек-

сации больших объемов данных, описываемых комбинаторными множествами» на 2021–2023 гг. — Руководитель).

Результаты диссертационной работы внедрены в деятельность «НИИ АЭМ ТУСУР» и в деятельность АО «Информационные спутниковые системы» имени академика М. Ф. Решетнёва» для решения практической задачи формирования выходных характеристик имитаторов энергопреобразующей аппаратуры, в деятельность ООО «ПлантаПлюс» для решения практической задачи улучшения информационной системы хранения и обработки экспериментальных данных, в деятельность ООО «Эль Контент» для решения практической задачи тестирования разработанного программного обеспечения систем обучения, в учебный процесс НИ ТПУ при обучении студентов по направлениям подготовки «Информатика и вычислительная техника» и «Информационные системы и технологии», в учебный процесс ФГБОУ ВО «ТУСУР» при обучении студентов по направлениям подготовки «Информатика и вычислительная техника» и «Управление в технических системах».

Личный вклад автора. Личный вклад автора настоящей диссертационной работы состоит в определении направлений исследований, в подготовке и проведении непосредственно научно-исследовательской работы, в самостоятельном формулировании выводов и научных положений.

Автором самостоятельно разработан комплексный метод формирования информационных объектов, основанный на k-й степени производящих функций, предложена модификация метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, разработаны алгоритмы комбинаторной генерации, построена база знаний производящих функций двух переменных, предложен подход к созданию программных систем компьютерной алгебры и систем тестирования на основе использования производящих функций. В постановке отдельных задач исследований, обсуждении результатов, проведении расчетов и программной реализации были привлечены сотрудники и студенты университета, что отражается в совместных работах. Разработка программного обеспечения проведена автором совместно с сотрудниками и студентами ТУСУР. Соавторы, принимавшие участие в отдельных направлениях исследований, указаны в списке основных публикаций по теме диссертации. Все результаты, составляющие научную основу диссертации и выносимые на защиту, получены автором лично.

Апробация работы. Основные результаты диссертационной работы докладывались и обсуждались на международных и всероссийских научных конференциях, в том числе на следующих конференциях:

– The International Conference of Numerical Analysis and Applied Mathematics (2012, 2015, 2016, 2018, 2021 гг., Греция);

– The International Conference «Commutative Ring Theory, Integer-Valued Polynomials and Polynomial Functions» (2012 г., Австрия, г. Грац, Грацский технический университет);

– The Palanga Conference in Combinatorics and Number Theory (2013 г., Литва, г. Паланга);

– Международная конференция «Дискретная математика, теория графов и их приложения» (2014 г., Беларусь, г. Минск, НАН Беларуси);

– The International Conference in Honour of Professor R.P. Agarwal (2014 г., Турция, г. Бурса, университет Улудаг);

– The International Conference on Lattice Path Combinatorics & Applications (2015 г., США, г. Помона, Калифорнийский государственный политехнический университет в г. Помона);

– The International Conference of The Jangjeon Mathematical Society (2015, 2016, 2019 гг.)

– The International Research Conference «Information Technologies in Science, Management, Social Sphere and Medicine» (2017 г., г. Томск);

– The Mediterranean International Conference of Pure & Applied Mathematics and Related Areas (2018–2021 гг.);

– Международная научно-техническая конференция студентов, аспирантов и молодых ученых «Научная сессия ТУСУР» (2012–2021 гг., г. Томск, ТУСУР);

– Международная научная конференция, посвященная памяти генерального конструктора ракетно-космических систем академика М.Ф. Решетнева (2013 г., г. Красноярск, СибГУ им. М.Ф. Решетнева);

– Международная научно-практическая конференция «Электронные средства и системы управления» (2013–2019 гг., г. Томск, ТУСУР);

– Международная научно-практическая конференция «Информационные процессы и технологии «Информатика – 2014» (2014 г., г. Севастополь);

– Всероссийская научно-практическая конференция студентов, аспирантов и молодых учёных «Технологии Microsoft в теории и практике программирования» (2015 г., г. Томск, ТПУ);

– Всероссийская научная конференция «Наука. Технологии. Инновации» (2015, 2016, 2018 гг., г. Новосибирск, НГТУ);

– Международная научно-практическая конференция молодых ученых «Прикладная математика и информатика: современные исследования в области естественных и технических наук» (2018–2019 гг., г. Тольятти, ТГУ);

– Международная научно-практическая конференция «Современные информационные технологии и ИТ-образование» (2018-2019 гг., г. Москва, МГУ имени М.В. Ломоносова);

– Международная конференция студентов, аспирантов и молодых ученых «Перспективы развития фундаментальных наук» (2021 г., г. Томск).

Публикации по теме диссертации. Основные результаты диссертационного исследования обобщены в 4 монографиях и опубликованы в 43 статьях в рецензируемых научных изданиях, рекомендуемых для опубликования основных научных результатов диссертаций на соискание ученых степеней кандидата и доктора наук. При этом 12 статей опубликовано в рецензируемых журналах из перечня ВАК, 31 статья опубликована в зарубежных научных изданиях, в том числе 27 в индексируемых Web of Science и/или Scopus (из них 11 статей в журналах, входящих в первый и второй квартили Web of Science и/или Scopus). Результаты исследований представлены также в виде 60 публикаций в тезисах и материалах научных конференций. Разработанные технические решения и методы защищены четырьмя свидетельствами о регистрации программ для ЭВМ. В опубликованных работах материалы диссертации изложены полно.

Объем и структура работы. Диссертация состоит из введения, шести глав основной части, заключения, списка литературы и двух приложений. Полный объем диссертации составляет 319 страниц, включая 43 рисунка и 11 таблиц. Список литературы содержит 299 наименований.

Глава 1. Анализ современного состояния исследований в области теории производящих функций

В данной главе вводятся основные понятия, применяемые в ходе диссертационного исследования. Приводится аналитический обзор современного состояния исследований в области теории производящих функций и методов на основе их применения, в том числе для задач описания информационных объектов. Рассматривается классификация типов производящих функций, а также способы оперирования коэффициентами производящих функций и подходы к их вычислению. Кроме того, обсуждаются области применения математического аппарата производящих функций и актуальные направления исследований.

1.1 Понятие производящей функции

Ключевым понятием, которое исследуется в диссертационной работе, является математический аппарат производящих функций, имеющий широкое приложение к перечислительным задачам. Производящие функции являются одним из популярных и мощных инструментов для решения задач теоретической информатики, дискретного анализа, статистики и других областей науки. Математический аппарат производящих функций активно применяется при решении задач из области комбинаторики, так как производящие функции позволяют получить компактное представление дискретных структур, а также предоставляют широкий набор функциональных возможностей при работе с такими структурами. В математическом анализе производящие функции используются как эффективный инструмент представления и исследования полиномов.

Согласно Р.П. Стенли [1], самый полезный метод численного представления некоторой функции $f(n)$, считающей число элементов в конечном множестве, — это метод производящих функций. Метод производящих функций позволяет не только посчитать количество элементов в конечном множестве, но и получить различные комбинаторные и практические приложения за счет выполнения операций над производящими функциями. Также стоит отметить фундаментальную работу Р.Л. Грэхема, Д.Э. Кнута и О. Паташника [2] в области математических основ

информатики, в которой в качестве ключевой идеи выделяется применение именно понятия производящих функций.

Ниже представлено определение производящей функции, приведенное в работах С.А. Ландо [3]:

Определение 1. *Обыкновенной производящей функцией произвольной (бесконечной) последовательности чисел* $f(0), f(1), f(2), \ldots$ *называется выражение вида*

$$f(0) + f(1)x + f(2)x^2 + \ldots = \sum_{n \geqslant 0} f(n)\, x^n.$$

По аналогии с обычными функциями, для краткого обозначения производящей функции часто используется запись в виде одной заглавной буквы с указанием в скобках ее аргумента, например:

$$F(x) = \sum_{n \geqslant 0} f(n)\, x^n.$$

Кроме того, последовательность чисел производящей функции представляет собой разложение в ряд по степеням некоторой формальной переменной x. В таком случае переменная x носит лишь формальный характер и с ней не связывают каких-либо конкретных значений, поэтому сходимость такого ряда не имеет значения и не исследуется. Таким образом, вместо понятия «производящая функция» можно использовать понятие «формальный степенной ряд». Тем не менее, в некоторых случаях для заданного формального степенного ряда можно взаимно однозначно поставить в соответствие аналитическую функцию. Например, это могут быть следующие разложения функций:

$$F(x) = 1 + x + x^2 + \ldots + x^n + \ldots = \frac{1}{1-x}$$

или

$$F(x) = 1 + x + \frac{1}{2!}x^2 + \ldots + \frac{1}{n!}x^n + \ldots = e^x.$$

Первые упоминания о применении метода производящих функций можно встретить в работах А. де Муавра [4] при решении линейных рекуррентных уравнений (1730 г.). Д. Стирлинг расширил результаты работ де Муавра путем исследования методов решения рекуррентных уравнений более сложного типа. Дальнейшее применение метода производящих функций можно встретить в работах Л. Эйлера при решении задач, связанных с исследованием разбиений.

В то время развитие метода производящих функций во многом шло за счет задач о разбиениях, исследования в данной области хорошо обобщены в работах Г. Эндрюса [5]. Также большое внимание было уделено развитию теории производящих функций со стороны Д. Риордана [6], согласно которому производящие функции являются ключевым орудием современной комбинаторики и могут быть средством унификации при исследовании комбинаторных проблем.

В отечественной литературе значительное внимание теория производящих функций получила в работах Н.Я. Виленкина [7], К.А. Рыбникова [8], С.А. Ландо [3], В.Н. Сачкова [9], Г.П. Егорычева [10], О.В. Кузьмина [11] и других ученых.

1.2 Типы производящих функций

Проведение классификации производящих функций может быть реализовано в зависимости от количества параметров, от характера описываемого объекта (например, полиномы), от количества формальных переменных и других дополнительных условий.

Помимо обыкновенных производящих функций, как отдельный тип выделяют экспоненциальные производящие функции [12].

Определение 2. *Экспоненциальной производящей функцией произвольной (бесконечной) последовательности чисел* $f(0), f(1), f(2), \ldots$ *называется выражение вида*

$$\frac{f(0)}{0!} + \frac{f(1)}{1!}x + \frac{f(2)}{2!}x^2 + \ldots = \sum_{n \geqslant 0} \frac{f(n)}{n!}\, x^n.$$

Отличием экспоненциальных производящих функции от обыкновенных производящих функций является изменение правил работы операций их произведения, композиции, дифференцирования и интегрирования. При этом следует отметить, что любая экспоненциальная производящая функция может быть представлена в виде обыкновенной производящей функции и исследована соответствующими методами. В математической статистике данный тип производящих функций фигурирует как производящая функция моментов [6].

Также существуют обобщения экспоненциальных производящих функций [1], например, таковыми являются Эйлеровы производящие функции или r-экспоненциальные производящие функции.

Определение 3. *Эйлеровой производящей функцией произвольной (бесконечной) последовательности чисел* $f(0), f(1), f(2), \ldots$ *называется выражение вида*

$$\sum_{n \geqslant 0} \frac{f(n)}{(n)!}\, x^n,$$

где $(n)! = (1+q)(1+q+q^2)\cdots(1+q+\ldots+q^{n-1})$.

Эйлерова производящая функция сводится к экспоненциальной при условии $q = 1$.

Определение 4. *r-производящей функцией произвольной (бесконечной) последовательности чисел* $f(0), f(1), f(2), \ldots$ *называется выражение вида*

$$\sum_{n \geqslant 0} \frac{f(n)}{(n!)^r}\, x^n.$$

Также r-производящая функция сводится к экспоненциальной при условии $r = 1$.

Приведем пример множества производящих функций для одного известного комбинаторного объекта, называемого треугольником Паскаля [2]. Треугольник Паскаля может быть представлен следующими производящими функциями:

– двумерная производящая функция $\frac{1}{1-y-xy}$ (основана на понятии биномиальных коэффициентов);

– двумерная производящая функция $\frac{1}{1-y-x}$ (основана на нумерации элементов треугольника числом отрезков каждого типа на путях, ведущих в соответствующую точку треугольника);

– экспоненциальная производящая функция e^{x+y};

– степень обыкновенной производящей функции $\frac{x}{1-x}$;

– массив Риордана $\left(\frac{1}{1-x}, \frac{x}{1-x}\right)$.

Другим типом производящих функций являются производящие функции Дирихле, определяемые следующим образом [12]:

Определение 5. *Производящей функцией Дирихле произвольной (бесконечной) последовательности чисел* $f(0), f(1), f(2), \ldots$ *называется выражение вида*

$$\frac{f(1)}{1^s} + \frac{f(2)}{2^s} + \frac{f(3)}{3^s} + \ldots = \sum_{n>0} \frac{f(n)}{n^s}.$$

Например, производящая функция Дирихле для последовательности единиц есть дзета-функция Римана [2]:

$$\sum_{n>0} \frac{1}{n^s} = \zeta(s).$$

Производящие функции Дирихле полезны, когда генерируемая последовательность является мультипликативной функцией. Значительный вклад в развитие аналитических свойств производящих функций Дирихле внес С.М. Воронин [13].

В данной диссертации как основной тип производящих функций рассматриваются обыкновенные производящие функции, тем не менее предлагаемые в диссертационной работе подходы могут быть применены и для других типов производящих функций.

1.3 Производящие функции полиномов

Производящие функции широко применяются для определения разного рода полиномов. Полиномы играют важную роль при решении различных задач из областей теории операторов, аналитических функций, интерполяции, приближений и численного анализа, а также в электростатике, статистической квантовой механике, теории чисел, комбинаторике, теории стохастических процессов, сортировке и сжатии данных, и так далее.

Основы общей теории ортогональных полиномов были заложены П.Л. Чебышевым. Классические труды А.А. Маркова, Т.И. Стилтьеса, С.Н. Бернштейна, Г. Сеге [14] и других ученых значительно способствовали дальнейшему развитию общей теории полиномов и созданию принципиально новых методов их исследования.

Обширное применение полиномов в различных областях математики отразилось на многочисленных исследованиях. На данный момент существует достаточно много больших обзорных работ по теории специальных полиномов, например, серия книг проекта «Bateman Project» под руководством английского математика А. Эрдели [15–17], книги таких авторов, как Р.П. Боас и Р.К. Бак [18], Я.Л. Геронимус [19], Н.Н. Лебедев [20], П.К. Суетин [21], В.В. Прасолов [22] и других ученых.

За последние десятилетия сформировался значительный прогресс с точки зрения полученных теоретических и практических результатов в области

исследования производящих функций специальных полиномов. В частности, большое значение для исследования производящих функций полиномов имеют работы Э.Б. Макбрайд [23] и Х.М. Шривастава [24–27]. Кроме того, развились целые научные школы по данному направлению (например, корейская школа под руководством Т. Кима [28–31], турецкая школа Й. Шимшека [32–37] и другие). Таким образом, с учетом быстрорастущего количества публикаций по данной тематике, происходит процесс формирования отдельного научного направления, связанного с исследованием производящих функций полиномов.

Еще одним трендом, связанным с исследованием производящих функций, можно выделить направление по исследованию явных представлений полиномов. Так, например, Ф. Ки исследовал обобщенные числа Моцкина [38] и центральные числа Деланноя [39], К.Н. Бояджиев [40] рассматривал явные представления полиномов на основе тригонометрических функций, М. Ченкчи [41] рассматривал явные представления для обобщенных потенциальных полиномов, Х.М. Шривастава [42; 43] рассматривал явные представления разных полиномов, включая полиномы Бернулли и Норлунда. Еще одним важным направлением в теории полиномов является получение различных их свойств и тождеств, в данном случае можно отметить работы [44–46].

Производящие функции полиномов характеризуются набором параметров и зачастую одной формальной переменной, по которой собственно и идет разложение в ряд.

Полиномом будем называть конечную линейную комбинацию мономов x^k с вещественными коэффициентами, точнее:

Определение 6. *Полиномом степени n называется функция вида*

$$Y_n(x) = p_{(n,n)}x^n + p_{(n,n-1)}x^{n-1} + \ldots + p_{(n,1)}x + p_{(n,0)}, \qquad p_{(n,n)} \neq 0.$$

Двойная индексация коэффициентов $p_{(n,m)}$ полинома $Y_n(x)$ естественна при использовании метода производящих функций. Числа $p_{(n,m)}$ — это коэффициенты двойного степенного ряда

$$\sum_{n\geqslant 0}\sum_{m\geqslant 0} p_{(n,m)}\, t^n x^m.$$

Набор коэффициентов $p_{(n,m)}$ однозначно определяет полином. Если все коэффициенты двух полиномов равны, то данные полиномы считаются одинаковыми.

Определение 7. *Пусть $F(x,t)$ — производящий степенной ряд по (формальной) переменной t*

$$F(x,t) = \sum_{n\geqslant 0} f_n(x)t^n,$$

где каждый коэффициент $f_n(x)$ является многочленом от x. Тогда говорят, что $F(x,t)$ есть производящая функция для последовательности полиномов $f_n(x)$ (производящая функция полиномов).

Для производящих функций полиномов различают следующие виды производящих функций [24]:

1. Линейные производящие функции полиномов:

Определение 8. *Функция от двух переменных $F(x,t)$, имеющая такое разложение в ряд по степеням формальной переменной, что*

$$F(x,t) = \sum_{n=0}^{\infty} c_n f_n(x)t^n,$$

где последовательность $\langle c_n\rangle_{n=0}^{\infty}$ может содержать параметры функции $f_n(x)$ и $f_n(x)$ не зависит от t, называется линейной производящей функцией;

2. Билинейные производящие функции полиномов:

Определение 9. *Функция от трех переменных $F(x,y,t)$, имеющая такое разложение в ряд по степеням формальной переменной, что*

$$F(x,y,t) = \sum_{n=0}^{\infty} \gamma_n f_n(x) f_n(y)t^n,$$

где последовательность $\langle\gamma_n\rangle_{n=0}^{\infty}$ не зависит от x и t, называется билинейной производящей функцией для последовательности полиномов $\langle f_n(x)\rangle_{n=0}^{\infty}$;

3. Двумерные производящие функции полиномов:

Д. Рейнвилл [47] и Э.Б. Макбрайд [23] ввели следующее определение:

Определение 10. *Функция от трех переменных $F(x,y,t)$, имеющая такое разложение в ряд по степеням формальной переменной, что*

$$H(x,y,t) = \sum_{n=0}^{\infty} h_n f_n(x) g_n(y)t^n,$$

где последовательность $\langle h_n\rangle_{n=0}^{\infty}$ *не зависит от* x, y *и* t, *называется двумерной производящей функцией для последовательности полиномов* $\langle f_n(x)\rangle_{n=0}^{\infty}$ *или для последовательности полиномов* $\langle g_n(x)\rangle_{n=0}^{\infty}$.

Данное определение было расширено Х.Л. Маноча и Х.М. Шривастава [24]:

Определение 11. *Функция от трех переменных* $F(x,y,t)$, *имеющая такое разложение в ряд по степеням формальной переменной, что*

$$\Psi(x,y,t) = \sum_{n=0}^{\infty} \psi_n f_{\alpha(n)}(x) g_{\beta(n)}(y) t^n,$$

где последовательность $\langle \psi_n\rangle_{n=0}^{\infty}$ *не зависит от* x, y *и* t, *функции* $\langle f_{\alpha(n)}(x)\rangle_{n=0}^{\infty}$ *и* $\langle g_{\beta(n)}(x)\rangle_{n=0}^{\infty}$ *разные, а* $\alpha(n)$ *и* $\beta(n)$ *не обязательно равны, называется двумерной производящей функцией для последовательности полиномов* $\langle f_{\alpha(n)}(x)\rangle_{n=0}^{\infty}$ *или для последовательности полиномов* $\langle g_{\beta(n)}(x)\rangle_{n=0}^{\infty}$;

4. Многомерные производящие функции полиномов:

Определение 12. *Функция от* $r+1$ *переменных* $G(x_1,\ldots,x_r,t)$, *имеющая такое разложение в ряд по степеням формальной переменной, что*

$$G(x_1,\ldots,x_r,t) = \sum_{n=0}^{\infty} c_n g_n(x_1,\ldots,x_r) t^n,$$

где последовательность $\langle c_n\rangle_{n=0}^{\infty}$ *не зависит от* $x_1,\ldots,x_r$ *и* t, *называется многомерной производящей функцией для последовательности полиномов* $\langle g_n(x_1,\ldots,x_r)\rangle_{n=0}^{\infty}$.

Для получения производящих функций на основе информации о их коэффициентах существуют различные методики, например:

1. Техника перегруппировки рядов (или манипуляция рядами) [24]. Данная техника основана на применении определенного набора лемм и их вариаций, однако перегруппировка рядов требует большого количества манипуляций, что вызывает большие затруднения ее применения;

2. Техника декомпозиции [48; 49]. Идея данной техники заключается в том, что заданный ряд разбивается на две части с четными и нечетными элементами, после чего применяются тождества на основе обобщенных гипергеометрических рядов для получения искомой производящей функции;

3. Операторные методы [15–17; 24]. Данные методы основаны на одинарных, двукратных и многократных интегральных преобразованиях и на некоторых операторах, использующих частные производные;

4. Техника дробного дифференцирования [24; 50]. Данная техника основана на использовании дифференциальных уравнений дробного порядка, в которых неизвестная функция содержится под знаком производной дробного порядка;

5. Теневое исчисление. Теневое исчисление изучает связи между различными последовательностями полиномов и степенными рядами. Изначально теневое исчисление представлялось набором правил для манипулирования индексами в последовательностях [51]. С. Роман [52] и Д.-К. Рота [53; 54] ввели современное понятие теневого исчисления, заключающееся в использовании линейных функционалов на пространствах полиномов. Основное внимание уделяется изучению последовательностей Аппеля и Шеффера, включая последовательности полиномов биномиального типа.

Одной из ключевых задач при работе с производящими функциями полиномов выделяют вычисление соответствующих им коэффициентов. В последнее время идет тенденция разработки методов получения явных формул для полиномов (например, в работах Х.М. Шривастава [42; 43], К.Н. Бояджиева [40], М. Ченкчи [41] и многих других). Однако стоит отметить, что в этих работах рассматриваются лишь отдельные частные случаи в области получения явных формул для полиномов. Поэтому исследования новых свойств производящих функций, позволяющих найти явные представления для полиномов как информационных объектов и обобщить ранее предложенные методы, являются актуальными.

1.4 Операции над производящими функциями

Операции над производящими функциями в основном заключаются в вычислении явных выражений для соответствующих им функций коэффициентов. Описание таких операций представлено во многих обзорных работах по производящим функциям, например, в книгах [6; 12]. Прежде чем приступить к описанию операций над производящими функциями необходимо определиться с нулевой степенью производящей функции, поскольку степени производящих функций имеют ключевое значение для операции композиции производящих функций.

В диссертационной работе будем считать верным следующее утверждение:

Аксиома 1. *Для произвольной производящей функции* $A(x) = \sum\limits_{n\geqslant 0} a(n)\,x^n$ *ее нулевая степень равна единице:* $(A(x))^0 = 1$.

Для производящих функций обычно определяют следующий перечень доступных операций [2; 4]. Пусть заданы обыкновенные производящие функции $G(x) = \sum\limits_{n\geqslant 0} g(n)\,x^n$, $F(x) = \sum\limits_{n\geqslant 0} f(n)\,x^n$ и $A(x) = \sum\limits_{n\geqslant 1} a(n)\,x^n$, тогда для них доступны такие операции, как:

1. Сдвиг
$$x^m G(x) = \sum_{n\geqslant m} g(n-m)\,x^n;$$
2. Замена переменной
$$G(cx) = \sum_{n\geqslant 0} g(n)\,c^n x^n;$$
3. Сложение
$$G(x) + F(x) = \sum_{n\geqslant 0} (g(n) + f(n))\,x^n;$$
4. Умножение
$$G(x)\cdot F(x) = \sum_{n\geqslant 0}\sum_{k=0}^{n} g(k) f(n-k)\,x^n;$$
5. Дифференцирование
$$(G(x))' = \sum_{n\geqslant 0} (n+1) g(n+1)\,x^n;$$
6. Интегрирование
$$\int_0^x G(t)\,dt = \sum_{n\geqslant 1} \frac{1}{n} g(n-1)\,x^n;$$
7. Композиция
$$G(A(x)) = \sum_{n\geqslant 0} g(n)\,(A(x))^n.$$

Формальное выражение для коэффициентов композиции производящих функций берет свое начало с формулы Ф. Фаа ди Бруно [55; 56]. Дальнейшее развитие работ, связанное с данной формулой, рассмотрено в работе Л. Комте [57]. Также отметим распространенный вариант обозначения коэффициентов производящих функций [4]: запись $[x^n]G(x)$ показывает коэффициент при x^n в выражении для производящей функции $G(x)$.

Отдельно стоит рассмотреть процесс получения коэффициентов обратных производящих функций (относительно произведения и операции композиции производящих функций). Для этого рассмотрим следующие понятия, приведенные в [3; 12].

Определение 13. *Взаимной производящей функцией $G(x)$ для производящей функции $F(x) = \sum\limits_{n\geqslant 0} f(n)\,x^n$ является формальный степенной ряд, удовлетворяющий условию*

$$G(x) \cdot F(x) = 1. \tag{1.1}$$

Определение 14. *Обратной производящей функцией $\overline{F(x)}$ для производящей функции $F(x) = \sum\limits_{n>0} f(n)\,x^n$, где $f(1) \neq 0$, является формальный степенной ряд, удовлетворяющий условию*

$$F(\overline{F(x)}) = x. \tag{1.2}$$

В литературе также встречаются обозначения обратной производящей функции $F^{[-1]}(x)$ или $RevF$.

Исследованием и применением композиции производящих функций, в том числе экспоненциальных производящих функций, включая их комбинаторный смысл, занимался Р. Стенли [58]. Он показал, что обратная производящая функция $\overline{F(x)}$ существует тогда и только тогда, когда $f(1) \neq 0$. Причем такой формальный степенной ряд является единственным. В большинстве случаев простой явной формулы для $\overline{F(x)}$ не существует. Тем не менее можно рассмотреть задачу поиска коэффициентов соответствующего степенного ряда. Классическим методом определения коэффициентов обратных производящих функций является вычисление общего выражения коэффициентов, исходя из поведения нескольких первых коэффициентов, полученных явно. В литературе не представлено общих простых подходов для вычисления соответствующих коэффициентов. Известен вариант формулы обращения Лагранжа [58] для решения функционального уравнения

$A(x) = G(xA(x))$. Однако для общего случая $A(x) = G(xA(x)^m)$, где производящая функция $A(x)$ не известна, а m — любое целое число, общего решения в виде явных формул коэффициентов производящих функций не существует.

Лемма 1 (Формула обращения Лагранжа)**.** *Пусть* $H(x) = \sum_{n\geqslant 0} h(n)\, x^n$*, где* $h(0) \neq 0$ *и* $A(x)$ *определена следующим образом:*

$$A(x) = xH(A(x)).$$

Тогда

$$n[x^n]A(x)^k = k[x^{n-k}]H(x)^n.$$

Существует множество обобщений и применений данной формулы обращения Лагранжа. Исследованию формул обращения Лагранжа, в том числе и для случая производящих функций многих переменных, посвящены обзорные работы И.М. Гессель [59; 60], также можно отметить работы Г. Лабелле [61–63], К. Кратценталера [64], Д. Мерлини, Р. Спругноли и М. Верри [65]. Большое внимание уделялось и исследованию q-аналогов формулы обращения Лагранжа [66].

Также стоит отметить операцию «дробной композиции». Так, согласно Р. Стенли [58], не решенной задачей является поиск комбинаторного смысла, например, для дробной композиции $(e^x - 1)^{\langle 1/2\rangle} = \sum_{n\geqslant 1} a(n)\frac{x^n}{n!}$.

1.5 Степени производящих функций

Использование степеней производящих функций является органичным обобщением метода производящих функций. Более того, коэффициенты степени производящей функции играют ключевую роль в операции композиции производящих функций, что было отмечено в предыдущем параграфе.

Получение выражений коэффициентов степеней производящих функций впервые осуществил Л. Эйлер, рассматривая конкретный вид коэффициентов производящей функции $(1 + x + x^2 + \ldots + x^n + \ldots)^k$, он получил

$$(1 + x + x^2 + \ldots + x^n + \ldots)^k = \left(\frac{1}{1-x}\right)^k = \sum_{n\geqslant 0} \binom{n+k-1}{k-1} x^n.$$

Известная формула бинома Ньютона

$$(1+x)^k = \sum_{n=0}^{k} \binom{k}{n} x^n$$

также является степенью производящей функции $(1+x)$.

Другие имеющие важное значение коэффициенты степеней производящих функций отражены в работе Л. Комте [57]. Например, степени производящих функций определяют числа Стирлинга первого рода

$$\ln(1+x)^k = \sum_{n>0} \frac{k!}{n!} \begin{bmatrix} n \\ k \end{bmatrix} x^n$$

и числа Стирлинга второго рода

$$(e^x - 1)^k = \sum_{n>0} \frac{k!}{n!} \left\{ {n \atop k} \right\} x^n.$$

Числа Стирлинга первого рода $\begin{bmatrix} n \\ k \end{bmatrix}$ определяют общее количество перестановок n элементов с k циклами. Числа Стирлинга второго рода $\left\{ {n \atop k} \right\}$ определяют общее количество неупорядоченных разбиений n-элементного множества на k непустых подмножеств [2; 57].

Кроме того, что коэффициенты степеней производящих функций играют ключевую роль при выполнении операции композиции производящих функций, они также важны при выполнении операции обращения производящих функций. Также коэффициенты степеней производящих функций находят свое применение при вычислении полиномов Белла второго рода $B(n,k,x)$, так как

$$(f(x+z) - f(z))^k = \sum_{n>k} \frac{k!}{n!} B(n,k,x)\, z^n.$$

Большое количество научных работ, посвященных комбинаторным проблемам и производящим функциям, базируются на применении коэффициентов степеней производящих функций (например, работы [9; 12; 57]). Однако как самостоятельный объект исследования коэффициенты степеней производящих не рассматриваются. С этой точки зрения можно отметить работы М. Дрмота [67], где подробно исследуются коэффициенты степеней производящих функций, однако здесь главной целью ставится поиск асимптотического распределения коэффициентов степеней производящих функций, а не решение задачи их получения в явном виде. Поэтому далее коэффициенты степеней производящих функций рассматриваются как основной объект исследования.

Для любой производящей функции $A(x) = \sum\limits_{n\geqslant 0} a(n)\, x^n$ ее степень $A(x)^k$, где $k > 0$, всегда можно представить в виде

$$A(x)^k = \sum_{n\geqslant 0} A(n,k)\, x^n.$$

Откуда будет справедливо следующее рекуррентное соотношение [68]:

$$A(n,k) = \begin{cases} a(n), & k = 1; \\ \sum\limits_{i=0}^{n} a(i)A(n-i,k-1), & k > 1. \end{cases}$$

Таким образом, k-ю степень производящей функции $A(x)$ можно рассматривать как производящую функцию, коэффициенты которой имеют два параметра n и k. Кроме того, по приведенному выше утверждению о нулевой степени производящей функции будем иметь $A(0,0) = 1$ и $A(n,0) = 0$ при $n > 0$.

Можно выделить следующие объекты, основанные на степенях производящих функций:

1. Потенциальные полиномы, введенные Л. Комте [57]: потенциальный полином $P_n^{(k)}$ представляет собой коэффициент k-й комплексной степени экспоненциальной производящей функции вида

$$\left(1 + \sum_{n>0} g_n \frac{x^n}{n!}\right)^k = 1 + \sum_{n>0} P_n^{(k)} \frac{x^n}{n!}.$$

Известна связь таких полиномов с полиномами Белла, но правила для операций сложения, умножения и обращения не представлены;

2. Массивы Риордана, введенные Л.У. Шапиро [69]: массив Риордана представляет собой пару производящих функций $D = (F(x),G(x))$, где $F(x) = \sum\limits_{n\geqslant 0} f(n)\, x^n$ и $G(x) = \sum\limits_{n>0} g(n)\, x^n$, что образует числовой треугольник $D = (d_{n,k})_{n,k\geqslant 0}$, где $d_{n,k} = [x^n]F(x) \cdot G(x)^k$. При этом ассоциативная подгруппа группы Массивов Риордана $D = (1,G(x))$ представляет собой функцию коэффициентов степени производящей функции $d_{n,k} = [x^n]G(x)^k$, то есть

$$G(x)^k = \sum_{n\geqslant k} d_{n,k} x^n.$$

Более подробная информация о свойствах и применимости массивов Риордана может быть найдена в работах [69–72];

3. Степенные матрицы, введенные Д.Э. Кнутом [73]: степенная матрица степенного ряда $V(x) = V_1x + V_2x^2 + \dots$ представляет собой бесконечный массив коэффициентов $v_{n,k} = \frac{n!}{k!}[x^n]V(x)^k$. Тогда k-я степень $V(x)$ может быть представлена в виде

$$V(x)^k = \sum_{n \geqslant k} \frac{k!}{n!} v_{n,k}\, x^n.$$

При этом правила вычисления коэффициентов $v_{n,k}$ были проработаны только для случая композиции производящих функций;

4. Обобщенные пирамиды Паскаля, введенные О.В. Кузьминым [11]: суть данного подхода заключается в построении арифметических треугольников комбинаторного происхождения, родственных треугольнику Паскаля, и их применение на практике.

1.6 Многомерные производящие функции

Приведенные выше исследования касаются случая производящей функции одной формальной переменной. Однако существует большое количество задач, связанных с производящими функциями многих переменных, в том числе для которых не известны явные представления, но известны производящие функции. Например, задача нахождения вероятности наличия конкретного ребра в идеальном графе для ацтекского бриллианта [74], задача подсчета всех гистограмм [75–77], задача нахождения явных формул для числовых треугольников Дика и Моцкина с кратностями [78] и другие.

Дадим определение многомерной производящей функции:

Определение 15. *Многомерной производящей функцией будем называть следующий формальный степенной ряд:*

$$F(x,y,\dots,z) = \sum_{n \geqslant 0} \sum_{m \geqslant 0} \dots \sum_{l \geqslant 0} f(n,m,\dots,k) x^n y^m \cdots z^l.$$

Далее дадим понятие порядка формального степенного ряда [79]:

Определение 16. *Порядком формального степенного ряда* $F(x,y,\ldots,z)$ *будем называть выражение* $ord(F)$*, которое определяется следующим образом:*

$$ord(F) = \begin{cases} \min\{r = n + m + \ldots + l : f(n,m,\ldots,l) \neq 0\}, & F(x,y,\ldots,z) \neq 0; \\ +\infty, & F(x,y,\ldots,z) = 0. \end{cases}$$

Тогда для двух формальных степенных рядов $F(x,y,\ldots,z)$ и $G(x,y,\ldots,z)$, где $ord(F) > 0$ и $ord(G) > 0$, порядок $F(x,y,\ldots,z) \cdot G(x,y,\ldots,z)$ будет равен

$$ord(F \cdot G) = ord(F) + ord(G).$$

Также как и для одномерного случая, одной из ключевых задач при работе с многомерными производящими функциями выделяют вычисление соответствующих им коэффициентов. Попытками систематизации процесса нахождения коэффициентов производящих функций многих переменных занимаются Р. Пемантл, М. Уилсон и другие [80–82] из проекта «Asymptotics of multivariate sequences» [83]. В данном случае, в отличие от производящих функций одной формальной переменной исследование свойств производящих функций многих переменных сводится непосредственно к решению системы нетривиальных функциональных уравнений для заданной производящей функции многих переменных. Важно отметить, что участники данного проекта исследуют только асимптотические методы, то есть для производящих функций многих переменных исследования проводились только с точки зрения приближенного вычисления коэффициентов разложения производящих функций. Поиску асимптотических оценок также посвящены работы Э. Бендера и Б. Ричмонда [84]. А. Райчев и М. Уилсон [85] исследовали асимптотики и добились более точной численной аппроксимации, что показали на путях решетки, квантовых случайных блужданиях и неперекрывающихся моделях. Кроме того, существуют подходы для частных случаев производящих функций многих переменных, например, О.А. Шишкина [86] рассматривала обобщения чисел и полиномов Бернулли для случая нескольких переменных.

Формирование общего подхода к исследованию степеней производящих функций многих переменных для получения их явных выражений, в том числе для полиномов, заданных производящими функциями многих переменных, имеет важное значение. Поэтому в данной диссертации уделяется особое внимание решению проблемы разложения в явном виде.

В качестве исследований наибольшее распространение получили частные случаи двумерного характера. Например, в работе [9] описывается следующая схема получения коэффициентов двумерной производящей функции для решения перечислительных задач в комбинаторике. Для отыскания соответствующих коэффициентов двумерной производящей функции (в работе она также называется двойной) используется рекуррентное соотношение, из которого выводится соответствующее дифференциальное уравнение, решение которого позволяет получить сначала саму двумерную производящую функцию $F(x,y)$, а последующее разложение позволяет уже получить ее коэффициенты.

1.7 Применение производящих функций

Производящие функции, как инструмент описания и изучения информационных объектов, имеют разнообразное применение как в математике, так и в теоретической информатике. Основное применение производящих функций связано с перечислительными задачами (подсчет разного рода информационных объектов, то есть определение количества объектов в некотором классе, что служит хорошей основой для решения задач хранения и генерации).

В общем виде метод производящих функций находит свое применение при решении следующих задач [12]:

– поиск явной формулы для членов числовых последовательностей;

– поиск рекуррентных формул;

– поиск средних и других статистических характеристик заданных последовательностей;

– поиск асимптотических формул;

– доказательство разного рода утверждений и подходов.

В качестве основных информационных объектов, для которых применяется аппарат производящих функций, можно выделить графы, деревья, пути, скобочные структуры и многое другое [12].

Поскольку числовая последовательность может описывать различные комбинаторные объекты, то приложение метода производящих функций достаточно обширно. Начало применения производящих функций для перечисления деревьев, расстановок скобок, баллотировочных последовательностей, решеточных путей

и рассечений многоугольников, а также для выявления связей между этими задачами можно отнести еще к Л. Эйлеру [58].

Производящие функции являются мощным инструментом для использования в системах исчисления, они применяются для разработки алгоритмов перевода между системами счисления (например, перевод числа из фиббоначиевой системы счисления в позиционную и обратно, а также соответствующие алгоритмы сложения и умножения в этой системе [3]).

Еще одной важной задачей, которую решают с использованием теории производящих функций, является задача перечисления помеченных и непомеченных деревьев. Поскольку деревья имеют рекурсивную структуру, то для этого часто используют метод композиции производящих функций [58]. В таком случае производящая функция описывает количество вариантов деревьев в зависимости от их размерности. В работе [9] рассматриваются производящие функции для корневых и свободных деревьев, в которых описывается число лесов с n вершинами, состоящих из k корневых деревьев. Более того, в зависимости от задачи применяют разные типы производящих функций [3]. Например, экспоненциальные производящие функции хорошо описывают помеченные объекты, тогда как обыкновенные производящие функции лучше представляют непомеченные. Такое разделение способствует к применению естественных формул перечисления производных объектов.

Также важной задачей является поиск производящих функций для разного рода формальных языков, например, для языка Дика, языка Моцкина и других языков. Кроме того, производящие функции находят свое применение для формальных грамматик с однозначным выводом [3].

Теория производящих функций используется для изучения дискретных случайных величин [87]. Согласно [8], если пространство элементарных событий состоит из конечного или счетного множества точек, то это пространство называют дискретным. В теории вероятностей производящие функции применяются достаточно давно [2; 8; 9], например, они используются для описания функций распределения вероятностей и моментов случайных величин. В работах Ф. Дюма и Л. Тхимонье [88] рассматривалась задача определения вероятности образования палиндрома за конечное время независимыми розыгрышами с помощью производящих функций многих переменных. Также производящие функции хорошо применимы к поиску средних значений, стандартных отклонений и других моментов распределений с минимальными усилиями [12].

Многие дискретные структуры имеют рекурсивную природу, которая позволяет получить для них функциональные или дифференциальные уравнения на соответствующую экспоненциальную производящую функцию. Другим важным применением производящих функций является решение задач, связанных с рекуррентными выражениями [2]. Например, возможно оперирование последовательностями чисел Фибоначчи (или им подобным) через использование их производящих функций.

Вид и способ применения производящих функций зависит от конкретных условий рассматриваемой задачи, поскольку не существует универсальных методов построения самих производящих функций. Безусловно стоит отметить подходы Д. Пойа, который рассматривал процесс построения неэквивалентных комбинаторных объектов общего вида [8]. В качестве такого применения можно рассмотреть перечисления изомеров органических молекул заданной структуры. Пусть имеется молекула, представленная на рисунке 1.1, где C — атом углерода, а вместо символа $\times$ могут находиться CH_3 (метил), C_2H_5 (этил), H (водород) и Cl (хлор). Тогда математической моделью молекул будет являться тетраэдр с атомом углерода в центре, а перечисление молекул будет ставиться как задача о числе классов эквивалентности.

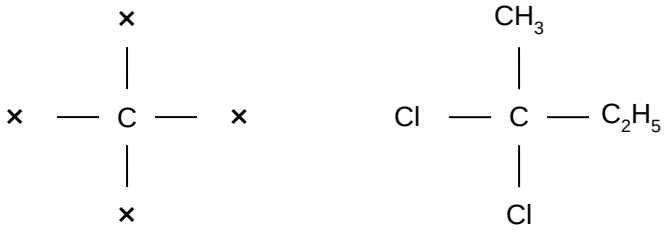


Рисунок 1.1 — Модель молекулы и ее пример (дихлорбутан)

1.8 Выводы по главе

В данной главе введены основные понятия диссертационного исследования. Ключевым исследуемым понятием является математический аппарат производящих функций, который имеет важное приложение к перечислительным задачам со стороны подсчета разного рода объектов, то есть для определения количества

объектов в некотором классе (в том числе для задач описания информационных объектов). В качестве основных информационных объектов, для которых применяется аппарат производящих функций, можно выделить графы, деревья, пути, скобочные структуры, полиномы и многие другие. Производящие функции являются одним из популярных и мощных инструментов для решения задач теоретической информатики, дискретного анализа, статистики и других областей науки. Математический аппарат производящих функций также широко применяется при решении задач из области комбинаторики, так как производящие функции позволяют получить компактное представление дискретных структур и предоставляют широкий набор функциональных возможностей при работе с такими структурами.

В данной главе приведена классификация производящих функций в зависимости от характера описываемого объекта (например, полиномы), от количества формальных переменных и других условий. Проведен аналитический обзор современного состояния исследований в области теории производящих функций и методов на основе их применения, в том числе для задач описания информационных объектов. Описаны основные операции над производящими функциями одной переменной: умножение, сложение, композиция и обращение. Отдельно рассмотрено использование степеней производящих функций как органичное обобщение метода производящих функций.

Стоит отметить, что существующие исследования в основном направлены на изучение производящих функций одной формальной переменной. Однако существует большое количество задач, связанных с производящими функциями многих переменных, в том числе для которых неизвестны явные представления (но известны производящие функции), например, задача нахождения вероятности наличия конкретного ребра в идеальном графе для ацтекского бриллианта, задача подсчета всех гистограмм и задача нахождения явных формул для числовых треугольников Дика и Моцкина с кратностями.

Также, как и для одномерного случая, одной из ключевых задач при работе с многомерными производящими функциями выделяют определение соответствующих им коэффициентов. Тем не менее для производящих функций многих переменных методы вычисления их коэффициентов развиты недостаточно. Поэтому формирование общего подхода исследования степеней производящих функций многих переменных с целью получения явных представлений для различных информационных объектов имеет важное значение, чему и уделяется внимание в дальнейших главах диссертации.

Глава 2. Комплексный метод формирования информационных объектов, основанный на k-й степени производящих функций

В данной главе представлены основные результаты в области разработки математического исчисления над коэффициентами степеней производящих функций для формирования и описания информационных объектов. Излагаются методы получения коэффициентов степеней производящих функций для одномерного, двумерного и трехмерного случая, а также для случая n-мерных рациональных производящих функций. С точки зрения приложения разработанных подходов найдены явные выражения для ряда известных последовательностей, взятых из онлайн-энциклопедии целочисленных последовательностей, и ряда информационных объектов. Рассматриваются вопросы получения свойств и явных представлений для различных классов полиномов и числовых треугольников.

2.1 Методы получения явных выражений коэффициентов степеней производящих функций для случая одной переменной

В работах [68; 89; 90] для производящей функции $F(x) = \sum\limits_{n>0} f(n)\, x^n$ без свободного члена, то есть выполняется условие $f(0) = 0$, вводится понятие композиты производящей функции. Также в данных работах определен соответствующий математический аппарат по работе с композитами производящих функций одной переменной.

2.1.1 Композита одномерной производящей функции

Определение 17. *Композитой производящей функции* $F(x) = \sum\limits_{n>0} f(n)\, x^n$ *будем называть последовательность коэффициентов производящей функции* $F(x)^k$ *и обозначать через* $F^{\Delta}(n,k)$*:*

$$(F(x))^k = \sum_{n>0} F^{\Delta}(n,k)\, x^n.$$

Выражение композиты $F^{\Delta}(n,k)$ производящей функции $F(x) = \sum\limits_{n>0} f(n)\, x^n$ может быть получено через понятие композиции натурального числа n:

$$F^{\Delta}(n,k) = \sum_{\pi_k \in C_n} f(\lambda_1)\, f(\lambda_2) \cdots f(\lambda_k), \tag{2.1}$$

где C_n — множество всех композиций π_k натурального числа n (множество всех представлений числа n через k слагаемых: $\sum\limits_{i=1}^{k} \lambda_i = n$).

Также композита $F^{\Delta}(n,k)$ может быть получена через разбиения [10]:

$$F^{\Delta}(n,k) = \sum \binom{k}{j_1\, j_2 \ldots\, j_{n-k+1}} f(1)^{j_1}\, f(2)^{j_2}\, \ldots\, f(n-k+1)^{j_{n-k+1}},$$

при этом суммирование организуется для всех неотрицательных целых чисел $j_1, j_2, \ldots, j_{n-k+1}$ $(k = \overline{1,n})$, для которых выполняется условие

$$\begin{cases} j_1 + j_2 + \ldots + j_{n-k+1} = k; \\ j_1 + 2j_2 + \ldots + (n-k+1)j_{n-k+1} = n. \end{cases}$$

Кроме того, справедливо следующее рекуррентное соотношение:

$$F^{\Delta}(n,k) = \begin{cases} f(n), & k = 1; \\ \sum\limits_{i=1}^{n-k+1} f(i) F^{\Delta}(n-i,k-1), & k \leqslant n. \end{cases}$$

Отметим, что для производящей функции $F(x) = \sum\limits_{n>0} f(n)\, x^n$ соответствующая ей композита $F^{\Delta}(n,k)$ всегда существует, причем в единственном виде.

Пример 1. *Пусть задана производящая функция* $F(x) = \frac{x}{1-x} = \sum\limits_{n>0} x^n$, *где* $f(0) = 0$ *и* $f(n) = 1$ *для* $n > 0$, *тогда соответствующая ей композита будет иметь вид*

$$F^{\Delta}(n,k) = \binom{n-1}{k-1}.$$

Данное выражение генерирует известный числовой треугольник Паскаля.

Если даны производящие функции $A(x) = \sum\limits_{n>0} a(n)\, x^n$, $F(x) = \sum\limits_{n>0} f(n)\, x^n$, $G(x) = \sum\limits_{n>0} g(n)\, x^n$ и их композиты $A^{\Delta}(n,k)$, $F^{\Delta}(n,k)$, $G^{\Delta}(n,k)$, а также производящие функции $B(x) = \sum\limits_{n\geqslant 0} b(n)\, x^n$ и $H(x) = \sum\limits_{n\geqslant 0} h(n)\, x^n$, тогда:

1. Если $A(x) = xF(x)$, то

$$A^{\Delta}(n,k) = F^{\Delta}(n-k,k);$$

2. Если $A(x) = \alpha F(x)$, то

$$A^{\Delta}(n,k) = \alpha^k F^{\Delta}(n,k);$$

3. Если $A(x) = F(\alpha x)$, то

$$A^{\Delta}(n,k) = \alpha^n F^{\Delta}(n,k);$$

4. Если $A(x) = F(x) \cdot G(x)$, то

$$A^{\Delta}(n,k) = \sum_{i=k}^{n-k} F^{\Delta}(i,k) G^{\Delta}(n-i,k);$$

5. Если $A(x) = F(x) + G(x)$, то

$$A^{\Delta}(n,k) = F^{\Delta}(n,k) + G^{\Delta}(n,k) + \sum_{j=1}^{k-1} \binom{k}{j} \sum_{i=j}^{n-k+j} F^{\Delta}(i,j) G^{\Delta}(n-i,k-j);$$

6. Если $B(x) = H(F(x))$, то

$$b(n) = \begin{cases} h(0), & n = 0; \\ \sum\limits_{k=1}^{n} h(k) F^{\Delta}(n,k), & n > 0; \end{cases} \tag{2.2}$$

7. Если $A(x) = G(F(x))$, то

$$A^{\Delta}(n,k) = \sum_{m=k}^{n} F^{\Delta}(n,m) G^{\Delta}(m,k).$$

2.1.2 Обращение производящих функций относительно операции умножения производящих функций

Также, как и для производящих функций, процесс обращения композит требует отдельного внимания и исследования. Рассмотрим сначала процесс обращения относительно операции умножения производящих функций.

Пусть производящие функции $H(x) = \sum\limits_{n \geqslant 0} h(n)\, x^n$ и $B(x) = \sum\limits_{n \geqslant 0} b(n)\, x^n$ являются взаимными и $H(x) \cdot B(x) = 1$. Тогда композитой взаимной производящей функции для $B(x)$ будем считать композиту производящей функции $xH(x)$.

Теорема 1. *Пусть заданы две производящие функции* $H(x) = \sum\limits_{n\geqslant 0} h(n)\,x^n$ *и* $B(x) = \sum\limits_{n\geqslant 0} b(n)\,x^n$, *для которых выполняется условие* $H(x)\cdot B(x) = 1$, *а также известна композита* $B_x^{\Delta}(n,k)$ *для производящей функции* $xB(x)$. *Тогда композита производящей функции* $xH(x)$ *будет равна*

$$H_x^{\Delta}(n,k) = \begin{cases} \frac{1}{B_x^{\Delta}(1,1)^k}, & n = k; \\ \sum\limits_{m=1}^{n-k} \binom{m+k-1}{k-1} \sum\limits_{j=1}^{m} \frac{(-1)^j \binom{m}{j}}{B_x^{\Delta}(1,1)^{j+k}} B_x^{\Delta}(n-k+j,j), & n > k. \end{cases} \tag{2.3}$$

Доказательство:

Из уравнения $H(x)\cdot B(x) = 1$ выразим

$$xH(x) = \frac{x}{B(x)} = \frac{x}{b(0) + B(x) - b(0)} = \frac{1}{b(0)} \frac{x}{1 + \frac{B(x)-b(0)}{b(0)}}.$$

Зная композиту производящей функции $xB(x)$ и используя приведенные выше правила вычислений композит, для производящей функции

$$F(x) = \frac{B(x) - b(0)}{b(0)}$$

получим выражение ее композиты

$$F^{\Delta}(n,k) = \sum_{j=1}^{k} b(0)^{-j}\,(-1)^{k-j} \binom{k}{j} B_x^{\Delta}(n+j,j).$$

Коэффициенты производящей функции

$$R(x)^k = \left(\frac{1}{b(0)} \frac{1}{1+x}\right)^k$$

определяются выражением

$$R(n,k) = \frac{1}{b(0)^k} \binom{n+k-1}{k-1} (-1)^n.$$

Тогда, используя правило композиции для $(xH(x))^k = R(F(x))^k$, получим

$$H(n,k) = \begin{cases} \frac{1}{b(0)^k}, & n = 0; \\ \sum_{m=1}^{n} \binom{m+k-1}{k-1} \sum_{j=1}^{m} \frac{(-1)^j \binom{m}{j}}{b(0)^{k+j}} B^{\Delta}(n+j,j), & n > 0. \end{cases}$$

Откуда получаем искомую формулу (2.3). □

В качестве приложения Теоремы 1 рассмотрим следующий пример.

Пример 2. *Найдем явное выражение для композиты производящей функции* $F(x) = x^2 \operatorname{cosec}(x)$. *Для этого выразим*

$$F(x) = x^2 \operatorname{cosec}(x) = \frac{x}{\frac{\sin(x)}{x}},$$

в следующей форме:

$$\frac{F(x)}{x}\frac{\sin(x)}{x} = 1.$$

Композита для $\sin(x)$ *равна*

$$\frac{1+(-1)^{n-k}}{2^k n!}\sum_{m=0}^{\lfloor \frac{k}{2} \rfloor}\binom{k}{m}(2m-k)^n(-1)^{\frac{n+k}{2}-m}.$$

Тогда, используя Теорему 1, получим следующее выражение композиты производящей функции $F(x)$: *если* $n > k$, *то*

$$F^{\Delta}(n,k) = \sum_{m=1}^{n-k}\binom{m+k-1}{k-1}\sum_{j=1}^{m}\binom{m}{j}\frac{1+(-1)^{n-k}}{2^j(n-k+j)!}\times$$
$$\times\sum_{i=0}^{\lfloor \frac{j}{2} \rfloor}\binom{j}{i}(2i-j)^{n-k+j}(-1)^{\frac{n-k}{2}-i}.$$

Далее получим явную формулу для коэффициентов взаимной производящей функции, возведенной в степень α.

Теорема 2. *Пусть задана производящая функция* $B(x) = \sum\limits_{n\geqslant 0} b(n)\,x^n$, $b(0) \neq 0$, *а также известны коэффициенты* $b(n,k) = [x^n]B(x)^k$. *Тогда явная формула для коэффициентов производящей функции*

$$H(x,\alpha) = \sum_{n\geqslant 0} h(n,\alpha)x^n = \left(\frac{1}{B(x)}\right)^{\alpha}$$

имеет вид

$$h(n,\alpha) = \sum_{k=0}^{n} b(0)^{-k-\alpha}(-1)^k\binom{k+\alpha-1}{k}\binom{n+\alpha}{n-k}b(n,k). \tag{2.4}$$

Доказательство:

Сначала рассмотрим случай, когда $b(0) = 1$. Представим $H(x,\alpha)$ как композицию двух производящих функций $R(F(x),\alpha)$, где

$$R(x,\alpha) = \left(\frac{1}{1+x}\right)^{\alpha} = \sum_{n\geqslant 0}\binom{n+\alpha-1}{n}(-x)^n$$

и

$$F(x) = B(x) - 1.$$

Найдем композиту $F^{\Delta}(n,k)$ через биномиальную теорему

$$F^{\Delta}(n,k) = [x^n](B(x)-1)^k = [x^n]\sum_{j=0}^{k}\binom{k}{j}B(x)^j(-1)^{k-j} = \sum_{j=0}^{k}\binom{k}{j}b(n,j)(-1)^{k-j}.$$

Поскольку $f(0) = 0$, применим правило (2.2) и получим

$$h(n,\alpha) = \sum_{k=1}^{n} F^{\Delta}(n,k)r(k), \qquad h(0) = r(0).$$

Тогда

$$h(n,\alpha) = \sum_{k=0}^{n}\sum_{j=0}^{k}\binom{k}{j}b(n,j)(-1)^j\binom{k+\alpha-1}{k}.$$

Изменим порядок суммирования в двойной сумме, получим

$$h(n,\alpha) = \sum_{j=0}^{n}(-1)^j\binom{j+\alpha-1}{j}b(n,j)\sum_{k=0}^{n-j}\binom{j+k+\alpha-1}{k}.$$

Используя известное тождество [42]

$$\sum_{k=0}^{m}\binom{s+k-1}{k} = \binom{s+m}{m}$$

и подставляя $m = n - j$ и $s = j + \alpha$, получим

$$\sum_{k=0}^{n-j}\binom{j+k+\alpha-1}{k} = \binom{n+\alpha}{n-j}.$$

Тогда искомая формула для случая $b(0) = 1$ равна

$$h(n,\alpha) = \sum_{j=0}^{n}(-1)^j\binom{j+\alpha-1}{j}\binom{n+\alpha}{n-j}b(n,j).$$

Теперь обобщим на случай $b(0) \neq 0$:

$$H(x,\alpha) = \left(\frac{1}{b(0)+B(x)-b(0)}\right)^{\alpha} = b(0)^{-\alpha}\left(\frac{1}{1+\frac{B(x)}{b(0)}-1}\right)^{\alpha}.$$

Также получим коэффициенты производящей функции следующего вида:

$$[x^n]\left(\frac{B(x)}{b(0)}-1\right)^k = F^{\Delta}(n,k)b(0)^{-k}.$$

Откуда получим искомую формулу (2.4). □

Также формулу (2.4) можно представить следующим образом:

Теорема 3. *Пусть задана производящая функция* $B(x)=\sum\limits_{n\geqslant 0} b(n)\,x^n$, $b(0)\neq 0$, *а также известны коэффициенты* $b(n,k)=[x^n]B(x)^k$. *Тогда явная формула для коэффициентов взаимной производящей функции* $H(x)$, *возведенной в степень* α, *имеет вид*

$$h(n,\alpha)=(n+1)\binom{n+\alpha}{n+1}\sum_{k=0}^{n}\frac{(-1)^k}{(k+\alpha)b(0)^{k+\alpha}}\binom{n}{k}b(n,k), \tag{2.5}$$

Доказательство:

Рассмотрим произведение биномиальных коэффициентов в формуле (2.4) и представим его как

$$\binom{k+\alpha-1}{k}\binom{n+\alpha}{n-k}=\binom{k+\alpha-1}{k}\binom{n+\alpha}{k+\alpha}.$$

Применяя гамма-функции, получим следующее выражение для приведенного произведения биномиальных коэффициентов:

$$\binom{k+\alpha-1}{k}\binom{n+\alpha}{k+\alpha}=\frac{\Gamma(k+\alpha)}{\Gamma(k+1)\Gamma(\alpha)}\frac{\Gamma(n+\alpha+1)}{\Gamma(k+\alpha+1)\Gamma(n-k+1)}=$$

$$=\frac{\Gamma(k+\alpha)\Gamma(n+\alpha+1)}{(k+\alpha)k!\Gamma(\alpha)\Gamma(k+\alpha)(n-k)!}=\frac{(n+\alpha)\Gamma(n+\alpha)}{(k+\alpha)\Gamma(\alpha)k!(n-k)!}=\frac{n+1}{k+\alpha}\binom{n+\alpha}{n+1}\binom{n}{k}$$

Подставляя данное выражение в (2.4), получим искомую формулу (2.5). □

Следствие 1. *Пусть задана производящая функция* $B(x)=\sum\limits_{n\geqslant 0} b(n)\,x^n$, $b(0)\neq 0$, *а также известны коэффициенты* $b(n,k)=[x^n]B(x)^k$. *Тогда явная формула для коэффициентов взаимной производящей функции* $H(x)$ *имеет вид*

$$h(n)=\sum_{k=0}^{n}(-1)^k\binom{n+1}{k+1}b(n,k).$$

В качестве приложения рассмотрим несколько примеров.

Пример 3. *Пусть дана производящая функция* $H(x) = \frac{1}{\sqrt{B(x)}}$. *Тогда для случая* $\alpha = \frac{1}{2}$ *формула* (2.5) *будет иметь вид*

$$h(n) = (n+1)\binom{n+\frac{1}{2}}{n+1}\sum_{k=0}^{n}\frac{(-1)^k}{(k+\frac{1}{2})b(0)^{k+\frac{1}{2}}}\binom{n}{k}b(n,k).$$

Поскольку

$$\binom{n+\frac{1}{2}}{n+1} = \frac{1}{2\,4^n}\binom{2n+1}{n+1},$$

то получаем

$$h(n) = \frac{(n+1)}{4^n}\binom{2n+1}{n+1}\sum_{k=0}^{n}\frac{(-1)^k}{(2k+1)b(0)^{k+\frac{1}{2}}}\binom{n}{k}b(n,k). \tag{2.6}$$

Пусть дана производящая функция $H(x) = \frac{1}{\sqrt{B(x)}} = \sqrt{1-x}$. *Тогда для производящей функции*

$$B(x) = \frac{1}{1-x}$$

коэффициенты ее степени $B(x)^k$ *будут равны*

$$b(n,k) = \binom{n+k-1}{n}.$$

С одной стороны, коэффициенты производящей функции $H(x) = \sqrt{1-x}$ *равны*

$$h(n) = -\frac{2}{4^n n}\binom{2n-2}{n-1}.$$

С другой стороны, на основании (2.6) *получаем*

$$h(n) = \frac{(n+1)}{4^n}\binom{2n+1}{n+1}\sum_{k=0}^{n}\frac{(-1)^k}{2k+1}\binom{n}{k}\binom{n+k-1}{n}.$$

Тогда получим следующее тождество:

$$\frac{(n+1)}{4^n}\binom{2n+1}{n+1}\sum_{k=0}^{n}\frac{(-1)^k}{2k+1}\binom{n}{k}\binom{n+k-1}{n} = -\frac{2}{4^n n}\binom{2n-2}{n-1}$$

или

$$\sum_{k=0}^{n}\frac{(-1)^k\binom{n}{k}\binom{n+k-1}{n}}{2k+1} = \frac{1}{1-4n^2}.$$

Пусть дана производящая функция $H(x) = \sqrt{\frac{x^2}{(\exp(x)-1)^2}}$. *Тогда получим следующее тождество:*

$$\sum_{k=0}^{n} \frac{(-1)^k (2k)! \binom{n}{k} S(n+2k,2k)}{(2k+1)(n+2k)!} = \frac{2\,4^n B(n)(n+1)!}{(2n+2)!},$$

где $S(n,k)$ — *числа Стирлинга второго рода,* $B(n)$ — *числа Бернулли.*

Пример 4. *Полиномы Норлунда относительно* α *(числа Бернулли порядка* α*) определяются следующей экспоненциальной производящей функцией [43]:*

$$\left(\frac{x}{e^x - 1}\right)^{\alpha} = \sum_{n \geqslant 0} B_n^{(\alpha)} \frac{x^n}{n!}.$$

Поскольку

$$\left(\frac{e^x - 1}{x}\right)^{k} = \sum_{n \geqslant 0} S(n+k,k) \frac{k!}{(n+k)!} x^n,$$

где $S(n,k)$ — *числа Стирлинга второго рода, то на основании формулы* (2.5) *получим следующую явную формулу для полиномов Норлунда:*

$$B_n^{(\alpha)} = (n+1)\binom{n+\alpha}{n+1} \sum_{k=0}^{n} \frac{(-1)^k}{(k+\alpha)\binom{n+k}{n}} \binom{n}{k} S(n+k,k).$$

Пример 5. *Числа Бернулли второго рода определяются следующей экспоненциальной производящей функцией [43]:*

$$\left(\frac{x}{\log(1+x)}\right)^{\alpha} = \sum_{n \geqslant 0} b(n,\alpha) \frac{x^n}{n!}.$$

Согласно формуле (2.5) *и на основании*

$$\left(\frac{\log(1+x)}{x}\right)^{k} = \sum_{n \geqslant 0} s(n+k,k) \frac{k!}{(n+k)!} x^n,$$

где $s(n,k)$ — *числа Стирлинга первого рода, получим следующую явную формулу для чисел Бернулли второго рода:*

$$b(n,\alpha) = \sum_{k=0}^{n} (-1)^k \binom{k+\alpha-1}{k} \binom{n+\alpha}{n-k} \binom{n+k}{k}^{-1} s(n+k,k)$$

или

$$b(n,\alpha) = (n+1)\binom{n+\alpha}{n+1} \sum_{k=0}^{n} \frac{(-1)^k}{(k+\alpha)\binom{n+k}{n}} \binom{n}{k} s(n+k,k).$$

2.1.3 Обращение производящих функций относительно операции композиции производящих функций

Далее рассмотрим процесс обращения относительно операции композиции производящих функций. Известные методы, основанные на формуле обращения Лагранжа, применяются только для ограниченного случая простых производящих функций [65]. Прямого и легко организуемого способа решения задачи нахождения явных выражений для коэффициентов обратных производящих функций не известно. Далее представлена теорема, позволяющая получать явные формулы для композиты обратных производящих функций.

Теорема 4. *Пусть задана производящая функция* $F(x) = \sum\limits_{n>0} f(n)\, x^n$, $f(1) \neq 0$, *а также известна ее композита* $F^{\Delta}(n,k)$. *Тогда композита обратной производящей функции* $A(x)$ *имеет вид*

$$A^{\Delta}(n,m) = \begin{cases} \frac{1}{F^{\Delta}(1,1)^n}, & n = m; \\ \frac{m}{n\, F^{\Delta}(1,1)^n} \sum\limits_{k=1}^{n-m} \binom{n+k-1}{n-1} \sum\limits_{j=1}^{k} (-1)^j \frac{F^{\Delta}(n-m+j,j)}{F^{\Delta}(1,1)^j} \binom{k}{j}, & n > m. \end{cases} \tag{2.7}$$

Доказательство:

Согласно формуле обращения Лагранжа для функционального уравнения вида $A(x) = xH(A(x))$ выполняется следующее соотношение:

$$[x^n]A(x)^k = \frac{k}{n}[x^{n-k}]H(x)^n.$$

Левая часть данного уравнения совпадает с определением композиты производящей функции $A(x)$, то есть

$$n\,[x^n]A(x)^k = n\,A^{\Delta}(n,k).$$

Рассмотрим k-ю степень производящей функции $xH(x)$

$$(xH(x))^k = \sum_{n \geqslant k} H_x^{\Delta}(n,k)\, x^n$$

и выразим через ее коэффициенты коэффициенты k-й степени производящей функции $H(x)$, получим

$$(H(x))^k = \sum_{n \geqslant k} H_x^{\Delta}(n,k) x^{n-k} = \sum_{m \geqslant 0} H_x^{\Delta}(m+k,k) x^m.$$

Отсюда выразим коэффициенты производящей функции $H(x)^n$ при степенях x^{n-k}, получим

$$[x^{n-k}]H(x)^n = H_x^{\Delta}(2n-k,n).$$

Объединяем данный результат с формулой обращения Лагранжа, тогда

$$A^{\Delta}(n,k) = \frac{k}{n} H_x^{\Delta}(2n-k,n). \tag{2.8}$$

Уравнение (1.2), определяющее обратные производящие функции, можно представить как

$$A(x)B(A(x)) = x,$$

где

$$B(x) = \frac{F(x)}{x}.$$

Также введем производящую функцию $H(x)$, которая является взаимной производящей функцией для $B(x)$, то есть $H(x) \cdot B(x) = 1$. Тогда получим запись уравнения, соответствующего формуле обращения Лагранжа,

$$A(x) = \frac{x}{B(A(x))} = xH(A(x)).$$

Объединяя формулу (2.8) с результатом Теоремы 1, получим искомую формулу (2.7). □

Следствие 2. *Пусть задана производящая функция* $F(x) = \sum\limits_{n>0} f(n)\, x^n$, $f(1) = 1$, *а также известна ее композита* $F^{\Delta}(n,k)$. *Тогда явная формула для коэффициентов обратной производящей функции* $A(x)$ *имеет вид*

$$a(n) = \begin{cases} 1, & n = 1; \\ \frac{1}{n} \sum\limits_{k=1}^{n-1} \binom{n+k-1}{n-1} \sum\limits_{j=1}^{k} (-1)^j \binom{k}{j} F^{\Delta}(n+j-1,j), & n > 1. \end{cases}$$

Пример 6. *Пусть дана производящая функция* $F(x) = x - x^2 - x^3$. *Для начала найдем явное выражение для композиты обратной производящей функции. В соответствии с доказательством для Теоремы 4 введем производящую функцию следующего вида:*

$$H(x) = \frac{x}{F(x)} = \frac{1}{1 - x - x^2}.$$

Композита производящей функции $xH(x)$ имеет выражение

$$H_x^{\Delta}(n,k) = \begin{cases} 1, & n = k; \\ \sum\limits_{i=1}^{n-k} \binom{i}{n-k-i}\binom{k+i-1}{i}, & n > k. \end{cases}$$

Применяя данную композиту к формуле (2.8)*, получаем следующее явное выражение для композиты обратной производящей функции для $F(x)$:*

$$A^{\Delta}(n,k) = \begin{cases} 1, & n = k; \\ \frac{k}{n}\sum\limits_{i=1}^{n-k} \binom{i}{n-k-i}\binom{n+i-1}{n-1}, & n > k. \end{cases}$$

Тогда коэффициенты обратной производящей функции для $F(x)$ имеют следующий вид: для $n > 1$

$$a(n) = \frac{1}{n}\sum_{k=1}^{n-1} \binom{k}{n-1-k}\binom{k+n-1}{n-1}.$$

2.2 Методы получения явных выражений коэффициентов степеней производящих функций для случая двух переменных

Для производящих функций многих переменных соответствующий математический аппарат развит слабо, а в общем виде применение коэффициентов степеней производящих функций многих переменных исследовано недостаточно. Поэтому задачами данного диссертационного исследования является выделение класса производящих функций многих переменных и разработка новых методов получения явных выражений коэффициентов их степеней, а именно методов получения коэффициентов степеней взаимных, обратных, сложения, умножения и композиции производящих функций многих переменных.

Для начала определим общий вид k-й степени многомерной производящей функции:

$$F(x,y,\ldots,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\cdots\sum_{l\geqslant 0} f(n,m,\ldots,l,k)x^n y^m\cdots z^l,$$

где $k \in \mathbb{N}_0$.

Тогда для $F(x,y,\ldots,z)$ с порядком $ord(F) > 0$, порядок для $F(x,y,\ldots,z)^k$, $k \in \mathbb{N}$, будет удовлетворять неравенству

$$ord(F^k) \geqslant k. \tag{2.9}$$

В общем виде коэффициенты $f(n,m,\ldots,l,k)$ определяются как

$$f(n,m,\ldots,l,k) =$$

$$= \sum_{\eta_1+\ldots+\eta_k=n} \left(\sum_{\mu_1+\ldots+\mu_k=m} \left(\ldots \left(\sum_{\lambda_1+\ldots+\lambda_k=l} \left(\prod_{i=1}^{k} f(\eta_i,\mu_i,\ldots,\lambda_i) \right) \right) \ldots \right) \right),$$

где $\eta_i,\mu_i,\ldots,\lambda_i \in \mathbb{N}_0$. Здесь суммирование ведется по всем разложениям чисел $n,m,\ldots,l$ ровно на k членов.

Также можно записать рекуррентное соотношение для вычисления коэффициентов на основе следующей формулы:

$$F(x,y,\ldots z)^k = F(x,y,\ldots z)^{k-1} F(x,y,\ldots z).$$

Чтобы степень $F(x,y,\ldots,z)^k$ можно было использовать в операции композиции производящих функций необходимо и достаточно, чтобы выполнялось условие (2.9). Из этого следует, что для одного из разложений должно быть выполнено условие, что элементы разложения должны быть больше нуля ($\lambda_i > 0$, $\mu_i > 0$ и т.д). Если выполняется данное условие, например, $\lambda_i > 0$, то при фиксированных остальных переменных $m,\ldots,k$ значения для $F(n,m_0,\ldots,k_0)$ будут представлять собой числовой треугольник.

Для того чтобы выделить класс коэффициентов степеней формальных степенных рядов, введем понятие композиты многомерной производящей функции.

Определение 18. *Композита $F^{\Delta}(n,m,\ldots,l,k)$ многомерной производящей функции*

$$F(x,y,\ldots,z) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} \ldots \sum_{l\geqslant 0} f(n,m,\ldots,k) x^n y^m \cdots z^l, \quad ord(F) \geqslant 1,$$

есть функция коэффициентов k-й степени многомерной производящей функции $F(x,y,\ldots,z)$:

$$F(x,y,\ldots,z)^k = \sum_{n\geqslant 0} \sum_{m\geqslant 0} \ldots \sum_{l\geqslant 0} F^{\Delta}(n,m,\ldots,l,k) x^n y^m \cdots z^l.$$

Пусть нулевая степень как и для одномерного случая будет определяться как $F(x,y,\ldots,z)^0 = 1$, тогда композита $F^{\Delta}(n,m,\ldots,l,k)$ для случая $k = 0$ будет определяться следующим образом:

$$F^{\Delta}(n,m,\ldots,l,0) = \begin{cases} 1, & n = m = \ldots = l = 0; \\ 0, & \text{иначе.} \end{cases}$$

Учитывая, что

$$F(x,y,\ldots,z)^k = F(x,y,\ldots,z)F(x,y,\ldots,z)^{k-1}, \quad F(x,y,\ldots,z)^0 = 1.$$

получим следующее рекуррентное выражение для вычисления композит:

$$F^{\Delta}(n,m,\ldots,l,k) =$$

$$= \begin{cases} f(n,m,\ldots,l), & k = 1; \\ \sum\limits_{i=0}^{n}\sum\limits_{j=0}^{m}\ldots\sum\limits_{s=0}^{l} f(i,j,\ldots,s)F^{\Delta}(n-i,m-j,\ldots,l-s,k-1), & k > 1. \end{cases}$$

2.2.1 Композита двумерной производящей функции

Определение 19. *Двумерной производящей функцией $F(x,y)$ будем называть следующий формальный степенной ряд:*

$$F(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} f(n,m)x^n y^m.$$

Тогда k-я степень двумерной производящей функции $F(x,y)$, для которой $ord(F) \geqslant 1$, определяется как

$$F(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} F^{\Delta}(n,m,k)x^n y^m,$$

где $F^{\Delta}(n,m,k)$ — композита двумерной производящей функции.

В общем случае для вычисления коэффициентов $F(x,y)^k$ можно воспользоваться вычислительным способом, основанным на формуле

$$F(n,m,k) = \sum_{\lambda_1+\lambda_2+\ldots\lambda_k=n}\left(\sum_{\mu_1+\mu_2+\ldots\mu_k=m}\left(\prod_{i=1}^{k} f(\lambda_i,\mu_i)\right)\right),$$

где $\lambda_i \geqslant 0$ и $\mu_k \geqslant 0$. В данном случае суммирование ведется по всем разложениям чисел n и m.

Однако данный путь вычисления является более трудоемким, поскольку формула имеет высокую вычислительную сложность

$$O(n,m,k) = k\,O_f(n,m)\,gr(n,k)\,gr(m,k)\binom{n+k-1}{n}\binom{m+k-1}{m},$$

где k — количество умножений, $O_f(n,m)$ — количество операций для вычисления значений $f(n,m)$, $gr(n,k)$ — количество операций для генерации разложения числа n на k членов.

Для приближенной оценки количества итераций положим $n = m = k$, тогда получим

$$O(n,n,n) = n\,O_f(n,n)\,gr(n,n)^2\binom{2n-1}{n}^2.$$

Положим

$$gr(n,n) \backsim n^2$$

и

$$\binom{2n}{n} \backsim \frac{2^{2n}}{\sqrt{n\pi}},$$

тогда вычислительная сложность будет не меньше, чем

$$O(n,n,n) \geqslant \frac{n^5\,2^{4n}\,O_f(n,n)}{\sqrt{n\pi}}.$$

С учетом эквивалентности

$$\binom{n+k-1}{n} \backsim \sqrt{\frac{n+k-1}{2\pi\,n\,(k-1)}}\frac{(n+k-1)^{n+k-1}}{n^n\,(k-1)^{k-1}},$$

получим более точную приближенную оценку

$$O(n,m,k) \backsim$$

$$\backsim k\sqrt{\frac{n+k-1}{2\pi\,n\,(k-1)}}\frac{(n+k-1)^{n+k-1}}{n^{n-2}\,(k-1)^{k-1}}\sqrt{\frac{m+k-1}{2\pi\,m\,(k-1)}}\frac{(m+k-1)^{m+k-1}}{m^{m-2}\,(k-1)^{k-1}}O_f(n,m).$$

Таким образом, на основе полученной приближенной оценки требуемого количества операций видно, что такой способ является трудоемким с точки зрения вычислительной сложности. Следовательно, применение данного способа для вычисления коэффициентов $F(x,y)^k$ является неэффективным при больших значениях параметров.

С другой стороны, предлагаемый метод получения явных выражений коэффициентов k-й степени производящих функций многих переменных позволяет получать более быстрые способы вычисления коэффициентов с точки зрения вычислительной сложности.

Рассмотрим эффект использования предлагаемого метода на следующем примере: необходимо найти явную формулу для коэффициентов k-й степени производящей функции двух переменных вида

$$A(x,y) = \sin(x+y).$$

Известно [68], что для производящей функции $F(x) = \sin(x)$ ее композита равна

$$A^{\Delta}(n,k) = [x^n]\sin(x)^k = \frac{(-1)^{n-k}+1}{2^k\,n!}\sum_{i=0}^{\frac{k}{2}}(2\,i-k)^n\binom{k}{i}\,(-1)^{\frac{n+k}{2}-i}.$$

Поскольку

$$[x^n\,y^m](x+y)^k = \delta_{k,n+m}\binom{n+m}{m},$$

тогда коэффициенты k-й степени производящей функции $F(x,y) = \sin(x+y)$ равны

$$F^{\Delta}(n,m,k) = \sum_{i=0}^{n+m}\delta_{i,n+m}\binom{n+m}{m}A^{\Delta}(i,k) = \binom{n+m}{m}A^{\Delta}(n+m,k) =$$

$$= \binom{n+m}{m}\frac{(-1)^{n+m-k}+1}{2^k\,(n+m)!}\sum_{i=0}^{\frac{k}{2}}(2\,i-k)^{n+m}\binom{k}{i}\,(-1)^{\frac{n+m+k}{2}-i}.$$

Полученная формула для $F^{\Delta}(n,m,k)$ имеет полиномиальный вид вычислительной сложности. Другими словами, если производящую функцию можно декомпозировать на элементарные производящие функции, для которых известны композиты, то вычислительная сложность нахождения ее коэффициентов и коэффициентов ее степеней будет иметь полиномиальный вид.

Далее введены и подробно рассмотрены основные операции для работы с композитами двумерных производящих функций, такие как композиция, сложение, умножение и обращение производящих функций двух переменных.

2.2.2 Композиция двумерных производящих функций

Рассмотрим задачу получения явных выражений коэффициентов композиции производящих функций двух переменных.

Теорема 5. *Пусть*

$$H(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} h(n,m)x^n y^m,$$

$$A(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} A^{\Delta}(n,m,k)x^n y^m, \quad ord(A) \geqslant 1,$$

$$B(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} B^{\Delta}(n,m,k)x^n y^m, \quad ord(B) \geqslant 1.$$

Тогда коэффициенты композиции двумерных производящих функций

$$G(x,y) = H(A(x,y),B(x,y)) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m$$

будут равны

$$g(n,m) = \sum_{k_a=0}^{n+m}\sum_{k_b=0}^{n+m-k_a} h(k_a,k_b)\sum_{i=0}^{n}\sum_{j=0}^{m} A^{\Delta}(i,j,k_a)B^{\Delta}(n-i,m-j,k_b). \qquad (2.10)$$

Доказательство:

Раскроем запись заданной композиции двумерных производящих функций

$$G(x,y) = H(A(x,y),B(x,y)) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} h(n,m)A(x,y)^n B(x,y)^m.$$

Затем рассмотрим производящую функцию следующего вида:

$$C(x,y) = A(x,y)^{k_a}B(x,y)^{k_b} = \sum_{n\geqslant 0}\sum_{m\geqslant 0} c(n,m,k_a,k_b)x^n y^m.$$

Чтобы получить явную формулу для коэффициентов $c(n,m,k_a,k_b)$, используем операцию умножения степеней производящих функций и получим

$$c(n,m,k_a,k_b) = \sum_{i=0}^{n}\sum_{j=0}^{m} A^{\Delta}(i,j,k_a)B^{\Delta}(n-i,m-j,k_b).$$

С учетом ограничения $k_a + k_b \leqslant n + m$, получим следующую схему суммирования для получения коэффициентов композиции производящих функций:

$$g(n,m) = \sum_{k_a=0}^{n+m}\sum_{k_b=0}^{n+m-k_a} h(k_a,k_b)c(n,m,k_a,k_b).$$

Подставляя в формулу для $g(n,m)$ выражение для $c(n,m,k_a,k_b)$, получим искомую формулу (2.10). □

Формула (2.10) обеспечивает получение явного выражения для членов композиции формальных степенных рядов двух переменных через композиты формальных степенных рядов входящих в эту композицию. Заметим, что для композиции $G(x,y) = H(A(x,y),A(x,y))$ формула (2.10) примет следующий упрощенный вид:

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b)\, A^{\Delta}(n,m,k_a+k_b). \tag{2.11}$$

Рассмотрим другие частные случаи применения Теоремы 5.

Следствие 3. *Пусть*

$$H(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} h(n,m)x^n y^m,$$

$$A(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} A^{\Delta}(n,m,k)x^n y^m, \quad ord(A) \geqslant 1,$$

$$B(x) = \sum_{n>0} b(n)x^n, \quad B(x)^k = \sum_{n\geqslant k} B^{\Delta}(n,k)x^n.$$

Тогда коэффициенты композиции двумерных производящих функций

$$G(x,y) = H(A(x,y),B(y)) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m$$

будут равны

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{j=0}^{m} A^{\Delta}(n,j,k_a)B^{\Delta}(m-j,k_b). \tag{2.12}$$

Доказательство:

Рассмотрим производящую функцию $B(y)$ как следующую двумерную производящую функцию:

$$B_y(x,y) = B(y)x^0 = B(y).$$

Композита производящей функции $B_y(x,y)$ имеет вид

$$B_y^{\Delta}(n,m,k) = B^{\Delta}(m,k)\delta(n,0).$$

Применяя (2.10) для композиции производящих функций

$$G(x,y) = H(A(x,y),B(y)) = H(A(x,y),B_y(x,y)),$$

получим

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{i=0}^{n} \sum_{j=0}^{m} A^{\Delta}(i,j,k_a) B^{\Delta}(m-j,k_b) \delta(n-i,0).$$

Используя свойства символа Кронекера, получаем $i = n$.

Упрощая формулу для $g(n,m)$, получим искомую формулу (2.12). □

Следствие 4. *Пусть*

$$H(x,y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} h(n,m) x^n y^m,$$

$$A(x,y)^k = \sum_{n\geqslant 0} \sum_{m\geqslant 0} A^{\Delta}(n,m,k) x^n y^m, \quad ord(A) \geqslant 1.$$

Тогда коэффициенты композиции двумерных производящих функций

$$G(x,y) = H(A(x,y),y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} g(n,m) x^n y^m$$

будут равны

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) A^{\Delta}(n,m-k_b,k_a). \tag{2.13}$$

Доказательство:

Рассмотрим переменную y как следующую двумерную производящую функцию:

$$B_y(x,y) = x^0 y = y.$$

Композита производящей функции $B_y(x,y)$ имеет вид

$$B^{\Delta}(n,m,k) = \delta(n,0)\delta(m,k).$$

Применяя (2.10) для композиции производящих функций

$$G(x,y) = H(A(x,y),y) = H(A(x,y),B_y(x,y)),$$

получим

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{i=0}^{n} \sum_{j=0}^{m} A^{\Delta}(i,j,k_a)\delta(n-i,0)\delta(m-j,k_b).$$

Используя свойства символа Кронекера, получаем $i = n$, $j = m - k_b$.

Упрощая формулу для $g(n,m)$, получим искомую формулу (2.13). □

Следствие 5. *Пусть*

$$H(x,y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} h(n,m)x^n y^m,$$

$$A(x) = \sum_{n>0} a(n)x^n, \quad A(x)^k = \sum_{n\geqslant k} A^{\Delta}(n,k)x^n,$$

$$B(x) = \sum_{n>0} b(n)x^n, \quad B(x)^k = \sum_{n\geqslant k} B^{\Delta}(n,k)x^n.$$

Тогда коэффициенты композиции двумерных производящих функций

$$G(x,y) = H(A(x),B(y)) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} g(n,m)x^n y^m$$

будут равны

$$g(n,m) = \sum_{k_a=0}^{n} \sum_{k_b=0}^{m} h(k_a,k_b)A^{\Delta}(n,k_a)B^{\Delta}(m,k_b). \tag{2.14}$$

Доказательство:

Рассмотрим производящую функцию $A(x)$ как следующую двумерную производящую функцию:

$$A_x(x,y) = A(x)y^0 = A(x).$$

Композита производящей функции $A_x(x,y)$ имеет вид

$$A_x^{\Delta}(n,m,k) = A^{\Delta}(n,k)\delta(m,0).$$

Рассмотрим производящую функцию $B(y)$ как следующую двумерную производящую функцию:

$$B_y(x,y) = B(y)x^0 = B(y).$$

Композита производящей функции $B_y(x,y)$ имеет вид

$$B_y^{\Delta}(n,m,k) = B^{\Delta}(m,k)\delta(n,0).$$

Применяя (2.10) для композиции производящих функций

$$G(x,y) = H(A(x),B(y)) = H(A_x(x,y),B_y(x,y)),$$

получим

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{i=0}^{n} \sum_{j=0}^{m} A^{\Delta}(i,k_a)\delta(j,0)B^{\Delta}(m-j,k_b)\delta(n-i,0).$$

Используя свойства символа Кронекера, получаем $i = n$, $j = 0$.

Упрощая формулу для $g(n,m)$, получим искомую формулу (2.14). □

Следствие 6. *Пусть*

$$H(x,y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} h(n,m)x^n y^m,$$

$$A(x) = \sum_{n>0} a(n)x^n, \quad A(x)^k = \sum_{n\geqslant k} A^{\Delta}(n,k)x^n.$$

Тогда коэффициенты композиции двумерных производящих функций

$$G(x,y) = H(A(x),y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} g(n,m)x^n y^m$$

будут равны

$$g(n,m) = \sum_{k=0}^{n} h(k,m)A^{\Delta}(n,k). \tag{2.15}$$

Доказательство:

Рассмотрим производящую функцию $A(x)$ как следующую двумерную производящую функцию:

$$A_x(x,y) = A(x)y^0 = A(x).$$

Композита производящей функции $A_x(x,y)$ имеет вид

$$A_x^{\Delta}(n,m,k) = A^{\Delta}(n,k)\delta(m,0).$$

Рассмотрим переменную y как следующую двумерную производящую функцию:

$$B_y(x,y) = x^0 y = y.$$

Композита производящей функции $B_y(x,y)$ имеет вид

$$B_y^{\Delta}(n,m,k) = \delta(n,0)\delta(m,k).$$

Применяя (2.10) для композиции производящих функций

$$G(x,y) = H(A(x),y) = H(A_x(x,y),B_y(x,y)),$$

получим

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{i=0}^{n} \sum_{j=0}^{m} A^{\Delta}(i,k_a)\delta(j,0)\delta(n-i,0)\delta(m-j,k_b).$$

Используя свойства символа Кронекера, получаем $i = n$, $j = 0$, $k_b = m$. Упрощая формулу для $g(n,m)$, получим искомую формулу (2.15). □

Следствие 7. *Пусть*

$$H(x) = \sum_{n \geqslant 0} h(n)x^n,$$

$$A(x,y)^k = \sum_{n \geqslant 0} \sum_{m \geqslant 0} A^{\Delta}(n,m,k)x^n y^m, \quad ord(A) \geqslant 1.$$

Тогда коэффициенты композиции двумерных производящих функций

$$G(x,y) = H(A(x,y)) = \sum_{n \geqslant 0} \sum_{m \geqslant 0} g(n,m)x^n y^m$$

будут равны

$$g(n,m) = \sum_{k=0}^{n+m} h(k)A^{\Delta}(n,m,k). \tag{2.16}$$

Доказательство:

Рассмотрим производящую функцию $H(x)$ как следующую двумерную производящую функцию:

$$H_x(x,y) = H(x)y^0 = H(x).$$

Коэффициенты производящей функции $H_x(x,y)$ имеют вид

$$h_x(n,m) = h(n)\delta(m,0).$$

Рассмотрим двумерную производящую функцию

$$B(x) = \sum_{n>0} b(n)x^n, \quad B(x)^k = \sum_{n \geqslant k} B^{\Delta}(n,k)x^n.$$

Применяя (2.10) для композиции производящих функций

$$G(x,y) = H(A(x,y)) = H_x(A(x,y),B(x,y)),$$

получим

$$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a)\delta(k_b,0) \sum_{i=0}^{n} \sum_{j=0}^{m} A^{\Delta}(i,j,k_a) B^{\Delta}(n-i,m-j,k_b).$$

Используя свойства символа Кронекера, получаем $k_b = 0$. Согласно свойству нулевой степени композиты $B^{\Delta}(n-i,m-j,0)$ равно 1 только при $n-i=0$ и $m-j=0$, иначе оно равно 0. Следовательно, получаем $i=n$, $j=m$.

Упрощая формулу для $g(n,m)$, получим искомую формулу (2.16). □

В таблице 2.1 показаны полученные результаты для частных случаев использования результатов Теоремы 5 для двумерной производящей функции $G(x,y)$, представленной в виде композиции производящих функций.

Таблица 2.1 — Частные случаи Теоремы 5

Композиция	Явная формула
$G(x,y) = H(A(x,y),B(y))$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{j=0}^{m} A^{\Delta}(n,j,k_a) B^{\Delta}(m-j,k_b)$
$G(x,y) = H(A(y),B(x,y))$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{j=0}^{m} A^{\Delta}(j,k_a) B^{\Delta}(n,m-j,k_b)$
$G(x,y) = H(A(x,y),B(x))$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{i=0}^{n} A^{\Delta}(i,m,k_a) B^{\Delta}(n-i,k_b)$
$G(x,y) = H(A(x),B(x,y))$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) \sum_{i=0}^{n} A^{\Delta}(i,k_a) B^{\Delta}(n-i,m,k_b)$
$G(x,y) = H(A(x,y),y)$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) A^{\Delta}(n,m-k_b,k_a)$
$G(x,y) = H(y,B(x,y))$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) B^{\Delta}(n,m-k_a,k_b)$
$G(x,y) = H(A(x,y),x)$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) A^{\Delta}(n-k_b,m,k_a)$
$G(x,y) = H(x,B(x,y))$	$g(n,m) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} h(k_a,k_b) B^{\Delta}(n-k_a,m,k_b)$
$G(x,y) = H(A(x),B(y))$	$g(n,m) = \sum_{k_a=0}^{n} \sum_{k_b=0}^{m} h(k_a,k_b) A^{\Delta}(n,k_a) B^{\Delta}(m,k_b)$
$G(x,y) = H(A(x),y)$	$g(n,m) = \sum_{k=0}^{n} h(k,m) A^{\Delta}(n,k)$
$G(x,y) = H(x,B(y))$	$g(n,m) = \sum_{k=0}^{m} h(n,k) B^{\Delta}(m,k)$
$G(x,y) = H(A(x,y))$	$g(n,m) = \sum_{k=0}^{n+m} h(k) A^{\Delta}(n,m,k)$

Далее рассмотрим процесс нахождения коэффициентов k-й степени композиции двумерных производящих функций путем обобщения двумерной производящей функции $H(x,y)$ из Теоремы 5 до степени k.

Теорема 6. *Пусть*

$$H(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} h(n,m,k)x^n y^m,$$

$$A(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} A^{\Delta}(n,m,k)x^n y^m, \quad ord(A)\geqslant 1,$$

$$B(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} B^{\Delta}(n,m,k)x^n y^m, \quad ord(B)\geqslant 1.$$

Тогда коэффициенты k-й степени композиции двумерных производящих функций

$$G(x,y)^k = H(A(x,y),B(x,y))^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m,k)x^n y^m$$

будут равны

$$g(n,m,k) = \sum_{k_a=0}^{n+m}\sum_{k_b=0}^{n+m-k_a} h(k_a,k_b,k)\sum_{i=0}^{n}\sum_{j=0}^{m} A^{\Delta}(i,j,k_a)B^{\Delta}(n-i,m-j,k_b). \quad (2.17)$$

Доказательство:

Доказательство строится аналогично доказательству для Теоремы 5. □

Следствие 8. *Пусть*

$$H(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} H^{\Delta}(n,m,k)x^n y^m, \quad ord(H)\geqslant 1,$$

$$A(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} A^{\Delta}(n,m,k)x^n y^m, \quad ord(A)\geqslant 1,$$

$$B(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} B^{\Delta}(n,m,k)x^n y^m, \quad ord(B)\geqslant 1.$$

Тогда композита композиции двумерных производящих функций

$$G(x,y) = H(A(x,y),B(x,y)) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m$$

будет равна

$$G^{\Delta}(n,m,k) = \quad (2.18)$$

$$= \sum_{k_a=0}^{n+m}\sum_{k_b=0}^{n+m-k_a} H^{\Delta}(k_a,k_b,k)\sum_{i=0}^{n}\sum_{j=0}^{m} A^{\Delta}(i,j,k_a)B^{\Delta}(n-i,m-j,k_b).$$

Следствие 8 может быть применено для вычисления композит за счет декомпозиции на более простые производящие функции.

В таблице 2.2 представлены полученные результаты для частных случаев использования результатов Теоремы 6.

Таблица 2.2 — Частные случаи Теоремы 6

Композиция	Явная формула
$G(x,y)^k = H(A(x,y),B(y))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) \sum\limits_{j=0}^{m} A^{\Delta}(n,j,k_a) B^{\Delta}(m-j,k_b)$
$G(x,y)^k = H(A(y),B(x,y))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) \sum\limits_{j=0}^{m} A^{\Delta}(j,k_a) B^{\Delta}(n,m-j,k_b)$
$G(x,y)^k = H(A(x,y),B(x))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) \sum\limits_{i=0}^{n} A^{\Delta}(i,m,k_a) B^{\Delta}(n-i,k_b)$
$G(x,y)^k = H(A(x),B(x,y))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) \sum\limits_{i=0}^{n} A^{\Delta}(i,k_a) B^{\Delta}(n-i,m,k_b)$
$G(x,y)^k = H(A(x,y),y)^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) A^{\Delta}(n,m-k_b,k_a)$
$G(x,y)^k = H(y,B(x,y))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) B^{\Delta}(n,m-k_a,k_b)$
$G(x,y)^k = H(A(x,y),x)^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) A^{\Delta}(n-k_b,m,k_a)$
$G(x,y)^k = H(x,B(x,y))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} \sum\limits_{k_b=0}^{n+m-k_a} h(k_a,k_b,k) B^{\Delta}(n-k_a,m,k_b)$
$G(x,y)^k = H(A(x),B(y))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n} \sum\limits_{k_b=0}^{m} h(k_a,k_b,k) A^{\Delta}(n,k_a) B^{\Delta}(m,k_b)$
$G(x,y)^k = H(A(x),y)^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n} h(k_a,m,k) A^{\Delta}(n,k_a)$
$G(x,y)^k = H(x,B(y))^k$	$g(n,m,k) = \sum\limits_{k_b=0}^{m} h(n,k_b,k) B^{\Delta}(m,k_b)$
$G(x,y)^k = H(A(x,y))^k$	$g(n,m,k) = \sum\limits_{k_a=0}^{n+m} h(k_a,k) A^{\Delta}(n,m,k_a)$

2.2.3 Сложение двумерных производящих функций

Рассмотрим задачу получения явных выражений для композит суммы двух двумерных производящих функций.

Теорема 7. *Пусть*

$$A(x,y)^k = \sum_{n \geqslant 0} \sum_{m \geqslant 0} A^{\Delta}(n,m,k) x^n y^m, \quad ord(A) \geqslant 1,$$

$$B(x,y)^k = \sum_{n \geqslant 0} \sum_{m \geqslant 0} B^{\Delta}(n,m,k) x^n y^m, \quad ord(B) \geqslant 1.$$

Тогда композита суммы двух двумерных производящих функций

$$G(x,y) = A(x,y) + B(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m$$

будет равна

$$G^{\Delta}(n,m,k) = \sum_{k_a=0}^{n+m}\binom{k}{k_a}\sum_{i=0}^{n}\sum_{j=0}^{m} A^{\Delta}(i,j,k_a)B^{\Delta}(n-i,m-j,k-k_a). \qquad (2.19)$$

Доказательство:

Рассмотрим следующую двумерную производящую функцию

$$H(x,y) = x + y$$

и ее k-ю степень

$$H(x,y)^k = (x+y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} H^{\Delta}(n,m,k)x^n y^m = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\binom{k}{n}\delta(m,k-n)x^n y^m.$$

Применяя (2.18) для композиции производящих функций

$$G(x,y) = A(x,y) + B(x,y) = H(A(x,y),B(x,y)),$$

получаем

$$G^{\Delta}(n,m,k) = \sum_{k_a=0}^{n+m}\sum_{k_b=0}^{n+m-k_a}\binom{k}{k_a}\delta(k_b,k-k_a)\sum_{i=0}^{n}\sum_{j=0}^{m} A^{\Delta}(i,j,k_a)B^{\Delta}(n-i,m-j,k_b).$$

Используя свойства символа Кронекера, получаем $k_b = k - k_a$.

Упрощая формулу для $G^{\Delta}(n,m,k)$, получим искомую формулу (2.19). □

2.2.4 Умножение двумерных производящих функций

Рассмотрим задачу получения явных выражений для композит произведения двух двумерных производящих функций.

Теорема 8. *Пусть*

$$A(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} A^{\Delta}(n,m,k)x^n y^m, \quad ord(A) \geqslant 1,$$

$$B(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} B^{\Delta}(n,m,k)x^n y^m, \quad ord(B) \geqslant 1.$$

Тогда композита произведения двух двумерных производящих функций

$$G(x,y) = A(x,y) \cdot B(x,y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} g(n,m) x^n y^m$$

будет равна

$$G^{\Delta}(n,m,k) = \sum_{i=0}^{n} \sum_{j=0}^{m} A^{\Delta}(i,j,k) B^{\Delta}(n-i,m-j,k). \tag{2.20}$$

Доказательство:

Рассмотрим двумерную производящую функцию

$$H(x,y) = xy$$

и ее k-ю степень

$$H(x,y)^k = (xy)^k = \sum_{n\geqslant 0} \sum_{m\geqslant 0} H^{\Delta}(n,m,k) x^n y^m = \sum_{n\geqslant 0} \sum_{m\geqslant 0} \delta(n,k)\delta(m,k) x^n y^m.$$

Применяя (2.18) для композиции производящих функций

$$G(x,y) = A(x,y) \cdot B(x,y) = H(A(x,y),B(x,y)),$$

получаем

$$G^{\Delta}(n,m,k) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} \delta(k_a,k)\delta(k_b,k) \sum_{i=0}^{n} \sum_{j=0}^{m} A^{\Delta}(i,j,k_a) B^{\Delta}(n-i,m-j,k_b).$$

Используя свойства символа Кронекера, получаем $k_a = k$, $k_b = k$.

Упрощая формулу для $G^{\Delta}(n,m,k)$, получим искомую формулу (2.20). □

2.2.5 Обращение двумерных производящих функций

По аналогии с одномерным случаем, дадим определения взаимной и обратной двумерной производящей функции.

Определение 20. *Взаимной производящей функцией $G(x,y)$ для двумерной производящей функции*

$$F(x,y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} f(n,m) x^n y^m$$

является формальный степенной ряд, удовлетворяющий условию

$$F(x,y) \cdot G(x,y) = 1.$$

Определение 21. *Обратной производящей функцией* $\overline{F}(x,y)$ *для двумерной производящей функции*

$$F(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} f(n,m)x^n y^m$$

относительно формальной переменной x *является формальный степенной ряд, удовлетворяющий условию*

$$F(\overline{F}(x,y),y) = x.$$

Теорема 9. *Пусть*

$$F(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} f(n,m)x^n y^m, \quad f(0,0)\neq 0, \quad F(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} f(n,m,k)x^n y^m,$$

$$G(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m, \quad g(0,0)\neq 0, \quad G(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m,k)x^n y^m,$$

$$F(x,y)\cdot G(x,y) = 1.$$

Тогда коэффициенты $g(n,m,k)$ *будут равны*

$$g(n,m,k) = \sum_{i=0}^{n+m} \binom{n+m+k}{i+k}\binom{i+k-1}{i}\frac{f(n,m,i)}{f(0,0)^{i+k}}(-1)^i. \tag{2.21}$$

Доказательство:

Рассмотрим k-ю степень производящей функции $G(x,y)$ как композицию следующих производящих функций:

$$G(x,y)^k = \left(\frac{1}{F(x,y)}\right)^k = \frac{1}{f(0,0)^k}\left(\frac{1}{1+\left(\frac{F(x,y)}{f(0,0)}-1\right)}\right)^k = \frac{1}{f(0,0)^k}H(A(x,y))^k,$$

где

$$H(x) = \frac{1}{1+x}, \quad H(x)^k = \sum_{n\geqslant 0} h(n,k)x^n,$$

$$A(x,y) = \frac{F(x,y)}{f(0,0)} - 1, \quad A(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} A^{\Delta}(n,m,k)x^n y^m.$$

Коэффициенты $h(n,k)$ могут быть вычислены как

$$h(n,k) = (-1)^n\binom{n+k-1}{n}.$$

На основе биномиальных коэффициентов, композита $A^{\Delta}(n,m,k)$ равна

$$A^{\Delta}(n,m,k) = \sum_{i=0}^{k} \binom{k}{i} \frac{f(n,m,i)}{f(0,0)^i} (-1)^{k-i}.$$

Применяя (2.17) для $G(x,y)$, получим

$$g(n,m,k) = \frac{1}{f(0,0)^k} \sum_{k_a=0}^{n+m} h(k_a,k) A^{\Delta}(n,m,k_a) =$$

$$= \sum_{k_a=0}^{n+m} \sum_{i=0}^{k_a} \binom{k_a+k-1}{k_a} \binom{k_a}{i} \frac{f(n,m,i)}{f(0,0)^{i+k}} (-1)^i.$$

Тогда, после изменения порядка суммирования, получим

$$g(n,m,k) = \sum_{i=0}^{n+m} \sum_{k_a=0}^{n+m-i} \binom{k_a+i+k-1}{k_a+i} \binom{k_a+i}{i} \frac{f(n,m,i)}{f(0,0)^{i+k}} (-1)^i.$$

Далее избавимся от k_a во втором биномиальном коэффициенте:

$$g(n,m,k) = \sum_{i=0}^{n+m} \sum_{k_a=0}^{n+m-i} \binom{k_a+i+k-1}{k_a} \binom{i+k-1}{i} \frac{f(n,m,i)}{f(0,0)^{i+k}} (-1)^i.$$

Используем тождество (1.49) из [91]:

$$\sum_{k_a=0}^{n+m-i} \binom{k_a+i+k-1}{k_a} = \binom{n+m+k}{n+m-i}.$$

В результате получаем искомую формулу (2.21). □

Теорема 10. *Пусть*

$$F(x,y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} f(n,m) x^n y^m, \quad f(0,0) = 0, \quad F(x,y)^k = \sum_{n\geqslant 0} \sum_{m\geqslant 0} F^{\Delta}(n,m,k) x^n y^m,$$

$$\overline{F}(x,y) = \sum_{n\geqslant 0} \sum_{m\geqslant 0} \overline{f}(n,m) x^n y^m, \quad \overline{f}(0,0) = 0, \quad \overline{F}(x,y)^k = \sum_{n\geqslant 0} \sum_{m\geqslant 0} \overline{F}^{\Delta}(n,m,k) x^n y^m,$$

$$F(\overline{F}(x,y),y) = x.$$

Тогда композита $\overline{F}^{\Delta}(n,m,k)$ *будет равна*

$$\overline{F}^{\Delta}(n,m,k) = \frac{k}{n} \sum_{i=0}^{n+m} \binom{2n+m-k}{i+n} \binom{i+n-1}{i} \frac{F^{\Delta}(i+n-k,m,i)}{f(1,0)^{i+n}} (-1)^i. \quad (2.22)$$

Доказательство:

Используя формулу обращения Лагранжа для функционального уравнения

$$\overline{F}(x,y) = xG(\overline{F}(x,y),y),$$

где

$$G(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m, \quad g(0,0) \neq 0, \quad G(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m,k)x^n y^m,$$

получим

$$\overline{F}^{\Delta}(n,m,k) = \frac{k}{n} g(n-k,m,n). \tag{2.23}$$

Представим полученное уравнение следующим образом:

$$x = \frac{\overline{F}(x,y)}{G(\overline{F}(x,y),y)} = F(\overline{F}(x,y),y),$$

где

$$F(x,y) = \frac{x}{G(x,y)},$$

Тогда, применив (2.21) для $G(x,y)$, получим коэффициенты для k-й степени производящей функции $G(x,y)$:

$$g(n,m,k) = \sum_{i=0}^{n+m} \binom{n+m+k}{i+k} \binom{i+k-1}{i} \frac{F^{\Delta}(i+n,m,i)}{f(1,0)^{i+k}} (-1)^i. \tag{2.24}$$

Используя (2.23) и (2.24), получим искомую формулу (2.22). □

Далее рассмотрим обратную задачу: пусть в уравнении Лагранжа вида

$$A(x,y) = xG(A(x,y),y) \tag{2.25}$$

известна производящая функция $A(x,y)$ и ее композита $A^{\Delta}(n,m,k)$, тогда необходимо найти коэффициенты k-й степени производящей функции $G(x,y)$. Далее композиту производящей функции $xG(x,y)$ будем называть левой композитой относительно производящей функции $A(x,y)$ в уравнении (2.25).

Поскольку $g(0,0) \neq 0$ и $\frac{1}{g(0,0)} \neq 0$, то уравнение (2.25) представим следующим образом:

$$x = \frac{A(x,y)}{x\,G(A(x,y),y)} = B(A(x,y),y),$$

где

$$B(x,y) = \frac{x}{G(x,y)}. \tag{2.26}$$

Получаем, что $A(x,y)$ является обратной производящей функцией для $B(x,y)$ относительно формальной переменной x.

Если для уравнения

$$B(A(x,y),y) = x,$$

рассмотреть переменную y как некоторый фиксируемый параметр, тогда выполняется и обратное соотношение

$$A(B(x,y),y) = x,$$

то есть $B(x,y)$ является обратной производящей функцией для $A(x,y)$ относительно формальной переменной x.

Поскольку $B(x,y)$ является обратной производящей функцией для $A(x,y)$ относительно формальной переменной x, то на основе Теоремы 10 получаем следующее выражение для ее композиты:

$$B^{\Delta}(n,m,k) = \frac{k}{n}\sum_{i=0}^{n+m}\binom{2n+m-k}{i+n}\binom{i+n-1}{i}\frac{A^{\Delta}(i+n-k,m,i)}{a(1,0)^{i+n}}(-1)^i. \quad (2.27)$$

Учитывая вид выражения (2.26) и применяя к нему Теорему 9, получаем искомые коэффициенты для производящей функции $G(x,y)^k$

$$g(n,m,k) = \sum_{i=0}^{n+m}\binom{n+m}{i+k}\binom{i+k-1}{i}\frac{B^{\Delta}(n-k,m,i)}{b(0,0)^{i+k}}(-1)^i. \quad (2.28)$$

На основании формул (2.27) и (2.28) можно предложить следующую схему вычисления композит производящих функций, участвующих в уравнении (2.25)

$$\begin{array}{ccc} G(x,y) & \longrightarrow & A(x,y) \\ \updownarrow & & \updownarrow \\ \dfrac{x}{G(x,y)} & \longleftarrow & \dfrac{x}{A(x,y)} \end{array}$$

Таким образом, у каждой композиты $A^{\Delta}(n,m,k)$, где $A^{\Delta}(1,1,1) \neq 0$, имеется левая композита.

Обобщая и рассматривая уравнение (2.25), через замену

$$A(x,y) = x\, B_1(x,y)$$

получим

$$B_1(x,y) = G(x\, B_1(x,y),y) \tag{2.29}$$

С другой стороны, рассмотрим уравнение

$$B_2(x,y) = B_1(x\, B_2(x,y),y)$$

и подставим вместо $B_1(x,y)$ правую часть уравнения (2.29):

$$B_2(x,y) = G(x\, B_2(x,y)\, B_1(x\, B_2(x,y),y) = G(x\, B_2(x,y)^2,y).$$

По индукции получим

$$B_r(x,y) = G(x\, B_r(x,y)^r,y),$$

где $r \in \mathbb{N}$.

Тогда для функции $B_r(x,y)$ коэффициенты ее k-й степени имеют вид

$$b_r(n,m,k) = \frac{k}{r\,n+k} g(n,m,r\,n+k).$$

Таким образом, формируется следующая схема отношений производящих функций:

$$\begin{array}{ccccccccccc}
\ldots & \longrightarrow & B_{-1}(x,y) & \longrightarrow & G(x,y) & \longrightarrow & B_1(x,y) & \longrightarrow & B_2(x,y) & \longrightarrow & \ldots \\
 & & \updownarrow & & \updownarrow & & \updownarrow & & \updownarrow & & \\
\ldots & \longleftarrow & \dfrac{x}{B_{-1}(x,y)} & \longleftarrow & \dfrac{x}{G(x,y)} & \longleftarrow & \dfrac{x}{B_1(x,y)} & \longleftarrow & \dfrac{x}{B_2(x,y)} & \longleftarrow & \ldots
\end{array}$$

2.3 Методы получения явных выражений коэффициентов степеней производящих функций для случая трех переменных

Далее представлено обобщение разработанных методов на случай трехмерных производящих функций.

2.3.1 Композита трехмерной производящей функции

Определение 22. *Трехмерной производящей функцией $F(x,y,z)$ будем называть следующий формальный степенной ряд:*

$$F(x,y,z) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} f(n,m,l)x^n y^m z^l.$$

Тогда k-я степень трехмерной производящей функции $F(x,y,z)$, для которой $ord(F) \geqslant 1$, определяется как

$$F(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} F^{\Delta}(n,m,l,k)x^n y^m z^l.$$

где $F^{\Delta}(n,m,l,k)$ — композита трехмерной производящей функции.

Далее введены и подробно рассмотрены основные операции для работы с композитами трехмерных производящих функций.

2.3.2 Композиция трехмерных производящих функций

Рассмотрим задачу получения явных выражений коэффициентов композиции производящих функций трех переменных.

Теорема 11. *Пусть*

$$H(x,y,z) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} h(n,m,l)\, x^n y^m z^l,$$

$$A(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} A^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(A) \geqslant 1,$$

$$B(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} B^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(B) \geqslant 1,$$

$$C(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} C^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(C) \geqslant 1.$$

Тогда коэффициенты композиции трехмерных производящих функций

$$G(x,y,z) = H(A(x,y,z),B(x,y,z),C(x,y,z)) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} g(n,m,l)\, x^n y^m z^l$$

будут равны

$$g(n,m,l) = \sum_{k_a=0}^{n+m+l}\ \sum_{k_b=0}^{n+m+l-k_a}\ \sum_{k_c=0}^{n+m+l-k_a-k_b} h(k_a,k_b,k_c)\cdot h_{ABC}(n,m,l,k_a,k_b,k_c),$$

где

$$h_{ABC}(n,m,l,k_a,k_b,k_c) = \tag{2.30}$$

$$= \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l} h_{AB}(n_i,m_i,l_i,k_a,k_b)\cdot C^{\Delta}(n-n_i,m-m_i,l-l_i,k_c),$$

$$h_{AB}(n,m,l,k_a,k_b) = \tag{2.31}$$

$$= \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l} A^{\Delta}(n_i,m_i,l_i,k_a)\cdot B^{\Delta}(n-n_i,m-m_i,l-l_i,k_b).$$

Доказательство:

Доказательство строится аналогично доказательству для Теоремы 5. □

Следствие 9. *Пусть*

$$H(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} H^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(H)\geqslant 1,$$

$$A(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} A^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(A)\geqslant 1,$$

$$B(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} B^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(B)\geqslant 1,$$

$$C(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} C^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(C)\geqslant 1.$$

Тогда композита композиции трехмерных производящих функций

$$G(x,y,z) = H(A(x,y,z),B(x,y,z),C(x,y,z)) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} g(n,m,l)\, x^n y^m z^l$$

будет равна

$$G^{\Delta}(n,m,l,k) = \sum_{k_a=0}^{n+m+l}\ \sum_{k_b=0}^{n+m+l-k_a}\ \sum_{k_c=0}^{n+m+l-k_a-k_b} H^{\Delta}(k_a,k_b,k_c,k)\cdot h_{ABC}(n,m,l,k_a,k_b,k_c).$$

2.3.3 Сложение трехмерных производящих функций

Рассмотрим задачу получения явных выражений для композит суммы двух трехмерных производящих функций.

Теорема 12. *Пусть*

$$A(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} A^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(A) \geqslant 1,$$

$$B(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} B^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(B) \geqslant 1.$$

Тогда композита суммы двух трехмерных производящих функций

$$G(x,y,z) = A(x,y,z) + B(x,y,z) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} g(n,m,l)\, x^n y^m z^l$$

будет равна

$$G^{\Delta}(n,m,l,k) = \tag{2.32}$$

$$= \sum_{k_a=0}^{n+m+l} \binom{k}{k_a} \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l} A^{\Delta}(n_i,m_i,l_i,k_a)\cdot B^{\Delta}(n-n_i,m-m_i,l-l_i,k-k_a).$$

Доказательство:

Рассмотрим двумерную производящую функцию

$$H(x,y) = x + y$$

и ее k-ю степень

$$H(x,y)^k = (x+y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0} H^{\Delta}(n,m,k)\, x^n y^m = \sum_{n\geqslant 0}\sum_{m\geqslant 0} \binom{k}{n} \delta(m,k-n)\, x^n y^m.$$

Следовательно, композита производящей функции $H(x,y)$ равна

$$H^{\Delta}(n,m,k) = \binom{k}{n}\, \delta(m,k-n).$$

Далее представим производящую функцию $H(x,y)$ в виде следующей трехмерной производящей функции:

$$H_{xy}(x,y,z) = H(x,y)\, z^0 = H(x,y).$$

Тогда композита производящей функции $H_{xy}(x,y,z)$ примет вид

$$H^{\Delta}_{xy}(n,m,l,k) = H^{\Delta}(n,m,k)\,\delta(l,0) = \binom{k}{n}\,\delta(m,k-n)\,\delta(l,0).$$

Также рассмотрим трехмерную производящую функцию

$$C(x,y,z) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} c(n,m,l)\,x^n y^m z^l, \quad ord(B)\geqslant 1,$$

$$C(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} C^{\Delta}(n,m,l,k)\,x^n y^m z^l.$$

Применяя Теорему 11 для композиции производящих функций

$$G(x,y,z) = A(x,y,z) + B(x,y,z) = H_{xy}(A(x,y,z),B(x,y,z),C(x,y,z)),$$

получаем

$$G^{\Delta}(n,m,l,k) =$$

$$= \sum_{k_a=0}^{n+m+l}\ \sum_{k_b=0}^{n+m+l-k_a}\ \sum_{k_c=0}^{n+m+l-k_a-k_b} H^{\Delta}_{xy}(k_a,k_b,k_c,k)\cdot h_{ABC}(n,m,l,k_a,k_b,k_c) =$$

$$= \sum_{k_a=0}^{n+m+l}\ \sum_{k_b=0}^{n+m+l-k_a}\ \sum_{k_c=0}^{n+m+l-k_a-k_b} \binom{k}{k_a}\,\delta(k_b,k-k_a)\,\delta(k_c,0)\,h_{ABC}(n,m,l,k_a,k_b,k_c).$$

где $h_{ABC}(n,m,l,k_a,k_b,k_c)$ являются коэффициентами производящей функции

$$H_{ABC}(x,y,z) = A(x,y,z)^{k_a}\cdot B(x,y,z)^{k_b}\cdot C(x,y,z)^{k_c}.$$

Используя свойства символа Кронекера, получаем $k_c = 0$ и $k_b = k - k_a$. Упрощая формулу для $G^{\Delta}(n,m,l,k)$, получим

$$G^{\Delta}(n,m,l,k) = \sum_{k_a=0}^{n+m+l} \binom{k}{k_a}\,h_{ABC}(n,m,l,k_a,k-k_a,0).$$

Применяя (2.30) для $h_{ABC}(n,m,l,k_a,k-k_a,0)$, имеем

$$h_{ABC}(n,m,l,k_a,k-k_a,0) =$$

$$= \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l} h_{AB}(n_i,m_i,l_i,k_a,k-k_a)\cdot C^{\Delta}(n-n_i,m-m_i,l-l_i,0),$$

где $h_{AB}(n,m,l,k_a,k_b)$ являются коэффициентами производящей функции

$$H_{AB}(x,y,z) = A(x,y,z)^{k_a} \cdot B(x,y,z)^{k_b}.$$

Согласно свойству нулевой степени композиты $C^{\Delta}(n-n_i,m-m_i,l-l_i,0)$ равно 1 только при $n-n_i=0$, $m-m_i=0$ и $l-l_i=0$, иначе оно равно 0. Следовательно, получаем $n_i=n$, $m_i=m$ и $l_i=l$. Тогда

$$h_{ABC}(n,m,l,k_a,k-k_a,0) = h_{AB}(n,m,l,k_a,k-k_a).$$

Применяя (2.31) для $h_{AB}(n,m,l,k_a,k-k_a)$, имеем

$$h_{AB}(n,m,l,k_a,k-k_a) =$$

$$= \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l} A^{\Delta}(n_i,m_i,l_i,k_a)\cdot B^{\Delta}(n-n_i,m-m_i,l-l_i,k-k_a).$$

Объединяя полученные результаты, получим искомую формулу (2.32). □

2.3.4 Умножение трехмерных производящих функций

Рассмотрим задачу получения явных выражений для композит произведения двух трехмерных производящих функций.

Теорема 13. *Пусть*

$$A(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} A^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(A)\geqslant 1,$$

$$B(x,y,z)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} B^{\Delta}(n,m,l,k)\, x^n y^m z^l, \quad ord(B)\geqslant 1.$$

Тогда композита произведения двух трехмерных производящих функций

$$G(x,y,z) = A(x,y,z)\cdot B(x,y,z) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} g(n,m,l)\, x^n y^m z^l$$

будет равна

$$G^{\Delta}(n,m,l,k) = \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l} A^{\Delta}(n_i,m_i,l_i,k)\cdot B^{\Delta}(n-n_i,m-m_i,l-l_i,k). \quad (2.33)$$

Доказательство:

Для получения $G^{\Delta}(n,m,l,k)$ воспользуемся результатом Теоремы 11 для композиции производящих функций следующего вида:

$$G(x,y,z) = A(x,y,z) \cdot B(x,y,z) = H(A(x,y,z),B(x,y,z)),$$

где $H(x,y) = xy$.

В результате получим искомую формулу (2.33). □

2.4 Метод получения явных выражений коэффициентов рациональных производящих функций для случая n переменных

По такому же пути правила вычисления композит могут быть расширены на производящие функции для случая n переменных.

Пусть дана рациональная производящая функция многих переменных следующего вида:

$$F(x_1,x_2,\ldots,x_n) =$$

$$= \sum_{k_1 \geqslant 0} \sum_{k_2 \geqslant 0} \cdots \sum_{k_n \geqslant 0} f(k_1,k_2,\ldots,k_n)\, x_1^{k_1} x_2^{k_2} \cdots x_n^{k_n} = \frac{P(x_1,x_2,\ldots,x_n)}{Q(x_1,x_2,\ldots,x_n)},$$

где $P(x_1,x_2,\ldots,x_n)$ и $Q(x_1,x_2,\ldots,x_n)$ — производящие функции n переменных

$$P(x_1,x_2,\ldots,x_n) = \sum_{k_1 \geqslant 0} \sum_{k_2 \geqslant 0} \cdots \sum_{k_n \geqslant 0} p(k_1,k_2,\ldots,k_n)\, x_1^{k_1} x_2^{k_2} \cdots x_n^{k_n},$$

$$Q(x_1,x_2,\ldots,x_n) = \sum_{k_1 \geqslant 0} \sum_{k_2 \geqslant 0} \cdots \sum_{k_n \geqslant 0} q(k_1,k_2,\ldots,k_n)\, x_1^{k_1} x_2^{k_2} \cdots x_n^{k_n}.$$

Тогда для получения явных выражений коэффициентов функций данного вида можно воспользоваться следующим методом:

1. Представить производящую функцию $P(x_1,x_2,\ldots,x_n)$ как сумму производящих функций $P_i(x_1,x_2,\ldots,x_n)$, для которых можно получить их коэффициенты $p_i(k_1,k_2,\ldots,k_n)$ и композиты $P_i^{\Delta}(k_1,k_2,\ldots,k_n,k)$:

$$P(x_1,x_2,\ldots,x_n) = \sum_i P_i(x_1,x_2,\ldots,x_n);$$

2. Найти явные выражения коэффициентов $p_i(k_1,k_2,\ldots,k_n)$ и композит $P_i^{\Delta}(k_1,k_2,\ldots,k_n,k)$ для всех полученных производящих функций $P_i(x_1,x_2,\ldots,x_n)$;

3. Представить производящую функцию $Q(x_1,x_2,\ldots,x_n)$ как сумму производящих функций $Q_i(x_1,x_2,\ldots,x_n)$, для которых можно получить их коэффициенты $q_i(k_1,k_2,\ldots,k_n)$ и композиты $Q_i^{\Delta}(k_1,k_2,\ldots,k_n,k)$:

$$Q(x_1,x_2,\ldots,x_n) = 1 - \sum_i Q_i(x_1,x_2,\ldots,x_n);$$

4. Найти явные выражения коэффициентов $q_i(k_1,k_2,\ldots,k_n)$ и композит $Q_i^{\Delta}(k_1,k_2,\ldots,k_n,k)$ для всех полученных производящих функций $Q_i(x_1,x_2,\ldots,x_n)$;

5. Последовательно выполняя операции композиции и сложения для каждой $Q_i(x_1,x_2,\ldots,x_n)$, получить композиту $Q_s^{\Delta}(k_1,k_2,\ldots,k_n,k)$ производящей функции

$$1 - Q(x_1,x_2,\ldots,x_n) = \sum_i Q_i(x_1,x_2,\ldots,x_n);$$

6. Получить коэффициенты производящей функции

$$R(x_1,x_2,\ldots,x_n) = \frac{1}{Q(x_1,x_2,\ldots,x_n)} = \frac{1}{1 - \sum\limits_i Q_i(x_1,x_2,\ldots,x_n)}$$

по формуле

$$r(k_1,k_2,\ldots,k_n) = \sum_{k=0}^{k_1+k_2+\ldots+k_n} Q_s^{\Delta}(k_1,k_2,\ldots,k_n,k); \qquad (2.34)$$

7. Получить коэффициенты производящей функции

$$F(x_1,x_2,\ldots,x_n) = \frac{P(x_1,x_2,\ldots,x_n)}{Q(x_1,x_2,\ldots,x_n)} = P(x_1,x_2,\ldots,x_n) \cdot R(x_1,x_2,\ldots,x_n),$$

применив операцию умножения производящих функций

$$f(k_1,k_2,\ldots,k_n) = \sum_{l_1 \geqslant 0} \sum_{l_2 \geqslant 0} \ldots \sum_{l_n \geqslant 0} p(l_1,l_2,\ldots,l_n)\, r(k_1 - l_1, k_2 - l_2, \ldots, k_n - l_n).$$

По итогу выполнения всех требуемых действий будет получено явное выражение коэффициентов рациональных производящих функций многих переменных.

2.5 Методы формирования числовых треугольников на основе производящих функций и их приложения

Рассмотрим в качестве информационного объекта числовые треугольники. Числовые треугольники играют важную роль в комбинаторике, теории чисел и других областях, в том числе связанных с теоретической информатикой. Изучению числовых треугольников были посвящены работы таких ученых, как: Б. Паскаль, Л. Эйлер, Д. Бернулли, Д. Риордан, Д.Э. Кнут, Л.У. Шапиро, О.В. Кузьмин, П. Берри и другие. Существует большое разнообразие числовых треугольников, например, треугольник Паскаля, Бернулли-Эйлера, Каталана, Моцкина.

В общем виде числовые треугольники могут быть построены:

1. Явной формулой;
2. Рекуррентным выражением;
3. Производящей функцией;
4. Массивом Риордана.

Исследования числовых треугольников, заданных массивами Риордана, показали, что в основе построения лежит степень производящей функции. Поэтому предложено строить такие треугольники за счет соответствующих коэффициентов степеней производящих функций.

В данном разделе получены: метод получения производящих функций для центральных коэффициентов числовых треугольников, метод получения треугольника по производящей функции его центральных коэффициентов и метод получения производящей функции диагонали $T_{2n,n}$.

2.5.1 Метод получения производящих функций для центральных коэффициентов числовых треугольников

В табличном виде композита представляется числовым треугольником:

$$
\begin{array}{ccccccccc}
 & & & & G^{\Delta}(1,1) & & & & \\
 & & & G^{\Delta}(2,1) & & G^{\Delta}(2,2) & & & \\
 & & G^{\Delta}(3,1) & & G^{\Delta}(3,2) & & G^{\Delta}(3,3) & & \\
 & \iddots & & \vdots & & \vdots & & \ddots & \\
G^{\Delta}(n,1) & & G^{\Delta}(n,2) & & \dots & & G^{\Delta}(n,n-1) & & G^{\Delta}(n,n)
\end{array}
$$

При рассмотрении этого треугольника видно, что центральные коэффициенты представлены следующей последовательностью:

$$G^{\Delta}(1,1),\ G^{\Delta}(3,2),\ \ldots,\ G^{\Delta}(2n-1,n)\ \ldots$$

Запишем производящую функцию для центральных коэффициентов:

$$F(x) = G^{\Delta}(1,1) + G^{\Delta}(3,2)^{\Delta}x^1 + \ldots + G^{\Delta}(2n-1,n)^{\Delta}x^{n-1} + \ldots$$

Докажем теорему о производящей функции центральных коэффициентов.

Теорема 14. *Пусть заданы производящая функция* $H(x) = \sum\limits_{n\geqslant 0} h(n)\,x^n$, *где* $h(0) \neq 0$, *производящая функция* $G(x) = xH(x)$, *где* $G(x)^k = \sum\limits_{n\geqslant k} G^{\Delta}(n,k)\,x^n$, *и производящая функция* $A(x) = \sum\limits_{n>0} a(n)\,x^n$, *определяемая функциональным уравнением* $A(x) = xH(A(x))$. *Тогда производящая функция* $F(x)$, *порождающая центральные коэффициенты числового треугольника* $G^{\Delta}(n,k)$, *равна первой производной производящей функции* $A(x)$:

$$F(x) = A'(x) = \sum_{n\geqslant 1} G^{\Delta}(2n-1,n)x^{n-1}. \tag{2.35}$$

Доказательство:

Аналогично получению соотношения (2.8), получаем формулу

$$A^{\Delta}(n,k) = \frac{k}{n}G^{\Delta}(2n-k,n).$$

Тогда

$$A(x)^k = \sum_{n\geqslant k} A^{\Delta}(n,k)\,x^n = \sum_{n\geqslant k} \frac{k}{n}G^{\Delta}(2n-k,n)\,x^n$$

и

$$A(x) = \sum_{n\geqslant 1} \frac{1}{n}G^{\Delta}(2n-1,n)\,x^n.$$

Дифференцируя $A(x)$ относительно x, получим искомую формулу (2.35). □

В качестве приложения Теоремы 14 рассмотрим следующие примеры.

Пример 7. *Найдем производящую функцию для центральных биномиальных коэффициентов [92; 93] (последовательность A*000984 *в [94]):*

$$1, 2, 6, 20, 70, 252, 924, 3432, 12870, 48620, \ldots$$

Для этого запишем выражение

$$(xH(x))^k = \left(\frac{x}{1-x}\right)^k = \sum_{n\geqslant k} \binom{n-1}{k-1} x^n,$$

которое порождает треугольник Паскаля:

$$\begin{array}{ccccccccc} & & & & 1 & & & & \\ & & & 1 & & 1 & & & \\ & & 1 & & 2 & & 1 & & \\ & 1 & & 3 & & 3 & & 1 & \\ 1 & & 4 & & 6 & & 4 & & 1 \end{array}$$

Составим следующее функциональное уравнение:

$$A(x) = \frac{x}{1-A(x)}.$$

Решая его, получим

$$A(x) = \frac{1-\sqrt{1-4x}}{2}.$$

Дифференцируя $A(x)$, получим производящую функцию для центральных биномиальных коэффициентов:

$$A'(x) = \frac{1}{\sqrt{1-4x}}.$$

Пример 8. *Рассмотрим производящую функцию*

$$H(x) = \frac{1-x}{1-2x}.$$

Тогда для $G(x) = xH(x)$ получаем композиту

$$G^{\Delta}(n,k) = \sum_{i=0}^{n-k} 2^i (-1)^{n-k} \binom{k}{n-k-i} \binom{k+i+1}{k-1},$$

которая задает числовой треугольник (последовательность A105306 в [94]):

$$\begin{array}{ccccccccc} & & & & 1 & & & & \\ & & & 1 & & 1 & & & \\ & & 2 & & 2 & & 1 & & \\ & 4 & & 5 & & 3 & & 1 & \\ 8 & & 12 & & 9 & & 4 & & 1 \end{array}$$

Найдем производящую функцию для центральных коэффициентов данного числового треугольника. Для этого решим функциональное уравнение

$$A(x) = x\frac{1 - A(x)}{1 - 2A(x)}.$$

Откуда

$$A(x) = \frac{1 + x - \sqrt{1 - 6x + x^2}}{4}.$$

Дифференцируя $A(x)$, получим производящую функцию для центральных коэффициентов

$$A'(x) = \frac{1}{4} - \frac{x - 3}{4\sqrt{x^2 - 6\,x + 1}}.$$

Пример 9. *Рассмотрим производящую функцию*

$$H(x) = x\,\mathrm{ctg}(x).$$

Тогда для $G(x) = xH(x)$ получаем композиту

$$G^{\Delta}(n,k) = 2^{n-2\,k}\ (-1)^{\frac{n-k}{2}}\ \sum_{l=0}^{k} 2^l\ l!\ \binom{k}{l}\ \sum_{m=0}^{n-2\,k+l} \frac{m!\ \left[{l+m \atop l}\right]\ \left\{{n-2k+l \atop m}\right\}}{(l+m)!(n-2\,k+l)!},$$

которая задает числовой треугольник (последовательность A199542 в [94]):

$$\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 0 & & 1 & & & \\
 & & -\frac{1}{3} & & 0 & & 1 & & \\
 & 0 & & -\frac{2}{3} & & 0 & & 1 & \\
-\frac{1}{45} & & 0 & & -1 & & 0 & & 1
\end{array}$$

Найдем производящую функцию для центральных коэффициентов данного числового треугольника. Для этого решим функциональное уравнение

$$A(x) = xA(x)\,\mathrm{ctg}(A(x)).$$

Откуда

$$A(x) = \arctan(x).$$

Дифференцируя $A(x)$, получим производящую функцию для центральных коэффициентов

$$A'(x) = \frac{1}{1 + x^2}.$$

Теперь рассмотрим обратную задачу: известна производящая функция центральных коэффициентов и необходимо найти треугольник, имеющий заданные центральные коэффициенты. Для этого докажем теорему о существовании треугольника для данных центральных коэффициентов.

Теорема 15. *Для заданной производящей функции* $F(x) = \sum\limits_{n\geqslant 0} f(n)\, x^n$, $f(0) \neq 0$, *существует единственная производящая функция* $H(x) = \sum\limits_{n\geqslant 0} h(n)\, x^n$ *такая, что треугольник* $G^{\Delta}(n,k)$, *заданный выражением*

$$(xH(x))^k = \sum_{n\geqslant k} G^{\Delta}(n,k)\, x^n,$$

имеет центральные коэффициенты, равные $f(n)$.

Доказательство:

Для доказательства данной теоремы рассмотрим доказательство Теоремы 14 в обратном порядке.

Интегрируя $F(x)$ по x, получим производящую функцию $A(x)$

$$A(x) = \int F(x)dx = \sum_{n>0} f(n-1)\frac{x^n}{n},$$

где $a(0) = 0$.

Чтобы найти $H(x)$, решим следующее обратное функциональное уравнение:

$$A(x) = xH(A(x)).$$

Пусть $A(x) = t$, тогда $x = A^{[-1]}(t)$. Откуда

$$t = A^{[-1]}(t)H(t)$$

или

$$H(t) = \frac{t}{A^{[-1]}(t)}.$$

Решим данное уравнение с помощью композит. Производящая функция

$$B(x) = \frac{A^{[-1]}(t)}{t}$$

есть взаимная производящая функция для $H(t)$

$$H(t)\frac{A^{-1}(t)}{t} = 1.$$

Согласно теореме о взаимных композитах, композита производящей функции $tH(t)$ равна

$$G^{\Delta}(n,k) = \begin{cases} 1, & n = k; \\ \sum\limits_{m=1}^{n-k} (-1)^m \binom{k+m-1}{k-1} \sum\limits_{j=1}^{m} (-1)^j \binom{m}{j} B_t^{\Delta}(n-k+j,j), & n > k. \end{cases}$$

В нашем случае, композита $B_t^{\Delta}(n,k)$ равна композите производящей функции $A^{[-1]}(t)$. Для получения композиты производящей функции $A^{[-1]}(t)$ используем следующий алгоритм вычисления левой композиты:

1. Вычислить композиту $A^{\Delta}(n,k)$ производящей функции $A(x)$;

2. Используя формулу (2.3), вычислить взаимную композиту $A_R^{\Delta}(n,k)$ для композиты $A^{\Delta}(x)$;

3. Используя взаимную композиту $A_R^{\Delta}(n,k)$, вычислить композиту обратной производящей функции $A^{[-1]}(t)$ по формуле

$$A_{Inv}^{\Delta}(n,k) = \frac{k}{n} A_R^{\Delta}(2n-k,n).$$

Приведенный алгоритм всегда находит левую композиту. Откуда

$$H(x) = \sum_{n \geqslant 1} G^{\Delta}(n,1) x^{n-1}.$$

□

Данная теорема дает основание и алгоритм для вычисления треугольника, заданного некоторой производящей функцией $H(x)$, $h(0) \neq 0$:

$$(xH(x))^k = \sum_{n \geqslant k} T_{n,k}\, x^n,$$

если известны лишь его центральные коэффициенты.

Рассмотрим пример нахождения числового треугольника $T_{n,k}$, где центральными коэффициентами являются числа Каталана.

Пример 10. *Производящая функция для чисел Каталана задается известным выражением (последовательность A*000108 *в [94])*

$$F(x) = \frac{1 - \sqrt{1-4x}}{2x}.$$

Получим искомый треугольник и его выражение. Для этого необходимо найти интеграл от $F(x)$

$$\int \frac{1-\sqrt{1-4x}}{2x}dx = \log\left(\sqrt{1-4\,x}+1\right)-\sqrt{1-4\,x}.$$

Вот первые члены полученной производящей функции:

$$-1+\log 2+x+\frac{x^2}{2}+\frac{2\,x^3}{3}+\frac{5\,x^4}{4}+\frac{14\,x^5}{5}+7\,x^6+\frac{132\,x^7}{7}+\cdots$$

Поскольку $a(0)=0$*, имеем*

$$A(x)=1-\log(2)+\log\left(\sqrt{1-4\,x}+1\right)-\sqrt{1-4\,x}=$$

$$=\log\left(1-\frac{1-\sqrt{1-4\,x}}{2}\right)+2\left(\frac{1-\sqrt{1-4\,x}}{2}\right).$$

Найдем композиту производящей функции $A(x)$*.*

$$A(x)=\log(1-C(x))+2C(x),$$

где

$$C(x)=\frac{1-\sqrt{1-4\,x}}{2}.$$

Композита производящей функции $\log(1-x)+2x$ *равна*

$$\sum_{j=0}^{k}(-1)^{n-j}\;2^j\binom{k}{j}\frac{(k-j)!}{(n-j)!}\begin{bmatrix}n-j\\k-j\end{bmatrix}.$$

Тогда композита производящей функции $C(x)$ *равна*

$$\frac{k}{n}\binom{2\,n-k-1}{n-1}.$$

Откуда композита производящей функции $A(x)$ *равна произведению композит для* $C(x)$ *и* $\log(1-x)+2x$

$$A^{\Delta}(n,k)=\sum_{m=k}^{n}\frac{m}{n}\binom{2\,n-m-1}{n-1}\sum_{j=0}^{k}(-1)^{m-j}\;2^j\binom{k}{j}\frac{(k-j)!}{(m-j)!}\begin{bmatrix}m-j\\k-j\end{bmatrix}.$$

Используя (2.3), композита взаимной производящей функции для $A(x)$ *равна*

$$A_R^{\Delta}(n,k)=\begin{cases}1, & n=k;\\ \sum\limits_{m=1}^{n-k}\binom{k+m-1}{k-1}\;\sum\limits_{j=1}^{m}(-1)^j\binom{m}{j}A^{\Delta}(n-k+j,j), & n>k.\end{cases}$$

На основе взаимной композиты $A_R^{\Delta}(n,k)$ найдем композиту $A_{Inv}^{\Delta}(n,k)$ обратной производящей функции $A^{[-1]}(x)$. Откуда получим композиту взаимной производящей к обратной производящей функции

$$G^{\Delta}(n,k) = \begin{cases} 1, & n = k; \\ \sum\limits_{m=1}^{n-k} \binom{k+m-1}{k-1} \sum\limits_{j=1}^{m} (-1)^j \binom{m}{j} A_{Inv}^{\Delta}(n-k+j,j), & n > k. \end{cases}$$

Это и будет искомая композита, определяющая числовой треугольник, который имеет следующие начальные значения:

$$\begin{array}{ccccccccccccccc}
 & & & & & & & & 1 & & & & & & \\
 & & & & & & & \frac{1}{2} & & 1 & & & & & \\
 & & & & & & \frac{5}{12} & & 1 & & 1 & & & & \\
 & & & & & \frac{1}{2} & & \frac{13}{12} & & \frac{3}{2} & & 1 & & & \\
 & & & & \frac{551}{720} & & \frac{17}{12} & & 2 & & 2 & & 1 & & \\
 & & & \frac{11}{8} & & \frac{529}{240} & & \frac{23}{8} & & \frac{19}{6} & & \frac{5}{2} & & 1 & \\
 & & \frac{16657}{6048} & & \frac{2831}{720} & & \frac{1111}{240} & & 5 & & \frac{55}{12} & & 3 & & 1 \\
 & \frac{4289}{720} & & \frac{46999}{6048} & & \frac{1329}{160} & & \frac{6059}{720} & & \frac{95}{12} & & \frac{25}{4} & & \frac{7}{2} & & 1
\end{array}$$

2.5.2 Метод получения производящей функции для диагонали $T_{2n,n}$ в числовом треугольнике $T_{n,k}$

Рассмотрим числовой треугольник $T_{n,k}$, заданный k-й степенью производящей функции $G(x)$. Сформулируем теорему о диагонали $T_{2n,n}$ в данном числовом треугольнике.

Теорема 16. *Пусть k-я степень производящей функции $G(x) = \sum\limits_{n>0} g_n\, x^n$ задает числовой треугольник $T_{n,k}$:*

$$G(x)^k = \sum_{n \geqslant k} T_{n,k}\, x^n.$$

Тогда производящая функция $A(x) = \sum\limits_{n \geqslant 0} T_{2n,n}\, x^n$ для диагонали $T_{2\,n,n}$ равна

$$A(x) = \frac{x\, F'(x)}{F(x)},$$

где $F(x)$ является решением функционального уравнения $F(x) = x\, H(F(x))$ при

$$H(x) = \frac{G(x)}{x}.$$

Доказательство:

Рассмотрим следующий ряд Лорана:

$$\Phi(z) = \varphi\, z + \varphi_0 + \frac{\varphi_1}{z} + \cdots + \frac{\varphi_n}{z^n} + \cdots$$

Возведем данную производящую функцию в степень k

$$\Phi(z)^k = \Phi_k(z) + E_k(z),$$

где $\Phi_k(z)$ содержит все неотрицательные значения z и $E_k(z)$ содержит оставшиеся значения z. Согласно [95] $\Phi_k(z)$ есть полином Фабера.

Рассмотрим производящую функцию $G(z)$ в терминах $\Phi(z)$, то есть

$$G(z)^k = (z^2\,\Phi(1/z))^k = \sum_{n \geqslant k} T_{n,k}\, z^n.$$

Тогда

$$\Phi(z)^k = z^{2\,k} \sum_{n \geqslant k} T_{n,k}\, z^{-n}.$$

После преобразований полиномы Фабера будут равны

$$\Phi_n(z) = \sum_{k=0}^{n} T_{2\,n-k,n}\, z^k.$$

Для случая $z = 0$ имеем

$$\Phi_n(0) = T_{2\,n,n}. \tag{2.36}$$

Согласно [95; 96] производящая функция для полиномов Фабера равна

$$\frac{t\varphi'(t)}{\varphi(t) - z} = \sum_{n \geqslant 0} \Phi_n(z) t^{-n},$$

где $\varphi(t)$ есть обратная производящая функция относительно $\Phi(t)$.

Производящая функция для случая $z = 0$ равна

$$\frac{t\varphi'(t)}{\varphi(t)} = \sum_{n \geqslant 0} \Phi_n(0) t^{-n}.$$

Далее пусть $t = \frac{1}{x}$. С учетом того, что

$$(\varphi(1/x))' = \varphi'(1/x)\,(1/x)' = \frac{-\varphi'(1/x)}{x^2}$$

или

$$\varphi'(1/x) = -x^2\,(\varphi(1/x))',$$

получим производящую функцию для $\Phi_n(0)$

$$A(x) = -\frac{x(\varphi(1/x))'}{\varphi(1/x)} = \sum_{n\geqslant 0} \Phi_n(0) x^n. \tag{2.37}$$

Поскольку $\varphi(t)$ — обратная производящая функция относительно $\Phi(t)$, следующее тождество имеем место быть:

$$\Phi(\varphi(t)) = t.$$

Если подставить $1/x$ вместо t, получим следующее выражение:

$$\Phi(\varphi(1/x)) = 1/x.$$

Поскольку

$$\Phi(x) = x^2 G(1/x) = x\,H(1/x),$$

получим

$$\varphi(1/x) H(1/\varphi(1/x)) = \frac{1}{x}.$$

Тогда

$$\frac{1}{\varphi(1/x)} = x\,H(1/\varphi(1/x)).$$

Согласно теореме Лагранжа получим

$$\frac{1}{\varphi(1/x)} = F(x)$$

Откуда, согласно (2.36) и (2.37), получаем искомый результат

$$A(x) = -\frac{x(\frac{1}{F(x)})'}{\frac{1}{F(x)}} = \frac{x\,F'(x)}{F(x)} = \sum_{n\geqslant 0} T_{2\,n,n}\, x^n.$$

□

В качестве приложения Теоремы 16 рассмотрим следующие примеры.

Пример 11. *Рассмотрим треугольник Паскаля, который задается следующим образом:*

$$G(x)^k = \left(\frac{x}{1-x}\right)^k = \sum_{n\geqslant k} \binom{n-1}{n-k} x^n.$$

Решением функционального уравнения

$$F(x) = \frac{x}{1 - F(x)}$$

является производящая функция

$$F(x) = \frac{1 - \sqrt{1 - 4\,x}}{2}.$$

Откуда производящая функция для диагонали $T_{2\,n,n}$ с общим элементом $\binom{2\,n-1}{n}$ равна

$$A(x) = \frac{x\,F'(x)}{F(x)} = \frac{2\,x}{\left(1 - \sqrt{1 - 4\,x}\right)\,\sqrt{1 - 4\,x}}.$$

Пример 12. *Найдем производящую функцию для диагонали $T_{2\,n,n}$ в числовом треугольнике, определяемом k-й степенью производящей функции*

$$G(x) = x + x^2 + x^3.$$

Решая уравнение

$$F(x) = x(1 + F(x) + F(x)^2),$$

получим производящую функцию для чисел Моцкина (последовательность A001006 в [94])

$$F(x) = \frac{-\sqrt{-3\,x^2 - 2\,x + 1} - x + 1}{2\,x}.$$

Тогда

$$\frac{x\,F'(x)}{F(x)} = \frac{\sqrt{-3\,x^2 - 2\,x + 1} + x - 1}{(x - 1)\,\sqrt{-3\,x^2 - 2\,x + 1} - 3\,x^2 - 2\,x + 1}.$$

После преобразования получим

$$A(x) = \frac{1}{\sqrt{-3\,x^2 - 2\,x + 1}}.$$

Пример 13. *Найдем производящую функцию для диагонали $T_{2\,n,n}$ в числовом треугольнике, определяемом производящей функцией*

$$G(x)^k = \left(\frac{1 - \sqrt{1 - 4\,x}}{2}\right)^k = \sum_{n \geqslant k} \frac{k}{n} \binom{2\,n - k - 1}{n - k} x^n.$$

*Решение функционального уравнения Лагранжа для данной производящей функции равно (последовательность A*001764 *в [94])*

$$F(x)=\frac{2}{\sqrt{3\,x}}\sin\left(\frac{1}{3}\arcsin\left(\frac{\sqrt{27\,x}}{2}\right)\right).$$

Тогда искомая производящая функция примет вид

$$A(x)=\frac{xF'(x)}{F(x)}=1+\sum_{n>0}\frac{\binom{3n-1}{n}}{2}x^n=\frac{\sqrt{3\,x}}{2\sqrt{4-27\,x}}\,\mathrm{ctg}\left(\frac{1}{3}\arcsin\left(\frac{\sqrt{27x}}{2}\right)\right)+\frac{1}{2}.$$

Пример 14. *Рассмотрим треугольник, который задается следующим образом:*

$$G(x)^m=(x^2\,\mathrm{ctg}(x))^m=\sum_{n\geqslant m}T_{n,m}x^n,$$

где

$$T_{n,m}=(-1)^{\frac{n-m}{2}}\sum_{l=0}^{m}\frac{2^{n-2m+l}}{(n-2m+l)!}\binom{m}{l}\sum_{k=0}^{n-2m+l}\frac{s\,(l+k,l)\;S\,(n-2m+l,k)}{\binom{k+l}{l}}.$$

*Данный треугольник формирует последовательность A*199542 *в [94].*

Тогда

$$T_{2\,n,n}=(-1)^{\frac{n}{2}}\sum_{l=0}^{n}2^l\left(\sum_{k=0}^{l}\frac{k!\,S\,(l,k)\;s\,(l+k,l)}{(l+k)!}\right)\binom{n}{l}.$$

Решением для уравнения $F(x)=x\,F(x)\,\mathrm{ctg}(F(x))$ *будет производящая функция* $\arctan(x)$.

Тогда

$$\frac{xF'(x)}{F(x)}=\frac{x}{(1+x^2)\arctan(x)}=1-\frac{2\,x^2}{3}+\frac{26\,x^4}{45}-\frac{502\,x^6}{945}+\frac{7102\,x^8}{14175}+\cdots$$

Откуда получим искомую производящую функцию

$$A(x)=\frac{x}{(1+x^2)\arctan(x)}=\sum_{n\geqslant 0}(-1)^{\frac{n}{2}}\sum_{l=0}^{n}2^l\binom{n}{l}\sum_{k=0}^{l}\frac{k!\,S\,(l,k)\;s\,(l+k,l)}{(l+k)!}\,x^n.$$

Далее приведены тождества и соотношения для коэффициентов числового треугольника, полученные за счет совместного исследования диагонали $T_{2n,n}$ и диагонали центральных коэффициентов.

Теорема 17. *Пусть есть числовой треугольник $T_{n,k}$, образованный степенью производящей функции $G(x)^k = \sum\limits_{n\geqslant k} T_{n,k}\, x^n$. Тогда имеет место следующее тождество для центральных коэффициентов треугольника:*

$$T_{2n-1,n} = \sum_{i=1}^{n} \frac{1}{i} T_{2i-1,i}\, T_{2(n-i),n-i}. \tag{2.38}$$

Доказательство:

Данное тождество следует из Теоремы 16 и Теоремы 14. Отметим, что

$$F(x) = \sum_{n>0} \frac{1}{n} T_{2n-1,n}\, x^n$$

и

$$\frac{x\, F'(x)}{F(x)} = \sum_{n\geqslant 0} T_{2n,n}\, x^n.$$

Поскольку

$$x\, F'(x) = \left(\frac{x\, F'(x)}{F(x)}\right)\, F(x),$$

применим правило умножения производящих функций и получим искомый результат. □

В качестве приложения Теоремы 17 рассмотрим следующие примеры.

Пример 15. *Рассмотрим числовые треугольники, формируемые числами Стирлинга. Числа Стирлинга первого рода определяются производящей функцией*

$$\psi_k(x) = \sum_{n\geqslant k} s(n,k)\frac{x^n}{n!} = \frac{1}{k!}\ln^k(1+x).$$

Используя (2.38)*, получим следующее тождество для чисел Стирлинга первого рода:*

$$s(2n-1,n) = \sum_{i=1}^{n} \frac{\binom{2n-1}{2i-1}}{\binom{n}{i}} \frac{s\left(2\,i-1,i\right)\; s\left(2\;\left(n-i\right),n-i\right)}{i}.$$

Числа Стирлинга второго рода определяются производящей функцией

$$\Phi_k(x) = \sum_{n\geqslant k} S(n,k)\frac{x^n}{n!} = \frac{1}{k!}(e^x-1)^k.$$

Используя (2.38)*, получим следующее тождество для чисел Стирлинга второго рода:*

$$S(2n-1,n) = \sum_{i=1}^{n} \frac{\binom{2n-1}{2i-1}}{\binom{n}{i}} \frac{S\left(2\,i-1,i\right)\; S\left(2\;\left(n-i\right),n-i\right)}{i}.$$

Пример 16. *Рассмотрим числовой треугольник, образованный выражением*

$$(x\,e^x)^k = \sum_{n\geqslant k} \frac{k^{n-k}}{(n-k)!})x^n.$$

Используя (2.38)*, получим следующее тождество:*

$$\frac{n^{n-1}}{(n-1)!} = \sum_{i=1}^{n} \frac{i^{i-2}\ (n-i)^{n-i}}{(i-1)!\ (n-i)!}.$$

Или после его преобразования получим

$$n^{n-1} = \sum_{i=1}^{n} \binom{n-1}{i-1} i^{i-2}\ (n-i)^{n-i}.$$

Пример 17. *Рассмотрим числовой треугольник, образованный выражением*

$$G(x)^k = \left(\frac{x^2}{e^x-1}\right)^k = \sum_{n\geqslant k} T_{n,k}x^n,$$

где

$$T_{n,m} = \frac{m!}{(n-m)!} \sum_{k=0}^{n-m} \frac{k!\,S_1\,(m+k,m)\ S_2\,(n-m,k)}{(m+k)!}.$$

Решение функционального уравнения Лагранжа для данного случая

$$F(x) = x\frac{F(x)}{e^{F(x)}-1}$$

есть производящая функция $\ln(1+x)$ *(последовательность A*191578 *в [94]).*

Тогда согласно Теореме 16 получим

$$\frac{xF'(x)}{F(x)} = \frac{x}{(1+x)\ln(1+x)} = \sum_{n\geqslant 0} T_{2n,n}\,x^n,$$

где

$$T_{2n,n} = \sum_{k=0}^{n} \frac{k!\,S_2\,(n,k)\ S_1\,(n+k,n)}{(n+k)!}.$$

*Это последовательность A*002208 *в [94].*

Используя Теорему 17, получим следующее тождество:

$$\sum_{m=0}^{n-1} \frac{(-1)^m}{n-m} \sum_{k=0}^{m} \frac{k!\,S_2\,(m,k)\ S_1\,(m+k,m)}{(m+k)!} = 1.$$

2.6 Применение комплексного метода формирования информационных объектов, основанного на k-й степени производящих функций

2.6.1 Применение разработанных методов для нахождения явных выражений производящих функций

Далее рассмотрим приложения разработанных методов для получения коэффициентов двумерных и трехмерных производящих функций.

Пример 18. *Рассмотрим следующую композицию производящих функций:*

$$G(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m = H(A(x,y)) = H(x+y).$$

Композита двумерной производящей функции $A(x,y) = x + y$ *равна*

$$A^{\Delta}(n,m,k) = \binom{k}{n}\delta(m,k-n).$$

Применяя Теорему 5, получим выражение для коэффициентов $g(n,m)$ *для двумерной производящей функции* $G(x,y)$

$$g(n,m) = \sum_{k=0}^{n+m} h(k)A^{\Delta}(n,m,k) = \sum_{k=0}^{n+m} h(k)\binom{k}{n}\delta(m,k-n) = h(n+m)\binom{n+m}{n}.$$

Далее рассмотрим ряд частных случаев для конкретных двумерных производящих функций $G(x,y)$.

Пусть

$$H(x) = \sum_{n\geqslant 0} h(n)x^n = \sum_{n\geqslant 0} x^n = \frac{1}{1-x},$$

тогда

$$G(x,y) = H(x+y) = \frac{1}{1-x-y},$$

$$g(n,m) = h(n+m)\binom{n+m}{n} = \binom{n+m}{n}.$$

Пусть

$$H(x) = \sum_{n\geqslant 0} h(n)x^n = \sum_{n\geqslant 0} \frac{1}{n!}x^n = e^x,$$

тогда

$$G(x,y) = H(x+y) = e^{x+y},$$
$$g(n,m) = h(n+m)\binom{n+m}{n} = \frac{1}{(n+m)!}\binom{n+m}{n} = \frac{1}{n!m!}.$$

Пусть

$$H(x) = \sum_{n>0} h(n)x^n = \sum_{n>0} \frac{(-1)^{n-1}}{n}x^n = \log(1+x),$$

тогда

$$G(x,y) = H(x+y) = \log(1+x+y),$$
$$g(n,m) = h(n+m)\binom{n+m}{n} = \frac{(-1)^{n+m-1}}{n+m}\binom{n+m}{n}, \quad g(0,0) = 0.$$

Пусть производящая функция $H(x,y)$ задает числа Каталана (последовательность A000108 в [94])

$$H(x) = \sum_{n\geqslant 0} C_n x^n = \sum_{n\geqslant 0} \frac{1}{n+1}\binom{2n}{n}x^n = \frac{1-\sqrt{1-4x}}{2x},$$

тогда

$$G(x,y) = H(x+y) = \frac{1-\sqrt{1-4(x+y)}}{2(x+y)},$$
$$g(n,m) = h(n+m)\binom{n+m}{n} = C_{n+m}\binom{n+m}{n} = \frac{1}{n+m+1}\binom{2n+2m}{n+m}\binom{n+m}{n}.$$

Пример 19. *Рассмотрим двумерную производящую функцию для чисел Эйлера первого рода (последовательность A173018 в [94])*

$$E(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} \frac{E_{n,m}}{n!}x^n y^m = \frac{y-1}{y-e^{x(y-1)}}.$$

Представим двумерную производящую функцию $E(x,y)$ как композицию следующих производящих функций:

$$E(x,y) = \frac{y-1}{y-e^{x(y-1)}} = \frac{x(y-1)}{x(y-1)-x(e^{x(y-1)}-1)} = \frac{1}{1-x\frac{e^{x(y-1)}-1}{x(y-1)}} = H(A(x,y)),$$

где

$$H(x) = \sum_{n\geqslant 0} h(n)x^n = \sum_{n\geqslant 0} x^n = \frac{1}{1-x},$$

$$A(x,y) = x\frac{e^{x(y-1)}-1}{x(y-1)} = B(x,C(x,y)),$$
$$B(x,y) = \frac{x}{y}(e^y - 1),$$
$$C(x,y) = x(y-1).$$

Используя тождество для чисел Стирлинга второго рода

$$(e^x - 1)^k = \sum_{n \geqslant k} \left\{ {n \atop k} \right\} \frac{k!}{n!} x^n,$$

получим композиту для производящей функции $B(x,y)$

$$B^{\Delta}(n,m,k) = \left\{ {m+k \atop k} \right\} \frac{k!}{(m+k)!} \delta(n,k).$$

Применяя биномиальную теорему

$$(xy - x)^k = \sum_{m \geqslant 0} \binom{k}{m} (xy)^m (-x)^{k-m} = \sum_{m \geqslant 0} \binom{k}{m} x^k y^m (-1)^{k-m},$$

получим композиту для производящей функции $C(x,y)$

$$C^{\Delta}(n,m,k) = \binom{k}{m} (-1)^{k-m} \delta(n,k).$$

Используя полученные выражения и применяя Теорему 5 для композиции производящих функций $B(x,C(x,y))$*, получим композиту для производящей функции* $A(x,y)$

$$A^{\Delta}(n,m,k) = \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} B^{\Delta}(k_a,k_b,k) C^{\Delta}(n-k_a,m,k_b) =$$

$$= \sum_{k_a=0}^{n+m} \sum_{k_b=0}^{n+m-k_a} \left\{ {k_b+k \atop k} \right\} \frac{k!}{(k_b+k)!} \delta(k_a,k) \binom{k_b}{m} (-1)^{k_b-m} \delta(n-k_a,k_b).$$

Используя свойства символа Кронекера, получим $k_a = k$ *и* $k_b = n-k$*.*

После упрощения $A^{\Delta}(n,m,k)$ *получим*

$$A^{\Delta}(n,m,k) = \left\{ {n \atop k} \right\} \binom{n-k}{m} \frac{k!}{n!} (-1)^{n-k-m}.$$

Применяя Теорему 5 для композиции производящих функций $H(A(x,y))$*, получим известную формулу для чисел Эйлера первого рода [2, Уравнение (6.40)]:*

$$E_{n,m} = n! \sum_{k=0}^{n+m} h(k) A^{\Delta}(n,m,k) = \sum_{k=0}^{n} \left\{ {n \atop k} \right\} \binom{n-k}{m} (-1)^{n-k-m} k!.$$

Пример 20. *Рассмотрим производящую функцию для числового треугольника, которая определяет последовательность A*064189 *в [94]:*

$$G(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m = \frac{2}{1-x-2xy+\sqrt{1-2x-3x^2}} = \frac{M(x)}{1-xyM(x)},$$

где

$$M(x) = \frac{1-x-\sqrt{1-2x-3x^2}}{2x^2}$$

*является производящей функцией для чисел Моцкина (последовательность A*001006 *в [94]).*

Элементы $g(n,m)$ числового треугольника определяют число решеточных путей от $(0,0)$ до (n,m), находящихся над осью x и состоящих из шагов вида $(1,1)$, $(1,-1)$ и $(1,0)$. Рассмотрим двумерную производящую функцию $G(x,y)$ как композицию следующих производящих функций:

$$G(x,y) = \frac{H(x,y)}{xy} = \frac{B(M_{xy}(x,y))}{xy},$$

где

$$H(x,y) = \sum_{n>0}\sum_{m>0} h(n,m)x^n y^m = \frac{xyM(x)}{1-xyM(x)} = \frac{M_{xy}(x,y)}{1-M_{xy}(x,y)} = B(M_{xy}(x,y)),$$

$$B(x) = \sum_{n>0} b(n)x^n = \sum_{n>0} x^n = \frac{x}{1-x},$$

$$M_{xy}(x,y) = xyM(x).$$

Производящая функция $M(x)$ удовлетворяет уравнению

$$M(x) = 1 + xM(x) + x^2M(x).$$

Трансформируем его к виду

$$M_x(x) = xA(M_x(x)), \tag{2.39}$$

где

$$A(x) = 1 + x + x^2,$$

$$M_x(x) = xM(x).$$

Используя биномиальную теорему, получим коэффициенты k-й степени производящей функции $A(x)$

$$a(n,k) = [x^n](1+x+x^2)^k = \sum_{j=0}^{k} \binom{k}{j}\binom{j}{n-j}.$$

Используя теорему обращения Лагранжа для (2.39)*, получим композиту производящей функции* $M_x(x)$

$$M_x^{\Delta}(n,k) = \frac{k}{n}a(n-k,n).$$

Тогда композита двумерной производящей функции $M_{xy}(x,y)$ *будет равна*

$$M_{xy}^{\Delta}(n,m,k) = M_x^{\Delta}(n,k)\delta(m,k).$$

Применяя Теорему 5 для композиции производящих функций $B(M_{xy}(x,y))$*, получим явную формулу для коэффициентов производящей функции* $H(x,y)$

$$h(n,m) = \sum_{k=0}^{n+m} b(k)M_{xy}^{\Delta}(n,m,k) = \sum_{k=1}^{n+m} M_x^{\Delta}(n,k)\delta(m,k) = M_x^{\Delta}(n,m).$$

Откуда коэффициенты $g(n,m)$ *производящей функции* $G(x,y)$ *будут равны*

$$g(n,m) = h(n+1,m+1) = \frac{m+1}{n+1}\sum_{j=0}^{n-m}\binom{n+1}{j}\binom{j}{n-m-j}.$$

Пример 21. *Рассмотрим производящую функцию для числового треугольника, которая определяет последовательность* $A336524$ *в [94]*

$$G(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} g(n,m)x^n y^m = \frac{1-\sqrt{1-4x-4xy}}{2x}.$$

Элементы $g(n,m)$ *числового треугольника определяют количество непомеченных двоичных деревьев с* n *внутренними узлами и* m *различимыми внешними узлами. Рассмотрим двумерную производящую функцию* $G(x,y)$ *как композицию следующих производящих функций:*

$$G(x,y) = \frac{H(x,y)}{x} = \frac{C(A(x,y))}{x},$$

где

$$H(x,y) = \sum_{n>0}\sum_{m\geqslant 0} h(n,m)x^n y^m = \frac{1-\sqrt{1-4x-4xy}}{2} = C(A(x,y)),$$

$$C(x) = \sum_{n>0} c(n)x^n = \sum_{n>0} C_{n-1}x^n = \frac{1-\sqrt{1-4x}}{2},$$

$$A(x,y) = x + xy.$$

Используя биномиальную теорему для

$$(x+xy)^k = \sum_{m\geqslant 0} \binom{k}{m} x^{k-m}(xy)^m = \sum_{m\geqslant 0} \binom{k}{m} x^k y^m,$$

получим композиту двумерной производящей функции $A(x,y)$

$$A^{\Delta}(n,m,k) = \binom{k}{m}\delta(n,k).$$

Применяя Теорему 5 для композиции производящих функций $C(A(x,y))$, *получим явную формулу для коэффициентов производящей функции* $H(x,y)$

$$h(n,m) = \sum_{k=0}^{n+m} c(k)A^{\Delta}(n,m,k) = \sum_{k=1}^{n+m} C_{k-1}\binom{k}{m}\delta(n,k) = C_{n-1}\binom{n}{m}.$$

Откуда коэффициенты $g(n,m)$ *производящей функции* $G(x,y)$ *будут равны*

$$g(n,m) = h(n+1,m) = C_n\binom{n+1}{m} = \frac{1}{n+1}\binom{2n}{n}\binom{n+1}{m}.$$

Пример 22. *Рассмотрим следующую производящую функцию, описывающую вероятность наличия направленного вверх ребра в графе для ацтекского бриллианта размерности* n *[74; 97]:*

$$G(x,y,z) = \frac{\frac{z}{2}}{(1-yz)(1-(x+\frac{1}{x}+y+\frac{1}{y})\frac{z}{2}+z^2)} = \sum_{n>0}\sum_{i=-n}^{n}\sum_{j=-n}^{n} \rho(i,j,n)\, x^i y^j z^n,$$

и найдем явное выражение для коэффициентов данной производящей функции.

Избавимся от отрицательных степеней в переменных следующим путем:

$$F(x,y,z) = G(x,y,xyz) = \sum_{n>0}\sum_{i=-n}^{n}\sum_{j=-n}^{n} \rho(i,j,n)\, x^i y^j (xyz)^n =$$

$$= \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l>0} \rho(n-l,m-l,l)\, x^n y^m z^l = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l>0} f(n,m,l)\, x^n y^m z^l.$$

Представим производящую функцию $F(x,y,z)$ *как рациональную производящую функцию трех переменных*

$$F(x,y,z) = \frac{\frac{xyz}{2}}{(1-xy^2z)(1-(x+\frac{1}{x}+y+\frac{1}{y})\frac{xyz}{2}+x^2y^2z^2)} = \frac{P(x,y,z)}{Q(x,y,z)},$$

где будем использовать следующую декомпозицию производящих функций:

$$P(x,y,z) = \frac{\frac{xyz}{2}}{1-xy^2z} = \frac{1}{2}P_1(x,y,z)P_2(x,y,z) = \frac{1}{2}P_1(x,y,z)\frac{1}{1-P_3(x,y,z)},$$

$$P_1(x,y,z) = xyz, \quad P_2(x,y,z) = \frac{1}{1-xy^2z}, \quad P_3(x,y,z) = xy^2z,$$

$$Q(x,y,z) = 1 - \left(x + \frac{1}{x} + y + \frac{1}{y}\right)\frac{xyz}{2} + x^2y^2z^2 = 1 - (Q_1(x,y,z) + Q_2(x,y,z)),$$

$$Q_1(x,y,z) = \frac{1}{2}(x+y)(1+xy)z = \frac{1}{2}P_1(x+y,1+xy,z),$$

$$Q_2(x,y,z) = -x^2y^2z^2 = -P_1(x,y,z)^2.$$

Найдем явные выражения для всех производящих функций полученных при декомпозиции.

Для производящей функции $P_1(x,y,z)$ *ее коэффициенты определяются выражением*

$$p_1(n,m,l) = \delta(n,1)\,\delta(m,1)\,\delta(l,1),$$

а композита равна

$$P_1^{\Delta}(n,m,l,k) = \delta(n,k)\,\delta(m,k)\,\delta(l,k).$$

Для производящей функции $P_3(x,y,z)$ *ее коэффициенты определяются выражением*

$$p_3(n,m,l) = \delta(n,1)\,\delta(m,2)\,\delta(l,1),$$

а композита равна

$$P_3^{\Delta}(n,m,l,k) = \delta(n,k)\,\delta(m,2k)\,\delta(l,k).$$

Согласно формуле (2.34) *коэффициенты производящей функции* $P_2(x,y,z)$ *равны*

$$p_2(n,m,l) = \sum_{k=0}^{n+m+l} P_3^{\Delta}(n,m,l,k) = \sum_{k=0}^{n+m+l} \delta(n,k)\,\delta(m,2k)\,\delta(l,k) = \delta(m,2n)\,\delta(l,n).$$

Тогда, применяя правило умножения производящих функций, получим следующую формулу для коэффициентов производящей функции $P(x,y,z)$*:*

$$p(n,m,l) = \frac{1}{2}\sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l} p_1(n_i,m_i,l_i)\cdot p_2(n-n_i,m-m_i,l-l_i) =$$

$$= \frac{1}{2}\sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l}\delta(n_i,1)\,\delta(m_i,1)\,\delta(l_i,1)\,\delta(m-m_i,2n-2n_i)\,\delta(l-l_i,n-n_i) =$$

$$= \frac{1}{2}\delta(m,2n-1)\,\delta(l,n).$$

Чтобы получить композиту для $Q_1(x,y,z)$, *используем Теорему 11, где*

$$A(x,y,z) = x+y, \quad A(x,y,z)^k = (x+y)^k = \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l}\binom{k}{n}\delta(m,k-n)\,\delta(l,0),$$

$$B(x,y,z) = 1+xy, \quad B(x,y,z)^k = (1+xy)^k = \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l}\binom{k}{n}\delta(m,n)\,\delta(l,0),$$

$$C(x,y,z) = z, \quad C(x,y,z)^k = z^k = \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l}\delta(n,0)\,\delta(m,0)\,\delta(l,k).$$

Таким образом, получим

$$h_{AB}(n,m,l,k_a,k_b) =$$

$$= \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l}\binom{k_a}{n_i}\delta(m_i,k_a-n_i)\,\delta(l_i,0)\binom{k_b}{n-n_i}\delta(m-m_i,n-n_i)\,\delta(l-l_i,0) =$$

$$= \sum_{n_i=0}^{n}\binom{k_a}{n_i}\binom{k_b}{n-n_i}\delta(2n_i,n-m+k_a)\,\delta(l,0) =$$

$$= \binom{k_a}{\frac{n-m+k_a}{2}}\binom{k_b}{\frac{n+m-k_a}{2}}\frac{(-1)^{n-m+k_a}+1}{2}\,\delta(l,0),$$

$$h_{ABC}(n,m,l,k_a,k_b,k_c) =$$

$$= \sum_{n_i=0}^{n}\sum_{m_i=0}^{m}\sum_{l_i=0}^{l}h_{AB}(n_i,m_i,l_i,k_a,k_b)\,\delta(n-n_i,0)\,\delta(m-m_i,0)\,\delta(l-l_i,k_c) =$$

$$= h_{AB}(n,m,l-k_c,k_a,k_b).$$

Откуда

$$Q_1^{\Delta}(n,m,l,k) =$$

$$= \frac{1}{2^k}\sum_{k_a=0}^{n+m+l}\sum_{k_b=0}^{n+m+l-k_a}\sum_{k_c=0}^{n+m+l-k_a-k_b}P_1^{\Delta}(k_a,k_b,k_c,k)\cdot h_{ABC}(n,m,l,k_a,k_b,k_c) =$$

$$= \frac{1}{2^k} \sum_{k_a=0}^{n+m+l} \sum_{k_b=0}^{n+m+l-k_a} \sum_{k_c=0}^{n+m+l-k_a-k_b} \delta(k_a,k)\,\delta(k_b,k)\,\delta(k_c,k)\,h_{AB}(n,m,l-k_c,k_a,k_b) =$$

$$= \frac{1}{2^k} h_{AB}(n,m,l-k,k,k) = \binom{k}{\frac{n-m+k}{2}} \binom{k}{\frac{n+m-k}{2}} \frac{(-1)^{n-m+k}+1}{2^{k+1}}\,\delta(l,k).$$

Для производящей функции $Q_2(x,y,z)$ композита равна

$$Q_2^{\Delta}(n,m,l,k) = (-1)^k\,\delta(n,2k)\,\delta(m,2k)\,\delta(l,2k).$$

Чтобы получить композиту для суммы $Q_1(x,y,z) + Q_2(x,y,z)$, воспользуемся Теоремой 7 и получим

$$Q_s^{\Delta}(n,m,l,k) =$$

$$= \sum_{k_a=0}^{n+m+l} \binom{k}{k_a} \sum_{n_i=0}^{n} \sum_{m_i=0}^{m} \sum_{l_i=0}^{l} Q_1^{\Delta}(n_i,m_i,l_i,k_a) \cdot Q_2^{\Delta}(n-n_i,m-m_i,l-l_i,k-k_a).$$

После упрощения получим

$$Q_s^{\Delta}(n,m,l,k) = \binom{k}{l-k} \binom{2k-l}{\frac{n-m-l+2k}{2}} \binom{2k-l}{\frac{n+m-3l+2k}{2}} \frac{(-1)^{n+m+k}+(-1)^{l+k}}{2^{2k-l+1}}.$$

Согласно формуле (2.34) *коэффициенты производящей функции*

$$R(x,y,z) = \frac{1}{Q(x,y,z)} = \frac{1}{1-(Q_1(x,y,z)+Q_2(x,y,z))}$$

равны

$$r(n,m,l) = \sum_{k=0}^{n+m+l} Q_s^{\Delta}(n,m,l,k) =$$

$$= \sum_{k=0}^{n+m+l} \binom{k}{l-k} \binom{2k-l}{\frac{n-m-l+2k}{2}} \binom{2k-l}{\frac{n+m-3l+2k}{2}} \frac{(-1)^{n+m+k}+(-1)^{l+k}}{2^{2k-l+1}}.$$

В итоге, применяя правило умножения производящих функций, получим следующую явную формулу для коэффициентов производящей функции $F(x,y,z)$:

$$f(n,m,l) = \sum_{n_i=0}^{n} \sum_{m_i=0}^{m} \sum_{l_i=0}^{l} p(n_i,m_i,l_i) \cdot r(n-n_i,m-m_i,l-l_i) =$$

$$= \frac{1}{2} \sum_{n_i=1}^{n} r(n-n_i,m-2n_i+1,l-n_i).$$

Таким образом, получена явная формула для описания вероятности наличия направленного вверх ребра в графе для ацтекского бриллианта размера n:

$$\rho(i,j,n) = f(n+i,n+j,n).$$

2.6.2 Методы получения явных представлений для производящих функций некоторых классов полиномов

Решение многих задач прикладной математики и информатики, в том числе численного решения различных уравнений и систем, сводится к построению подходящих полиномов, которые удовлетворяют исходным объектам с заданной точностью. Сложность объектов исследования приводит к необходимости усложнения используемых полиномов, в частности, появляются новые классы полиномов. Как следствие, возникает потребность в методах эффективного вычисления явных выражений для соответствующих полиномов и связанных с ними разнообразных тождеств. Также зачастую возникает проблема представления полиномов с использованием их производящих функций.

Производящие функции являются важным элементом описания полиномов. Поэтому развитие методов оперирования производящими функциями специальных полиномов и их применение к получению явных формул является актуальной и востребованной задачей.

В данном разделе развиты методы оперирования производящими функциями специальных полиномов и их применение к получению универсальных методов вычисления явных формул. Введенный математический аппарат над коэффициентами целых степеней производящих функций является удобным и эффективным инструментом для решения разнообразных математических задач. Например, данный аппарат может быть применен для производящих функций полиномов. Как правило, такие производящие функции имеют две переменные и некоторое множество параметров.

Исследование проводится с помощью коэффициентов $F(n,k,x)$ степеней производящей функции вида $F(t,x)^k = \sum\limits_{n\geqslant k} F(n,k,x)\,t^n$, где $F(0,x) = 0$. Эти коэффициенты названы композитами производящих функций полиномов, дополнение касается наличия переменной x в композитах и сама композита является не числовым значением, а выражением полинома.

Для двух производящих функций $F(t,x)$ и $G(t,x)$ и их композит, соответственно, выполняются все вышеописанные операции: операции получения композит сложения $F(t,x) + G(t,x)$, умножения $F(t,x)G(t,x)$, композиции $G(F(t,x))$, взаимных производящих функций $G(x)F(x) = 1$ и обратных производящих функций $G(t,x) = F^{[-1]}(t,x)$.

Также стоит отметить, что важное значение имеют исследования, связанные с получением разных тождеств и свойств заданных полиномов. Так можно отметить в этом направлении работы Й. Шимшека [98–100], Т. Кима [101; 102], Ч.С. Рю [103–105] и других авторов.

Далее покажем применение введенного математического аппарата для получения явных формул для производящих функций некоторых классов полиномов.

Существует достаточно много обобщений полиномов Бернулли, например, проведем исследование следующего обобщения полиномов Бернулли, производящая функция которого имеет вид

$$e^{xt}\left(\frac{t}{e^t-1}\right)^{\alpha}=\sum_{n\geqslant 0}B_n^{(\alpha)}(x)\frac{t^n}{n!},\qquad |t|<2\pi, \tag{2.40}$$

где α — любое действительное или комплексное число.

Х.М. Шривастава и П.Г. Тодоров [42] нашли явную формулу для данного обобщения полиномов Бернулли, в основе которой лежит гипергеометрическая функция:

$$B_n^{(\alpha)}(x)=\sum_{k=0}^{n}\binom{n}{k}\binom{\alpha+k-1}{k}\frac{k!}{(2k)!}\times$$

$$\times\sum_{j=0}^{k}(-1)^j\binom{k}{j}j^{2k}(x+j)^{n-k}F\left(k-n,k-\alpha;2k+1;\frac{j}{x+j}\right).$$

Также известна связь обобщенных полиномов Бернулли с обобщенными числами Бернулли:

$$B_n^{(\alpha)}(x)=\sum_{k=0}^{n}\binom{n}{k}B_n^{(\alpha)}x^{n-k}.$$

При подстановке $x=0$ в (2.40) получаем производящую функцию для обобщенных чисел Бернулли

$$\left(\frac{t}{e^t-1}\right)^{\alpha}=\sum_{n\geqslant 0}B_n^{(\alpha)}\frac{t^n}{n!}.$$

Предложенный комплексный метод позволил получить следующую явную формулу, описывающую обобщенные полиномы Бернулли [106]:

$$B_n^{(\alpha)}(x)=n!\sum_{i=0}^{n}t_i(\alpha)\frac{x^{n-i}}{(n-i)!},$$

где

$$t_n(\alpha)=\sum_{k=0}^{n}(-1)^k\binom{k+\alpha-1}{\alpha-1}\sum_{j=0}^{k}\frac{j!(-1)^{k+j}\binom{k}{j}\left\{\begin{matrix}n+j\\ j\end{matrix}\right\}}{(n+j)!}.$$

Например, при $\alpha=2$ треугольник коэффициентов при x^n записан в последовательности $A197419$ в [94].

Далее рассмотрим вырожденные полиномы Бернулли $\beta_n(\lambda,x)$ и числа $\beta_n(\lambda)$, определенные Л. Карлицом [107; 108] следующими производящими функциями соответственно:

$$\frac{t(1+\lambda t)^{\frac{x}{\lambda}}}{(1+\lambda t)^{\frac{1}{\lambda}}-1}=\sum_{n\geqslant 0}\beta_n(\lambda,x)\frac{t^n}{n!},$$

$$\frac{t}{(1+\lambda t)^{\frac{1}{\lambda}}-1}=\sum_{n\geqslant 0}\beta_n(\lambda)\frac{t^n}{n!}.$$

Несколько первых вырожденных полиномов и чисел Бернулли:

$$\beta_0(\lambda,x)=1,$$

$$\beta_1(\lambda,x)=x-\frac{\lambda-1}{2},$$

$$\beta_2(\lambda,x)=x^2-\lambda x+\frac{\lambda^2-1}{6},$$

$$\beta_3(\lambda,x)=x^3-\frac{3(\lambda+1)}{2}x^2+\frac{\lambda(\lambda+3)}{2}x-\frac{\lambda^2-1}{4};$$

$$\beta_0(\lambda)=1,\qquad \beta_1(\lambda)=-\frac{\lambda-1}{2},\qquad \beta_2(\lambda)=\frac{\lambda^2-1}{6},\qquad \beta_3(\lambda)=-\frac{\lambda^2-1}{4}.$$

Так как $(1+\lambda t)^{\frac{1}{\lambda}}\to e^t$ при $\lambda\to 0$, получаем полиномы Бернулли

$$\frac{te^{xt}}{e^t-1}=\sum_{n\geqslant 0}B_n(x)\frac{t^n}{n!}.$$

Ф.Т. Говард [109] получил явную формулу для вырожденных полиномов Бернулли

$$\beta_n(\lambda)=m!b_m\lambda^m+\sum_{j=1}^{\left[\frac{n}{2}\right]}\frac{n}{2j}B_{2j}\begin{bmatrix}n-1\\ 2j-1\end{bmatrix}\lambda^{n-2j},$$

где $m\geqslant 2$, b_n — число Бернулли второго рода, B_{2j} — число Бернулли.

Также он доказал некоторые рекуррентные формулы для вырожденных чисел Бернулли (Теорема 4.1 и Формула (4.7) в [109]). Другие интересные свойства и тождества вырожденных чисел Бернулли, связанных с p-адическим инвариантным интегралом в Z_p, можно найти в [110–113].

Использование предложенного комплексного метода на основе правил для композит производящих функций полиномов позволило получить явные формулы для вырожденных полиномов и чисел Бернулли. Явная формула для вырожденных чисел Бернулли равна

$$\beta_n(\lambda) = n! \sum_{k=1}^{n} \sum_{i=0}^{k} \binom{k}{i} \sum_{j=0}^{i} \binom{i}{j} (-1)^j \binom{\frac{j}{\lambda}}{n+i} \lambda^{n+i},$$

где $n > 0$. Следовательно, явная формула для вырожденных полиномов Бернулли равна

$$\beta_n(\lambda,x) = \sum_{m=0}^{n} \frac{n!}{(n-m)!} \beta_{n-m}(\lambda) \binom{\frac{x}{\lambda}}{m} \lambda^m$$

или

$$\beta_n(\lambda,x) = \sum_{m=0}^{n} \binom{n}{m} \beta_{n-m}(\lambda) \sum_{l=0}^{m} \begin{bmatrix} m \\ l \end{bmatrix} x^l \lambda^{m-l}.$$

Если рассмотреть сумму

$$\frac{t}{(\lambda t+1)^{\frac{1}{\lambda}} - 1} + t = \frac{t(\lambda t+1)^{\frac{1}{\lambda}}}{(\lambda t+1)^{\frac{1}{\lambda}} - 1},$$

то получим следующую формулу для вырожденных чисел Бернулли:

$$\beta_n(\lambda) = \sum_{i=0}^{n} \beta_i(\lambda) \binom{\frac{1}{\lambda}}{n-i} \lambda^{n-i}$$

или

$$\sum_{i=0}^{n-1} \beta_i(\lambda) \binom{\frac{1}{\lambda}}{n-i} \lambda^{n-i} = 0,$$

где $n > 0$.

Откуда получена следующая рекурсивная формула:

$$\beta_n(\lambda) = -\sum_{i=0}^{n-1} \beta_i(\lambda) \binom{\frac{1}{\lambda}}{n-i+1} \lambda^{n-i+1},$$

где $n > 0$.

Далее покажем применимость предложенного комплексного метода для полиномов Бесселя, введенных в [114]:

$$y_n(x) = \sum_{k=0}^{n} \frac{(n+k)!}{(n-k)!k!} \left(\frac{x}{2}\right)^k.$$

Впоследствии Л. Карлиц [115] определил класс полиномов, связанных с полиномами Бесселя, через

$$p_n(x) = x^n y_{n-1}\left(\frac{1}{x}\right),$$

где $p_n(x)$ определяется явной формулой [52]

$$p_n(x) = \sum_{k=1}^{n} \frac{(2n-k-1)!}{(k-1)!(n-k)!} \left(\frac{1}{2}\right)^{n-k} x^k \tag{2.41}$$

и следующей производящей функцией:

$$\sum_{n=0}^{\infty} \frac{p_n(x)}{n!} t^n = e^{x(1-\sqrt{1-2t})}. \tag{2.42}$$

Используя понятие композиты, из производящей функции (2.42) можно получить явную формулу (2.41).

Для производящей функции

$$C(t) = \frac{1-\sqrt{1-4t}}{2}$$

композита задается следующим образом:

$$C^{\Delta}(n,k) = \frac{k}{n}\binom{2n-k-1}{n-1}.$$

Представим $e^{x(1-\sqrt{1-2t})}$ в виде композиции производящих функций $A(B(t))$, где

$$A(t) = x(1-\sqrt{1-2t}) = 2xC\left(\frac{t}{2}\right),$$

$$B(t) = \sum_{n=0}^{\infty} b(n)t^n = \sum_{n=0}^{\infty} \frac{1}{n!} t^n = e^t.$$

Получим следующую композиту для $A(t)$:

$$A^{\Delta}(n,k) = (2x)^k \left(\frac{1}{2}\right)^n C^{\Delta}(n,k) = 2^{k-n} x^k \frac{k}{n}\binom{2n-k-1}{n-1}.$$

Далее получим искомую явную формулу

$$p_n(x) = n! \sum_{k=1}^{n} A^{\Delta}(n,k)\, b(k) = \sum_{k=1}^{n} \frac{(2n-k-1)!}{(k-1)!\,(n-k)!} 2^{k-n} x^k,$$

которая совпадает с явной формулой (2.41).

Также покажем применение предложенного комплексного метода для описания полиномов Гегенбауэра. Полиномы Гегенбауэра $C_n^{(\alpha)}$, также известные как ультрасферические полиномы, являются решениями дифференциального уравнения Гегенбауэра для целых n:

$$(1-x^2)y'' - (2\alpha+1)xy' + n(n+2\alpha)y = 0.$$

Полиномы Гегенбауэра являются обобщением полиномов Лежандра и Чебышева, а также являются частным случаем многочленов Якоби и определяются следующим гипергеометрическим рядом [116]:

$$C_n^{(\alpha)}(x) = \frac{(2\alpha)_n}{n!}\, {}_2F_1\left(-n, 2\alpha+n; \alpha+\frac{1}{2}; \frac{1}{2}(1-x)\right).$$

Известно следующее явное представление полиномов Гегенбауэра:

$$C_n^{(\alpha)}(x) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-1)^k (\alpha)_{n-k}}{k!(n-2k)!} (2x)^{n-2k}. \tag{2.43}$$

Производящая функция для полиномов Гегенбауэра имеет выражение [117]

$$\left(1-2xt+t^2\right)^{-\alpha} = \sum C_n^{(\alpha)} t^n.$$

Применяя предложенный комплексный метод можно получить следующие явные формулы:

$$C_n^{(\alpha)}(x) = \sum_{m=0}^{n} \alpha^m (-1)^{n-m} \sum_{k=\max\left(m, \lceil \frac{n}{2} \rceil\right)}^{n} \frac{2^{2k-n}}{k!} \begin{bmatrix} k \\ m \end{bmatrix} \binom{k}{n-k} x^{2k-n},$$

$$C_n^{(\alpha)}(x) = \sum_{k=\lceil \frac{n}{2} \rceil}^{n} \binom{k}{n-k} \binom{k+\alpha-1}{k} (-1)^{n-k} (2x)^{2k-n}.$$

После небольшого преобразования получается формула (2.43).

Далее рассмотрим полиномы Коробова. Для фиксированного действительного числа p полиномы и числа Коробова определяются следующими производящими функциями соответственно [118; 119]:

$$F_K(x,t) = \frac{pt(1+t)^x}{(1+t)^p - 1} = \sum_{n \geqslant 0} K_n(p,x)\frac{t^n}{n!},$$

$$F_K(t) = \frac{pt}{(1+t)^p - 1} = \sum_{n \geqslant 0} K_n(p)\frac{t^n}{n!}.$$

П.Т. Янг [120] изучал полиномы Коробова как обратные многочлены вырожденных полиномов Бернулли и нашел некоторые полезные тождества и рекуррентные соотношения. Вырожденные полиномы Бернулли и полиномы Коробова связаны следующим образом:

$$\beta_n(\lambda,x) = \lambda^n K_n\left(\frac{1}{\lambda},\frac{x}{\lambda}\right),$$

$$K_n(p,x) = p^n \beta_n\left(\frac{1}{p},\frac{x}{p}\right).$$

Несколько первых полиномов и чисел Коробова:

$$K_0(p,x) = 1,$$

$$K_1(p,x) = x - \frac{p-1}{2},$$

$$K_2(p,x) = x^2 - px + \frac{p^2-1}{6},$$

$$K_3(p,x) = x^3 - \frac{3(p+1)}{2}x^2 + \frac{p(p+3)}{2}x - \frac{p^2-1}{4};$$

$$K_0(p) = 1, \quad K_1(p) = -\frac{p-1}{2}, \quad K_2(p) = \frac{p^2-1}{6}, \quad K_3(p) = -\frac{p^2-1}{4}.$$

Используя понятие композиты, найдем явную формулу для чисел Коробова. Для этого представим производящую функцию

$$\frac{pt}{(1+t)^p - 1}$$

в виде композиции производящих функций

$$G(t) = \frac{1}{1+t}$$

и

$$A(t) = \frac{(1+t)^p - 1}{tp} - 1,$$

тогда

$$G(A(t)) = \frac{1}{1 + \left(\dfrac{(1+t)^p - 1}{tp} - 1\right)}.$$

Найдем композиту для производящей функции $A(t)$. Так как

$$((1+t)^p - 1)^k = \sum_{n>0} \sum_{j=0}^{k} \binom{k}{j} (-1)^{k-j} \binom{pj}{n} t^n,$$

то коэффициенты

$$\left(\frac{(1+t)^p - 1}{tp}\right)^k$$

определяются по формуле

$$t(n,k) = \frac{1}{p^k} \sum_{j=0}^{k} \binom{k}{j} (-1)^{k-j} \binom{pj}{n+k}.$$

Тогда композита производящей функции $A(t)$ равна

$$A^{\Delta}(n,k) = \sum_{i=0}^{n} (-1)^{k-i} \binom{k}{i} t(n,i)$$

или

$$A^{\Delta}(n,k) = \sum_{i=0}^{n} \binom{k}{i} p^{-i} \sum_{j=0}^{i} \binom{i}{j} (-1)^{k+j} \binom{jp}{n+i}.$$

Ниже представлены первые члены композиты $A^{\Delta}(n,k)$ в виде треугольника:

$$\left[\frac{p-1}{2}\right]$$

$$\left[\frac{p^2-3p+2}{6}, \frac{p^2-2p+1}{4}\right]$$

$$\left[\frac{p^3-6p^2+11p-6}{24}, \frac{p^3-4p^2+5p-2}{6}, \frac{p^3-3p^2+3p-1}{8}\right]$$

Для $k = n$ получаем следующее тождество:

$$\sum_{k=0}^{n} \binom{n}{i} p^{-i} \sum_{j=0}^{i} \binom{i}{j} (-1)^{n+j} \binom{jp}{n+i} = \frac{1}{2^n} (p-1)^n.$$

Используя формулу для коэффициентов композиции производящих функций, получим следующее выражение для $G(A(t)) = \sum\limits_{n\geqslant 0} c_n t^n$:

$$c_n = \begin{cases} 1, & n = 0, \\ \sum\limits_{k=1}^{n} A^{\Delta}(n,k)(-1)^k = \sum\limits_{k=1}^{n}\sum\limits_{i=0}^{n} \binom{k}{i} p^{-i} \sum\limits_{j=0}^{i} \binom{i}{j} (-1)^j \binom{jp}{n+i}, & n > 0. \end{cases}$$

Тогда явная формула для чисел Коробова для $n > 0$ равна

$$K_n(p) = n! \sum_{k=1}^{n}\sum_{i=0}^{n} \binom{k}{i} p^{-i} \sum_{j=0}^{i} \binom{i}{j} (-1)^j \binom{jp}{n+i}.$$

Следовательно, получаем явную формулу для полиномов Коробова:

$$K_n(p,x) = \sum_{m=0}^{n} \frac{n!}{(n-m)!} K_{n-m}(p) \binom{x}{m}$$

или

$$K_n(p,x) = \sum_{m=0}^{n} \binom{n}{m} K_{n-m}(p) \sum_{l=0}^{m} \begin{bmatrix} m \\ l \end{bmatrix} x^l.$$

Далее рассмотрим полиномы Мейкснера. Полиномы Мейкснера первого рода определяются следующей рекуррентной формулой [121; 122]:

$$m_{n+1}(x;\beta,c) = \frac{(c-1)^n}{c^n} \left((x - b_n) m_n(x;\beta,c) - \lambda_n m_{n-1}(x;\beta,c) \right),$$

где

$$m_0(x;\beta,c) = 1,$$

$$m_1(x;\beta,c) = x - b_0,$$

$$b_n = \frac{(1+c)n + \beta c}{1-c},$$

$$\lambda_n = \frac{cn(n+\beta-1)}{(1-c)^2}.$$

Несколько первых полиномов Мейкснера первого рода:

$$m_0(x;\beta,c) = 1;$$

$$m_1(x;\beta,c) = \frac{x(c-1) + \beta c}{c};$$

$$m_2(x;\beta,c) = \frac{x^2(c-1)^2 + x((2\beta+1)c^2 - 2\beta c - 1) + (\beta+1)\beta c^2}{c^2}.$$

Также полиномы Мейкснера первого рода определяются следующей производящей функцией [52]:

$$\sum_{n=0}^{\infty} \frac{m_n(x;\beta,c)}{n!} t^n = \left(1 - \frac{t}{c}\right)^x (1-t)^{-x-\beta}. \tag{2.44}$$

Используя понятие композиты, из производящей функции (2.44) можно получить явную формулу для $m_n(x;\beta,c)$. Представим производящую функцию (2.44) в виде произведения производящих функций $A(t)B(t)$, где функции $A(t)$ и $B(t)$ раскладываются в ряд по формуле бинома Ньютона:

$$A(t) = \sum_{n=0}^{\infty} a(n)\, t^n = \left(1 - \frac{t}{c}\right)^x = \sum_{n=0}^{\infty} \binom{x}{n} (-1)^n c^{-n} t^n,$$

$$B(t) = \sum_{n=0}^{\infty} b(n)\, t^n = (1-t)^{-x-\beta} = \sum_{n=0}^{\infty} \binom{-x-\beta}{n} (-1)^n t^n.$$

Далее, используя правило умножения производящих функций, получим новую явную формулу для полиномов Мейкснера первого рода:

$$m_n(x;\beta,c) = n! \sum_{k=0}^{n} a(k)\, b(n-k) = (-1)^n n! \sum_{k=0}^{n} \binom{x}{k} \binom{-x-\beta}{n-k} c^{-k}.$$

Полиномы Мейкснера второго рода определяются следующей рекуррентной формулой [121; 122]:

$$M_{n+1}(x;\delta,\eta) = (x - b_n) M_n(x;\delta,\eta) - \lambda_n M_{n-1}(x;\delta,\eta),$$

где

$$M_0(x;\delta,\eta) = 1,$$

$$M_1(x;\delta,\eta) = x - b_0,$$

$$b_n = (2n + \eta)\delta,$$

$$\lambda_n = (\delta^2 + 1) n (n + \eta - 1).$$

Несколько первых полиномов Мейкснера второго рода:

$$M_0(x;\delta,\eta) = 1;$$

$$M_1(x;\delta,\eta) = x - \delta\eta;$$

$$M_2(x;\delta,\eta) = x^2 - x2\delta(1+\eta) + \eta((\eta+1)\delta^2 - 1).$$

Также полиномы Мейкснера второго рода определяются следующей производящей функцией [52]:

$$\sum_{n=0}^{\infty} \frac{M_n(x;\delta,\eta)}{n!} t^n = \left((1+\delta t)^2 + t^2\right)^{\frac{-\eta}{2}} \exp\left(x \ \mathrm{tg}^{-1}\left(\frac{t}{1+\delta t}\right)\right). \tag{2.45}$$

Используя понятие композиты, из производящей функции (2.45) можно получить явную формулу для $M_n(x;\delta,\eta)$. Представим производящую функцию (2.45) в виде произведения производящих функций $A(t)B(t)$, где

$$A(t) = \sum_{n=0}^{\infty} a(n)t^n = \left((1+\delta t)^2 + t^2\right)^{\frac{-\eta}{2}} = \left(1 + 2\delta t + (\delta^2+1)t^2\right)^{\frac{-\eta}{2}},$$

$$B(t) = \sum_{n=0}^{\infty} b(n)t^n = \exp\left(x \, \mathrm{tg}^{-1}\left(\frac{t}{1+\delta t}\right)\right).$$

Затем представим $A(t)$ в виде композиции производящих функций $A_1(A_2(t))$ и разложим в ряд $A_1(t)$ по формуле бинома Ньютона, то есть

$$A_1(t) = \sum_{n=0}^{\infty} a_1(n)t^n = (1+t)^{\frac{-\eta}{2}} = \sum_{n=0}^{\infty} \binom{\frac{-\eta}{2}}{n} t^n,$$

$$A_2(t) = 2\delta t + (\delta^2+1)t^2.$$

Композита для производящей функции $ax + bx^2$ равна

$$a^{2k-n} b^{n-k} \binom{k}{n-k}.$$

Тогда композита для производящей функции $A_2(t)$ равна

$$A_2^{\Delta}(n,k) = (2\delta)^{2k-n}(\delta^2+1)^{n-k}\binom{k}{n-k}.$$

Далее получим коэффициенты производящей функции $A(t)$:

$$a(0) = a_1(0) = 1,$$

$$a(n) = \sum_{k=1}^{n} A_2^{\Delta}(n,k)a_1(k) = \sum_{k=1}^{n} (2\delta)^{2k-n}(\delta^2+1)^{n-k}\binom{k}{n-k}\binom{\frac{-\eta}{2}}{k}.$$

Следующим шагом представим $B(t)$ в виде композиции производящих функций $B_1(B_2(t))$, где

$$B_1(t) = \sum_{n=0}^{\infty} b_1(n)t^n = e^t = \sum_{n=0}^{\infty} \frac{1}{n!} t^n,$$

$$B_2(t) = x \ \mathrm{tg}^{-1}\left(\frac{t}{1+\delta t}\right).$$

Также представим $B_2(t)$ в виде композиции производящих функций $xC_1(C_2(t))$, где

$$C_1(t) = \mathrm{tg}^{-1}(t),$$

$$C_2(t) = \frac{t}{1+\delta t}.$$

Композита для производящей функции $C_1(t)$ задается следующим образом [68]:

$$C_1^{\Delta}(n,k) = \left((-1)^{\frac{3n+k}{2}} + (-1)^{\frac{n-k}{2}}\right)\frac{k!}{2^{k+1}}\sum_{j=k}^{n}\frac{2^j}{j!}\binom{n-1}{j-1}\begin{bmatrix} j \\ k \end{bmatrix}.$$

Композита для производящей функции $\frac{at}{1-bt}$ равна

$$\binom{n-1}{k-1}a^k b^{n-k}.$$

Тогда композита для производящей функции $C_2(t)$ равна

$$C_2^{\Delta}(n,k) = \binom{n-1}{k-1}(-\delta)^{n-k}.$$

Далее получим композиту для производящей функции $B_2(t)$:

$$B_2^{\Delta}(n,k) = x^k \sum_{m=k}^{n} C_2^{\Delta}(n,m)C_1^{\Delta}(m,k).$$

Затем получим коэффициенты производящей функции $B(t)$:

$$b(0) = b_1(0) = 1,$$

$$b(n) = \sum_{k=1}^{n} B_2^{\Delta}(n,k)b_1(k) =$$

$$= \sum_{k=1}^{n}\frac{x^k}{n}\sum_{m=k}^{n}\binom{n}{m}\delta^{n-m}\left(\frac{1+(-1)^{m+k}}{(-1)^{\frac{m+k}{2}}-n}\right)\sum_{j=k}^{m}\frac{2^{j-k-1}}{(j-1)!}\binom{m}{j}\begin{bmatrix} j \\ k \end{bmatrix}.$$

Далее, используя правило умножения производящих функций, получим новую явную формулу для полиномов Мейкснера второго рода:

$$M_n(x;\delta,\eta) = n!\sum_{k=0}^{n} a(k)b(n-k).$$

Далее рассмотрим применение предложенного комплексного метода для чисел Эйлера второго рода, которые активно применяются в комбинаторике, теории чисел, теории графов, аналитической геометрии и других областях математики и теоретической информатики [2;123–125]. Числа Эйлера второго рода обозначаются как $\left\langle\!\left\langle {n \atop m} \right\rangle\!\right\rangle$ и могут быть представлены в форме числового треугольника.

Рассмотрим частный случай чисел Эйлера второго рода, который представлен последовательностью $A008517$ в [94]. Элементы данной последовательности определяются следующим рекуррентным соотношением для $1 \leqslant m \leqslant n$ [126]:

$$\left\langle\!\left\langle {n \atop m} \right\rangle\!\right\rangle = m \left\langle\!\left\langle {n-1 \atop m} \right\rangle\!\right\rangle + (2n-m) \left\langle\!\left\langle {n-1 \atop m-1} \right\rangle\!\right\rangle, \tag{2.46}$$

при этом $\left\langle\!\left\langle {1 \atop 1} \right\rangle\!\right\rangle = 1$ и $\left\langle\!\left\langle {n \atop m} \right\rangle\!\right\rangle = 0$ для $m < 1$ или $m > n$.

Для таких чисел Эйлера второго рода найдены новая явная формула (2.48) и доказана явная формула (2.47), что отражено в следующей теореме:

Теорема 18. *Для чисел Эйлера второго рода справедливо:*

$$\left\langle\!\left\langle {n \atop m} \right\rangle\!\right\rangle = \sum_{k=0}^{m} (-1)^{m-k} \binom{2n+1}{m-k} \left\{ {n+k \atop k} \right\}, \tag{2.47}$$

$$\left\langle\!\left\langle {n \atop m} \right\rangle\!\right\rangle = \sum_{k=0}^{m} (-1)^{m-k} k \binom{2n}{m-k} \left\{ {n+k-1 \atop k} \right\}. \tag{2.48}$$

Доказательство:

Для доказательства мы используем следующие известные тождества:

1. Рекуррентное соотношение для биномиальных коэффициентов [2]

$$\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}; \tag{2.49}$$

2. Рекуррентное соотношение для чисел Стирлинга второго рода [2]

$$\left\{ {n \atop m} \right\} = m \left\{ {n-1 \atop m} \right\} + \left\{ {n-1 \atop m-1} \right\}. \tag{2.50}$$

Применим метод математической индукции.

Для $n = 1$ и $m = 1$ обе формулы (2.47) и (2.48) равны 1. Предположим, что формула (2.47) справедлива для $\left\langle\!\left\langle {n-1 \atop m} \right\rangle\!\right\rangle$ и $\left\langle\!\left\langle {n-1 \atop m-1} \right\rangle\!\right\rangle$, тогда

$$\left\langle\!\left\langle {n-1 \atop m} \right\rangle\!\right\rangle = \sum_{k=0}^{m} (-1)^{m-k} \binom{2n-1}{m-k} \left\{ {n+k-1 \atop k} \right\},$$

$$\left\langle\!\left\langle {n-1 \atop m-1} \right\rangle\!\right\rangle = \sum_{k=0}^{m-1} (-1)^{m-k-1} \binom{2n-1}{m-k-1} \left\{ {n+k-1 \atop k} \right\}.$$

Воспользуемся правой частью тождества (2.46) и полученными выражениями для $\left\langle\!\left\langle {n-1 \atop m} \right\rangle\!\right\rangle$ и $\left\langle\!\left\langle {n-1 \atop m-1} \right\rangle\!\right\rangle$

$$m \left\langle\!\left\langle {n-1 \atop m} \right\rangle\!\right\rangle + (2n-m) \left\langle\!\left\langle {n-1 \atop m-1} \right\rangle\!\right\rangle =$$

$$= m \sum_{k=0}^{m} (-1)^{m-k} \binom{2n-1}{m-k} \left\{ {n+k-1 \atop k} \right\} +$$

$$+ (2n-m) \sum_{k=0}^{m-1} (-1)^{m-k-1} \binom{2n-1}{m-k-1} \left\{ {n+k-1 \atop k} \right\}.$$

Заметим, что вторая сумма равна 0 для $k = m$, тогда, комбинируя обе суммы, получим следующее выражение:

$$m \left\langle\!\left\langle {n-1 \atop m} \right\rangle\!\right\rangle + (2n-m) \left\langle\!\left\langle {n-1 \atop m-1} \right\rangle\!\right\rangle =$$

$$\sum_{k=0}^{m} (-1)^{m-k} \left(m \binom{2n-1}{m-k} - (2n-m) \binom{2n-1}{m-k-1} \right) \left\{ {n+k-1 \atop k} \right\}.$$

Рассматривая выражение внутри скобок

$$m \binom{2n-1}{m-k} - (2n-m) \binom{2n-1}{m-k-1} =$$

$$= m \left(\binom{2n-1}{m-k} + \binom{2n-1}{m-k-1} \right) - 2n \binom{2n-1}{m-k-1} =$$

$$= m \binom{2n}{m-k} - 2n \frac{m-k}{2n} \binom{2n}{m-k} = k \binom{2n}{m-k}$$

и комбинируя полученные результаты, получим тождество (2.48).

Далее воспользуемся тождествами (2.49) и (2.50) для формулы (2.47):

$$\sum_{k=0}^{m} (-1)^{m-k} \binom{2n+1}{m-k} \left\{ {n+k \atop k} \right\} =$$

$$= \sum_{k=0}^{m} (-1)^{m-k} k \binom{2n}{m-k} \left\{ {n+k-1 \atop k} \right\} +$$

$$+ \sum_{k=0}^{m} (-1)^{m-k} \binom{2n}{m-k} \left\{ {n+k-1 \atop k-1} \right\} + \tag{2.51}$$

$$+ \sum_{k=0}^{m} (-1)^{m-k} \binom{2n}{m-k-1} \left\{ {n+k \atop k} \right\}. \tag{2.52}$$

Выражение внутри суммы (2.51) равно 0 для $k = 0$. Также подставляя $k + 1$ вместо k, получим

$$\sum_{k=0}^{m} (-1)^{m-k} \binom{2n}{m-k} \left\{ {n+k-1 \atop k-1} \right\} = \sum_{k=1}^{m} (-1)^{m-k} \binom{2n}{m-k} \left\{ {n+k-1 \atop k-1} \right\} =$$

$$= \sum_{k=0}^{m-1} (-1)^{m-k-1} \binom{2n}{m-k-1} \left\{ {n+k \atop k} \right\} =$$

$$= -\sum_{k=0}^{m-1} (-1)^{m-k} \binom{2n}{m-k-1} \left\{ {n+k \atop k} \right\}.$$

Выражение внутри суммы (2.52) равно 0 для $k = m$. Тогда

$$\sum_{k=0}^{m} (-1)^{m-k} \binom{2n}{m-k-1} \left\{ {n+k \atop k} \right\} = \sum_{k=0}^{m-1} (-1)^{m-k} \binom{2n}{m-k-1} \left\{ {n+k \atop k} \right\}.$$

Комбинируя полученные результаты, имеем равенство правых частей формул (2.47) и (2.48)

$$\sum_{k=0}^{m} (-1)^{m-k} \binom{2n+1}{m-k} \left\{ {n+k \atop k} \right\} = \sum_{k=0}^{m} (-1)^{m-k} k \binom{2n}{m-k} \left\{ {n+k-1 \atop k} \right\}.$$

Откуда получим искомый результат

$$\begin{aligned}
& m \left\langle\!\left\langle {n-1 \atop m} \right\rangle\!\right\rangle + (2n-m) \left\langle\!\left\langle {n-1 \atop m-1} \right\rangle\!\right\rangle = \\
&= m \sum_{k=0}^{m} (-1)^{m-k} \binom{2n-1}{m-k} \left\{ {n+k-1 \atop k} \right\} + \\
&\quad + (2n-m) \sum_{k=0}^{m-1} (-1)^{m-k-1} \binom{2n-1}{m-k-1} \left\{ {n+k-1 \atop k} \right\} = \\
&= \sum_{k=0}^{m} (-1)^{m-k} k \binom{2n}{m-k} \left\{ {n+k-1 \atop k} \right\} = \\
&= \sum_{k=0}^{m} (-1)^{m-k} \binom{2n+1}{m-k} \left\{ {n+k \atop k} \right\} = \\
&= \left\langle\!\left\langle {n \atop m} \right\rangle\!\right\rangle.
\end{aligned}$$

Таким образом, доказана справедливость двух предложенных тождеств. □

2.6.3 Метод получения явных формул и тождеств для полиномов, заданных производящими функциями вида $F(t)^x \cdot G(t)^{\alpha}$

Рассмотрим случай производящих функций, которые могут быть представлены произведением степеней производящих функций $F(t)^x \cdot G(t)^{\alpha}$. Для таких производящих функций получены несколько свойств, которые представлены в следующей теореме:

Теорема 19. *Если $A(t)$ — производящая функция следующего вида:*

$$A(t) = F(t)^x \cdot G(t)^{\alpha} = \sum_{n \geqslant 0} a(n, x, \alpha)\, t^n,$$

тогда:

1. Для композиции производящих функций $D(t) = C(B(t)) = \sum\limits_{n \geqslant 0} d(n)\, t^n$, где $B(t) = tA(t)$ и $C(t) = \sum\limits_{n \geqslant 0} c(n)\, t^n$, выполняется

$$d(n) = d(n, x, \alpha) = \sum_{k=1}^{n} a(n-k, kx, k\alpha)c(k), \quad d(0) = c(0); \tag{2.53}$$

2. Для обратной производящей функции $\overline{B}(t)$ для производящей функции $B(t) = tA(t)$, выполняется

$$\overline{B}(t) = \sum_{n>0} \frac{1}{n} a(n-1, -nx, -n\alpha)\, t^n; \tag{2.54}$$

3. Выполняются следующие тождества:

$$\sum_{m=k}^{n} a(n-m, mx, m\alpha)\frac{k}{m}a(m-k, -mx, -m\alpha) = \delta_{n,k},$$

$$\sum_{m=k}^{n} \frac{m}{n}a(n-m, -nx, -n\alpha)a(m-k, kx, k\alpha) = \delta_{n,k}.$$

Доказательство:

Получим k-ю степень производящей функции $B(t) = tA(t)$

$$B(t)^k = (tA(t))^k = t^k \left(F(t)\right)^{xk} \left(G(t)\right)^{\alpha k} =$$

$$= t^k \sum_{n \geqslant 0} a(n, kx, k\alpha)t^n = \sum_{n \geqslant k} a(n-k, kx, k\alpha)t^n.$$

Следовательно, композита для $B(t) = tA(t)$ равна

$$B^{\Delta}(n,k) = a(n-k, kx, k\alpha). \tag{2.55}$$

Используя правило композиции производящих функций и формулу (2.55), получим (2.53). Композита обратной производящей функции $\overline{A}(t)$ для $A(t)$ равна

$$\overline{A}^{\Delta}(n,k) = \frac{k}{n} R^{\Delta}(2n-k, n), \tag{2.56}$$

где $R^{\Delta}(n,k)$ — композита производящей функции $R(t) = \frac{t}{A(t)}$.

Затем получим k-ю степень производящей функции $R(t)$

$$R(t)^k = \left(\frac{t}{A(t)}\right)^k = t^k \left(F(t)\right)^{-xk} \left(G(t)\right)^{-\alpha k} =$$

$$= t^k \sum_{n \geqslant 0} a(n, -kx, -k\alpha)\, t^n = \sum_{n \geqslant k} a(n-k, -kx, -k\alpha)\, t^n.$$

Следовательно, композита для $R(t)$ равна

$$R^{\Delta}(n,k) = a(n-k, -kx, -k\alpha). \tag{2.57}$$

Используя (2.55), (2.56) и (2.57), получим

$$\overline{B}^{\Delta}(n,k) = \frac{k}{n} a(2n-k-n, -nx, -n\alpha) = \frac{k}{n} a(n-k, -nx, -n\alpha).$$

Для $k = 1$ получаем (2.54).

Применяя (2.18) для композиции $C(t) = B(\overline{B}(t)) = t$, получаем

$$C^{\Delta}(n,k) = \sum_{m=k}^{n} \overline{B}^{\Delta}(n,m) B^{\Delta}(m,k) =$$

$$= \sum_{m=k}^{n} \frac{m}{n} a(n-m, -nx, -n\alpha) a(m-k, kx, k\alpha) = \delta_{n,k}.$$

Применяя (2.18) для композиции $D(t) = \overline{B}(B(t)) = x$, получаем

$$D^{\Delta}(n,k) = \sum_{m=k}^{n} B^{\Delta}(n,m) \overline{B}^{\Delta}(m,k) =$$

$$= \sum_{m=k}^{n} a(n-m, mx, m\alpha) \frac{k}{m} a(m-k, -mx, -m\alpha) = \delta_{n,k}.$$

□

Применим результаты Теоремы 19 для обобщенных полиномов Бернулли, обобщенных полиномов Эйлера, полиномов Эйлера-Фробениуса, Сильвестра, Абеля и других.

Обобщенные полиномы Бернулли определяются следующей производящей функцией [127; 128]:

$$B(t,x,\alpha)=e^{xt}\left(\frac{t}{e^t-1}\right)^{\alpha}=\left(e^t\right)^x\left(\frac{t}{e^t-1}\right)^{\alpha}=\sum_{n\geqslant 0}B_n^{(\alpha)}(x)\frac{t^n}{n!},$$

где

$$B_n^{(\alpha)}(x)=\sum_{i=0}^{n}\frac{n!}{(n+i)!}\binom{n+\alpha}{n-i}\binom{i+\alpha-1}{i}\sum_{j=0}^{i}(-1)^j\binom{i}{j}(x+j)^{n+i}.$$

Согласно (2.55) композита производящей функции $D(t)=tB(t,x,\alpha)$ равна

$$D^{\Delta}(n,k)=\frac{B_{n-k}^{(k\alpha)}(kx)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t)=tB(t,x,\alpha)$ равна

$$\overline{D}^{\Delta}(n,k)=\frac{k}{n}\frac{B_{n-k}^{(-n\alpha)}(-nx)}{(n-k)!}.$$

Также можно получить следующие новые тождества для обобщенных полиномов Бернулли:

$$\sum_{m=k}^{n}\frac{m}{n}\frac{B_{n-m}^{(-n\alpha)}(-nx)}{(n-m)!}\frac{B_{m-k}^{(k\alpha)}(kx)}{(m-k)!}=\delta_{n,k}$$

и

$$\sum_{m=k}^{n}\frac{B_{n-m}^{(m\alpha)}(mx)}{(n-m)!}\frac{k}{m}\frac{B_{m-k}^{(-m\alpha)}(-mx)}{(m-k)!}=\delta_{n,k}.$$

Обобщенные полиномы Эйлера определяются следующей производящей функцией [127]:

$$E(t,x,\alpha)=e^{xt}\left(\frac{2}{e^t+1}\right)^{\alpha}=\left(e^t\right)^x\left(\frac{2}{e^t+1}\right)^{\alpha}=\sum_{n\geqslant 0}E_n^{(\alpha)}(x)\frac{t^n}{n!},$$

где

$$E_n^{(\alpha)}(x)=\sum_{i=0}^{n}\frac{1}{2^i}\binom{i+\alpha-1}{i}\sum_{j=0}^{i}(-1)^j\binom{i}{j}(x+j)^n.$$

Согласно (2.55) композита производящей функции $D(t) = tE(t, x, \alpha)$ равна

$$D^{\Delta}(n,k) = \frac{E_{n-k}^{(k\alpha)}(kx)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tE(t, x, \alpha)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n}\frac{E_{n-k}^{(-n\alpha)}(-nx)}{(n-k)!}.$$

Также можно получить следующие новые тождества для обобщенных полиномов Эйлера:

$$\sum_{m=k}^{n} \frac{m}{n}\frac{E_{n-m}^{(-n\alpha)}(-nx)}{(n-m)!}\frac{E_{m-k}^{(k\alpha)}(kx)}{(m-k)!} = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} \frac{E_{n-m}^{(m\alpha)}(mx)}{(n-m)!}\frac{k}{m}\frac{E_{m-k}^{(-m\alpha)}(-mx)}{(m-k)!} = \delta_{n,k}.$$

Полиномы Фробениуса-Эйлера определяются следующей производящей функцией [129]:

$$H(t, x, \alpha, \lambda) = e^{xt}\left(\frac{1-\lambda}{e^t-\lambda}\right)^{\alpha} = \left(e^t\right)^x\left(\frac{1-\lambda}{e^t-\lambda}\right)^{\alpha} = \sum_{n\geqslant 0} H_n^{(\alpha)}(x,\lambda)\frac{t^n}{n!},$$

где

$$H_n^{(\alpha)}(x,\lambda) = \sum_{i=0}^{n}\frac{1}{(1-\lambda)^i}\binom{i+\alpha-1}{i}\sum_{j=0}^{i}(-1)^j\binom{i}{j}(x+j)^n.$$

Согласно (2.55) композита производящей функции $D(t) = tH(t, x, \alpha, \lambda)$ равна

$$D^{\Delta}(n,k) = \frac{H_{n-k}^{(k\alpha)}(kx,\lambda)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tH(t, x, \alpha, \lambda)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n}\frac{H_{n-k}^{(-n\alpha)}(-nx,\lambda)}{(n-k)!}.$$

Также можно получить следующие новые тождества для полиномов Фробениуса-Эйлера:

$$\sum_{m=k}^{n} \frac{m}{n} \frac{H_{n-m}^{(-n\alpha)}(-nx,\lambda)}{(n-m)!} \frac{H_{m-k}^{(k\alpha)}(kx,\lambda)}{(m-k)!} = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} \frac{H_{n-m}^{(m\alpha)}(mx,\lambda)}{(n-m)!} \frac{k}{m} \frac{H_{m-k}^{(-m\alpha)}(-mx,\lambda)}{(m-k)!} = \delta_{n,k}.$$

Обобщенные полиномы Сильвестра определяются следующей производящей функцией [130]:

$$F(t,x,\alpha) = (1-t)^{-x} e^{\alpha x t} = \left(\frac{e^{\alpha t}}{1-t}\right)^{x} = \sum_{n\geqslant 0} F_n(x,\alpha) t^n,$$

где

$$F_n(x,\alpha) = \sum_{i=0}^{n} \frac{(\alpha x)^{n-i}}{(n-i)!} \binom{i+x-1}{i}.$$

Согласно (2.55) композита производящей функции $D(t) = tF(t,x,\alpha)$ равна

$$D^{\Delta}(n,k) = F_{n-k}(kx,\alpha).$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tF(t,x,\alpha)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n} F_{n-k}(-nx,\alpha).$$

Также можно получить следующие новые тождества для обобщенных полиномов Сильвестра:

$$\sum_{m=k}^{n} \frac{m}{n} F_{n-m}(-nx,\alpha) F_{m-k}(kx,\alpha) = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} F_{n-m}(mx,\alpha) \frac{k}{m} F_{m-k}(-mx,\alpha) = \delta_{n,k}.$$

Обобщенные полиномы Лаггера определяются следующей производящей функцией [52]:

$$L(t,x,\alpha) = (1-t)^{-\alpha-1} e^{\frac{xt}{t-1}} = \left(e^{\frac{t}{t-1}}\right)^{x} \left(\frac{1}{1-t}\right)^{\alpha+1} = \sum_{n\geqslant 0} L_n^{(\alpha)}(x) t^n,$$

где

$$L_n^{(\alpha)}(x) = \sum_{i=0}^{n} \frac{(-x)^i}{i!} \binom{n+\alpha}{n-i}.$$

Согласно (2.55) композита производящей функции $D(t) = tL(t, x, \alpha)$ равна

$$D^{\Delta}(n,k) = L_{n-k}^{(k\alpha+k-1)}(kx).$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tL(t, x, \alpha)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n} L_{n-k}^{(-n\alpha-n-1)}(-nx).$$

Также можно получить следующие новые тождества для обобщенных полиномов Лаггера:

$$\sum_{m=k}^{n} \frac{m}{n} L_{n-m}^{(-n\alpha-n-1)}(-nx) L_{m-k}^{(k\alpha+k-1)}(kx) = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} L_{n-m}^{(m\alpha+m-1)}(mx) \frac{k}{m} L_{m-k}^{(-m\alpha-m-1)}(-mx) = \delta_{n,k}.$$

Полиномы Абеля определяются следующей производящей функцией [52;131]:

$$A(t, x, \alpha) = e^{\frac{W(\alpha t)x}{\alpha}} = \left(e^{\frac{W(\alpha t)}{\alpha}}\right)^x = \sum_{n \geqslant 0} A_n(x, \alpha) \frac{t^n}{n!},$$

где $W(t)$ есть функция Ламбера W и

$$A_n(x, \alpha) = x(x - \alpha n)^{n-1}.$$

Согласно (2.55) композита производящей функции $D(t) = tA(t, x, \alpha)$ равна

$$D^{\Delta}(n,k) = \frac{A_{n-k}(kx, \alpha)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tA(t, x, \alpha)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n} \frac{A_{n-k}(-nx, \alpha)}{(n-k)!}.$$

Также можно получить следующие новые тождества для полиномов Абеля:

$$\sum_{m=k}^{n} \frac{m}{n} \frac{A_{n-m}(-nx, \alpha)}{(n-m)!} \frac{A_{m-k}(kx, \alpha)}{(m-k)!} = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} \frac{A_{n-m}(mx,\alpha)}{(n-m)!}\frac{k}{m}\frac{A_{m-k}(-mx,\alpha)}{(m-k)!} = \delta_{n,k}.$$

Полиномы Бесселя определяются следующей производящей функцией [52]:

$$B(t,x) = e^{x(1-\sqrt{1-2t})} = \left(e^{1-\sqrt{1-2t}}\right)^{x} = \sum_{n\geqslant 0} B_n(x)\frac{t^n}{n!},$$

где

$$B_n(x) = \begin{cases} 1, & n = 0; \\ \sum\limits_{k=1}^{n} \frac{(2n-k-1)!}{(n-k)!(k-1)!}\frac{x^k}{2^{n-k}}, & n > 0. \end{cases}$$

Согласно (2.55) композита производящей функции $D(t) = tB(t,x)$ равна

$$D^{\Delta}(n,k) = \frac{B_{n-k}(kx)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tB(t,x)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n}\frac{B_{n-k}(-nx)}{(n-k)!}.$$

Также можно получить следующие новые тождества для полиномов Бесселя:

$$\sum_{m=k}^{n} \frac{m}{n}\frac{B_{n-m}(-nx)}{(n-m)!}\frac{B_{m-k}(kx)}{(m-k)!} = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} \frac{B_{n-m}(mx)}{(n-m)!}\frac{k}{m}\frac{B_{m-k}(-mx)}{(m-k)!} = \delta_{n,k}.$$

Полиномы Стирлинга определяются следующей производящей функцией [52; 132]:

$$S(t,x) = \left(\frac{t}{1-e^{-t}}\right)^{x} = \sum_{n\geqslant 0} S_n(x)\frac{t^n}{n!},$$

где

$$S_n(x) = \sum_{i=0}^{n} \binom{x+i}{i} \sum_{j=0}^{i} \frac{j!}{(n+j)!}(-1)^{n+j}\binom{i}{j}\left\{ {n+j \atop j} \right\}.$$

Согласно (2.55) композита производящей функции $D(t) = tS(t,x)$ равна

$$D^{\Delta}(n,k) = S_{n-k}(kx+k-1).$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tS(t,x)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n} S_{n-k}(-nx-n-1).$$

Также можно получить следующие новые тождества для полиномов Стирлинга:

$$\sum_{m=k}^{n} \frac{m}{n} S_{n-m}(-nx-n-1) S_{m-k}(kx+k-1) = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} S_{n-m}(mx+m-1) \frac{k}{m} S_{m-k}(-mx-m-1) = \delta_{n,k}.$$

Полиномы Наруми определяются следующей производящей функцией [52]:

$$S(t,x,\alpha) = \left(\frac{t}{\ln(1+t)}\right)^{\alpha} (1+t)^{x} = \sum_{n \geqslant 0} S_n(x,\alpha) \frac{t^n}{n!},$$

где

$$S_n(x,\alpha) = n! \sum_{i=0}^{n} \binom{x}{n-i} \sum_{j=0}^{i} \binom{j+\alpha-1}{j} \sum_{l=0}^{j} (-1)^l \binom{j}{l} \frac{l!}{(l+i)!} \begin{bmatrix} l+i \\ l \end{bmatrix}.$$

Согласно (2.55) композита производящей функции $D(t) = tS(t,x,\alpha)$ равна

$$D^{\Delta}(n,k) = \frac{S_{n-k}(kx,k\alpha)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tS(t,x,\alpha)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n} \frac{S_{n-k}(-nx,-n\alpha)}{(n-k)!}.$$

Также можно получить следующие новые тождества для полиномов Наруми:

$$\sum_{m=k}^{n} \frac{m}{n} \frac{S_{n-m}(-nx,-n\alpha)}{(n-m)!} \frac{S_{m-k}(kx,k\alpha)}{(m-k)!} = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} \frac{S_{n-m}(mx,m\alpha)}{(n-m)!} \frac{k}{m} \frac{S_{m-k}(-mx,-m\alpha)}{(m-k)!} = \delta_{n,k}.$$

Полиномы Петерса определяются следующей производящей функцией [52]:

$$S(t,x,\mu,\lambda)=(1+t)^x\left(\frac{1}{1+(1+t)^\lambda}\right)^\mu=\sum_{n\geqslant 0}S_n(x,\mu,\lambda)\frac{t^n}{n!},$$

где

$$S_n(x,\mu,\lambda)=n!\sum_{i=0}^{n}\binom{x}{n-i}\sum_{j=0}^{i}\frac{1}{2^{j+\mu}}\binom{j+\mu-1}{j}\sum_{l=0}^{j}(-1)^l\binom{j}{l}\binom{l\lambda}{i}.$$

Согласно (2.55) композита производящей функции $D(t)=tS(t,x,\mu,\lambda)$ равна

$$D^{\Delta}(n,k)=\frac{S_{n-k}(kx,k\mu,\lambda)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t)=tS(t,x,\mu,\lambda)$ равна

$$\overline{D}^{\Delta}(n,k)=\frac{k}{n}\frac{S_{n-k}(-nx,-n\mu,\lambda)}{(n-k)!}.$$

Также можно получить следующие новые тождества для полиномов Петерса:

$$\sum_{m=k}^{n}\frac{m}{n}\frac{S_{n-m}(-nx,-n\mu,\lambda)}{(n-m)!}\frac{S_{m-k}(kx,k\mu,\lambda)}{(m-k)!}=\delta_{n,k}$$

и

$$\sum_{m=k}^{n}\frac{S_{n-m}(mx,m\mu,\lambda)}{(n-m)!}\frac{k}{m}\frac{S_{m-k}(-mx,-m\mu,\lambda)}{(m-k)!}=\delta_{n,k}.$$

Полиномы Гегенбауэра определяются следующей производящей функцией [133]:

$$C(t,x,\alpha)=(1-2xt+t^2)^{-\alpha}=\left(\frac{1}{1-2xt+t^2}\right)^{\alpha}=\sum_{n\geqslant 0}C_n^{(\alpha)}(x)t^n,$$

где

$$C_n^{(\alpha)}(x)=\sum_{i=0}^{n}(-1)^{n-i}\binom{i}{n-i}\binom{i+\alpha-1}{i}(2x)^{2i-n}.$$

Согласно (2.55) композита производящей функции $D(t)=tC(t,x,\alpha)$ равна

$$D^{\Delta}(n,k)=C_{n-k}^{(k\alpha)}(x).$$

Согласно (2.57) композита для обратной производящей функции $\overline{D}(t)$ для $D(t) = tC(t,x,\alpha)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n} C_{n-k}^{(-n\alpha)}(x).$$

Также можно получить следующие новые тождества для полиномов Гегенбауэра:

$$\sum_{m=k}^{n} \frac{m}{n} C_{n-m}^{(-n\alpha)}(x) C_{m-k}^{(k\alpha)}(x) = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} C_{n-m}^{(m\alpha)}(x) \frac{k}{m} C_{m-k}^{(-m\alpha)}(x) = \delta_{n,k}.$$

Полиномы Мейкснера первого рода определяются следующей производящей функцией [52; 134]:

$$M(t,x,\beta,c) = \left(\frac{c-t}{c(1-t)}\right)^{x} \left(\frac{1}{1-t}\right)^{\beta} = \sum_{n\geqslant 0} M_n(x,\beta,c)\frac{t^n}{n!},$$

где

$$M_n(x,\beta,c) = (-1)^n n! \sum_{i=0}^{n} \binom{x}{i} \binom{-x-\beta}{n-i} c^{-i}.$$

Согласно (2.55) композита производящей функции $D(t) = tM(t,x,\beta,c)$ равна

$$D^{\Delta}(n,k) = \frac{M_{n-k}(kx,k\beta,c)}{(n-k)!}.$$

Согласно (2.57) композита обратной производящей функции $\overline{D}(t)$ для $D(t) = tM(t,x,\beta,c)$ равна

$$\overline{D}^{\Delta}(n,k) = \frac{k}{n} \frac{M_{n-k}(-nx,-n\beta,c)}{(n-k)!}.$$

Также можно получить следующие новые тождества для полиномов Мейкснера первого рода:

$$\sum_{m=k}^{n} \frac{m}{n} \frac{M_{n-m}(-nx,-n\beta,c)}{(n-m)!} \frac{M_{m-k}(kx,k\beta,c)}{(m-k)!} = \delta_{n,k}$$

и

$$\sum_{m=k}^{n} \frac{M_{n-m}(mx,m\beta,c)}{(n-m)!} \frac{k}{m} \frac{M_{m-k}(-mx,-m\beta,c)}{(m-k)!} = \delta_{n,k}.$$

2.6.4 Обобщенное тождество Теппера и его применение

М. Теппером в работе [135] предложено следующее тождество, которое было доказано К. Лонгом [136] и Ф. Паппом [137]:

$$\sum_{j=0}^{n}(-1)^j\binom{n}{j}(x-j)^n = n!, \tag{2.58}$$

где n — положительное целое число, x — любое действительное число.

Для случая, когда $x = 0$, данное тождество принимает вид известной формулы Эйлера. Г.У. Гоулд [138] представил детальное описание данной формулы Эйлера. Используя теорию разностных операторов, он предложил следующую модификацию формулы (2.58):

$$\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}(A+Bj)^p = \begin{cases}0, & 0 \leqslant p < n;\\ B^n n!, & p = n,\end{cases}$$

где A и B являются константами.

Существует большое количество разных исследований относительно обобщений тождества Теппера. Например, Г.П. Егорычев [10] получил еще одно доказательство тождества. Схожие результаты были получены М. Баятом и Х. Теймури [139;140]. Также ими было предложено еще одно обобщение тождества Теппера [141]: для любого положительного n и вещественных x и λ справедливо

$$\sum_{k=0}^{n}(-1)^{n-k}\binom{n}{k}(x+k)^{n|\lambda} = n!, \tag{2.59}$$

где $x^{n|\lambda}$ определено как:

$$x^{n|\lambda} = \begin{cases}x(x+\lambda)\cdots(x+(n-1)\lambda), & n > 0;\\ 1, & n = 0.\end{cases}$$

Для $\lambda = 0$ тождество (2.59) примет вид (2.58). Схожий результат получили К. Чжао и Т. Ван [142].

Также существуют обобщения тождества Теппера, включающие полиномы. Например, для любого полинома $f(x)$ степени m, справедливо [139;141;143;144]

$$\sum_{k=0}^{n}(-1)^k\binom{n}{k}f(x+k) = (-1)^m a_m m!\delta_{n,m},$$

где a_m — старший коэффициент в полиноме $f(x)$. Схожие результаты можно найти в [145]:

Рассмотрим вариант доказательства тождества Теппера (2.58) с помощью метода теории производящих функций и полиномов Бернулли, обобщим и приведем доказательство формулы Гоулда для любого значения переменной x.

Полиномы Бернулли $B_n(x)$ определяются производящей функцией [52]

$$A(t,x) = \frac{t}{e^t - 1} e^{xt} = \sum_{n \geqslant 0} B_n(x) \frac{t^n}{n!}.$$

Пусть производящая функция $V(t,x)$ будет обратной для производящей функцией полиномов Бернулли:

$$V(t,x) = \frac{t}{A(t,x)}. \tag{2.60}$$

Предлагаемая идея заключается в рассмотрении функции вида

$$V(t,x) = (e^t - 1)e^{-xt}$$

и разложении данной функции двумя разными способами, что в последствии позволит приравнять коэффициенты при степенях переменной t. Таким образом, будет получено искомое тождество.

Производящую функцию (2.60) можно представить в виде следующего формального степенного ряда:

$$V(t,x) = \sum_{n=0}^{\infty} y_n(x) \frac{t^n}{n!}.$$

Тогда для коэффициентов $y_n(x)$ данной производящей функции может быть получена явная формула

$$y_n(x) = (1 - x)^n + (-1)^{n+1} x^n.$$

Рассмотрим k-ю степень производящей функции (2.60)

$$V(t,x)^k = \sum_{n \geqslant k} v(n,k) t^n. \tag{2.61}$$

Используя (2.60) и (2.61), получаем следующую производящую функцию:

$$(e^{-xt+t} - e^{-xt})^k = \sum_{n \geqslant k} K(x; n,k) \frac{t^n}{n!}, \tag{2.62}$$

где

$$K(x;n,k) = \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{j} (j-kx)^n.$$

С другой стороны, можно получить следующее выражение:

$$(e^t-1)^k e^{-kxt} = \sum_{n \geqslant k} \left(\sum_{i=k}^{n} \left\{ {i \atop k} \right\} \frac{k!}{i!} \frac{(-kx)^{n-i}}{(n-i)!} \right) t^n. \tag{2.63}$$

Объединяя полученные результаты в (2.62) и (2.63), мы приходим к следующей теореме:

Теорема 20. *Справедливо следующее выражение:*

$$K(x;n,k) = k! \sum_{i=k}^{n} \left\{ {i \atop k} \right\} \binom{n}{i} (-kx)^{n-i}, \tag{2.64}$$

где $\left\{ {i \atop k} \right\}$ *обозначает числа Стирлинга второго рода.*

Подставляя $k = n$ в (2.64), получаем

$$K(x;n,n) = n!.$$

Так как правая часть полученного выражения не зависит от x, то мы получаем тождество Теппера (2.58).

Далее рассмотрим модификацию тождества Теппера, основанную на следующей теореме:

Теорема 21. *Для любого значения переменной* x *выполняется следующее:*

$$T(n,k,x) = \frac{1}{n!} \sum_{j=0}^{n} (-1)^{n-j} \binom{n}{j} (j+x)^{n+k} \tag{2.65}$$

или

$$T(n,k,x) = \sum_{j=0}^{k} \binom{n+k}{j} \left\{ {n+k-j \atop n} \right\} x^j.$$

Доказательство:

Проинтегрировав тождество Теппера (2.58) по переменной x, получим

$$\frac{1}{(n+1)!} \sum_{j=0}^{n} (-1)^{n-j} \binom{n}{j} (j+x)^{n+1} = x + C.$$

Для нахождения значения C, подставим $x = 0$ в полученное выражение

$$C = \frac{1}{(n+1)!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}j^{n+1}.$$

Так как

$$\left\{ n+k \atop n \right\} = \frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}j^{n+k},$$

тогда получаем

$$C = \frac{1}{n+1}\left\{ n+1 \atop n \right\} = \frac{n}{2}.$$

Следовательно,

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}(j+x)^{n+1} = (n+1)x + \frac{n(n+1)}{2}.$$

Если повторить данные действия по интегрированию, то для $k = 1$ получим

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}(x+j)^{n+2} =$$

$$= \frac{(n+2)(n+1)}{2}x^2 + \frac{(n+2)(n+1)n}{2}x + \left\{ n+2 \atop n \right\}.$$

Получим выражение для общего случая интегрирования k раз

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}(j+x)^{n+k} = \sum_{j=0}^{k}\binom{n+k}{j}\left\{ n+k-j \atop n \right\}x^j.$$

Следовательно, теорема доказана. □

Из выражения (2.65) можно отметить, что $T(n,k,x)$ является полиномом степени k. Если взять производную от этих полиномов по переменной x, тогда получим следующую дифференциальную формулу:

$$\frac{d}{dx}T(n,k,x) = (n+k)T(n,k-1,x).$$

Далее рассмотрим применение полученного обобщения для получения новых тождеств для чисел и полиномов Бернулли и Эйлера.

Полиномы Эйлера $E_n(x)$ определяются следующей производящей функцией:

$$\frac{2e^{xt}}{1+e^t} = \sum_{n\geqslant 0} E_n(x)\frac{t^n}{n!}.$$

Известно, что для полинома $P(x)$ можно задать два символьных оператора, которые основываются на свойствах полиномов Бернулли и Эйлера [146;147]. Пусть

$$B_n(x+h) = (B(x)+h)^n,$$

$$E_n(x+h) = (E(x)+h)^n,$$

где $B(x)^n$ и $E(x)^n$ заменены условно на $B_n(x)$ и $E_n(x)$ соответственно, и

$$P(B(x)+1) - P(B(x)) = P'(x),$$

$$P(E(x)+1) + P(E(x)) = 2P(x).$$

Так как

$$P(x+h) = R(x),$$

где $P(x)$ и $R(x)$ являются полиномами от переменной x, тогда:

$$P(B(x)+h) = R(B(x)),$$

$$P(E(x)+h) = R(E(x)).$$

Далее, применяя символьные операторы, мы приходим к следующим тождествам, которые содержат полиномы Бернулли и Эйлера:

Теорема 22. *Для полиномов Бернулли и Эйлера выполняются тождества:*

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}B_{n+k}(j+x) = \sum_{j=0}^{k}\binom{n+k}{j}\left\{{n+k-j \atop n}\right\}B_j(x), \qquad (2.66)$$

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}E_{n+k}(j+x) = \sum_{j=0}^{k}\binom{n+k}{j}\left\{{n+k-j \atop n}\right\}E_j(x). \qquad (2.67)$$

Доказательство:

Применяем оператор $P(B(x+h)) = R(B(x))$ для обобщенного тождества Теппера (2.65) и полиномов Бернулли, в результате получаем (2.66).

Аналогично, получаем выражение (2.67) для полиномов Эйлера. □

Для полиномов Бернулли тождество Теппера (2.65) при $k = 0$ принимает следующий вид:

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}B_n(j+x)=1.$$

Для полиномов Эйлера тождество Теппера (2.65) при $k = 0$ принимает следующий вид:

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}E_n(j+x)=1.$$

Формула (2.65) позволяет получать явные выражения для полиномов Бернулли и Эйлера через числа Стирлинга второго рода. Применяя (2.65) к известным результатам

$$B_n(x)=\sum_{k=0}^{n}\frac{1}{k+1}\sum_{j=0}^{k}(-1)^j\binom{k}{j}(j+x)^n$$

и

$$E_n(x)=\sum_{k=0}^{n}\frac{1}{2^k}\sum_{j=0}^{k}(-1)^j\binom{k}{j}(j+x)^n,$$

получаем следующие представления полиномов: [148; 149]

$$B_n(x)=\sum_{k=0}^{n}\frac{(-1)^k k!}{k+1}\sum_{j=0}^{n-k}\binom{n}{j}\left\{\begin{matrix}n-j\\k\end{matrix}\right\}x^j$$

и

$$E_n(x)=\sum_{k=0}^{n}\frac{(-1)^k k!}{2^k}\sum_{j=0}^{n-k}\binom{n}{j}\left\{\begin{matrix}n-j\\k\end{matrix}\right\}x^j.$$

Теорема 23. *Пусть n и k — неотрицательные целые числа. Тогда*

$$\sum_{j=0}^{k}\binom{n+k}{j}\left\{\begin{matrix}n+k-j\\n\end{matrix}\right\}B_j=\frac{n+k}{n!}\sum_{j=0}^{n}\left(\sum_{m=1}^{j-1}m^{n+k-1}\right)(-1)^{n-j}\binom{n}{j}.$$

Доказательство:

Известно [150], что

$$B_n(j)=B_n+n\sum_{m=1}^{j-1}m^{n-1}.$$

Запишем формулу обобщенного тождества Теппера для полиномов Бернулли (2.66) для $x = 0$:

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}B_{n+k}(j) = \sum_{j=0}^{k}\binom{n+k}{j}\left\{ {n+k-j \atop n} \right\}B_j.$$

Подставим значение $B_{n+k}(j)$ в приведенную выше формулу, так как

$$\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}B_{n+k} = 0,$$

то получаем искомый результат. □

Тождество (2.66) также можно расширить для обобщенных полиномов Бернулли. Обобщенные полиномы Бернулли $B_n^{\alpha}(x)$ определяются следующей производящей функцией для произвольного значения параметра α [42]:

$$e^{xt}\left(\frac{t}{e^t-1}\right)^{\alpha} = \sum_{n\geqslant 0}B_n^{\alpha}(x)\frac{t^n}{n!}.$$

Теорема 24. *Пусть n и k — неотрицательные целые числа. Тогда*

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}B_{n+k}^{\alpha}(j+x) = \sum_{j=0}^{k}\binom{n+k}{j}\left\{ {n+k-j \atop n} \right\}B_j^{\alpha}(x).$$

Доказательство:

Для обобщенных полиномов Бернулли выполняется известное соотношение

$$B_n^{\alpha}(x+h) = \sum_{k=0}^{n}\binom{n}{k}B_k^{\alpha}(x)h^{n-k}.$$

Это означает, что указанные ранее два символьных оператора могут быть применены для обобщенных полиномов Бернулли. Следовательно, рассматриваемое тождество является истинной. □

Тождество (2.67) можно расширить для полиномов Эйлера-Фробениуса порядка α. Полиномы Эйлера-Фробениуса порядка α определяются следующей производящей функцией [36; 151]:

$$\left(\frac{1-\lambda}{e^t-\lambda}\right)^{\alpha}e^{xt} = \sum_{n\geqslant 0}H_n^{\alpha,\lambda}(x)\frac{t^n}{n!},$$

где λ — произвольный параметр и $\lambda \neq 1$, α — положительное целое число. Заметим, что [36]

$$E_n(x) = H_n^{1,-1}(x).$$

Теорема 25. *Для полиномов Эйлера-Фробениуса* $H_n^{\alpha,\lambda}(x)$ *порядка* α *выполняется следующее:*

$$\frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{n}{j}H_{n+k}^{\alpha,\lambda}(j+x)=\sum_{j=0}^{k}\binom{n+k}{j}\left\{{n+k-j \atop n}\right\}H_j^{\alpha,\lambda}(x).$$

Доказательство:

Для полиномов Эйлера-Фробениуса выполняется следующее известное соотношение [152]:

$$H_n^{\alpha,\lambda}(x+h)=\sum_{k=0}^{n}\binom{n}{k}H_k^{\alpha,\lambda}(x)h^{n-k}.$$

Это означает, что указанные ранее два символьных оператора могут быть применены для обобщенных полиномов Эйлера-Фробениуса. Следовательно, рассматриваемое тождество является истинным. □

2.7 Выводы по главе

В данной главе изложены методы получения коэффициентов степеней производящих функций для случая одномерных, двумерных и трехмерных производящих функций, а также для n-мерных рациональных производящих функций. Рассмотрены правила преобразования коэффициентов степеней взаимных, обратных и композиции производящих функций многих переменных, что позволяет получить их явные представления. За счет разработанного математического исчисления над коэффициентами степеней производящих функций многих переменных представлен комплексный метод формирования информационных объектов. Комплексный метод состоит из совокупности методов и правил для оперирования производящими функциями одной, двух и трех переменных, а также n-мерных рациональных производящих функций. С точки зрения приложения разработанных подходов найдены явные выражения для ряда известных последовательностей онлайн-энциклопедии целочисленных последовательностей и соответствующих им информационных объектов.

Рассмотрен вопрос применения предложенного комплексного метода для формирования и описания информационных объектах на следующих примерах: специальные числа и полиномы, числовые треугольники. Применение предложенного комплексного метода позволяет для информационных объектов, описываемых производящими функциями, получать явные представления, обладающие меньшей вычислительной сложностью по сравнению с вычислением коэффициентов на основе формул, построенных на разбиениях.

Решена задача нахождения производящих функций центральных элементов треугольника $T_{n,k}$, заданного степенями производящей функции $(xH(x))^k = \sum\limits_{n \geqslant k} T_{n,k}\, x^n$, где $H(x)$ — производящая функция с $h(0) \neq 0$. Решена обратная задача: нахождение вида треугольника по производящей функции центральных коэффициентов. Также для заданных треугольников решена задача нахождения производящих функций для диагонали $T_{2n,n}$. На основе полученных результатов видна однозначная связь между центральными коэффициентами и всем числовым треугольником. Поэтому, зная лишь производящую функцию центральных коэффициентов, можно восстановить весь числовой треугольник и, наоборот, представить весь числовой треугольник в виде его центральных коэффициентов.

Кроме того, полученные результаты дополняют имеющиеся методы в теории производящих функций и теории полиномов на предмет эффективного определения явных представлений для соответствующих полиномов и других информационных объектов, а также позволяют значительно расширить область применения методов комбинаторной генерации, что отражено в следующей главе.

Результаты данной главы опубликованы в следующих публикациях [68; 89; 106; 134; 153–169].

Глава 3. Метод построения алгоритмов комбинаторной генерации с использованием производящих функций многих переменных

В данной главе представлены основные результаты в области разработки методов и алгоритмов комбинаторной генерации. Предложена модификация метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ с учетом описания функции мощности множества производящей функцией многих переменных и применения приближенных вычислений и двоичного поиска для поиска выбранного сына ИЛИ-узла. Рассмотрено применение разработанного комплексного метода получения явных выражений коэффициентов производящих функций многих переменных для нахождения выражения функции мощности комбинаторного множества. Предложенный модифицированный метод применен для разработки алгоритмов комбинаторной генерации для формирования информационных объектов, представленных следующими комбинаторными множествами: сочетания элементов множества; решеточные пути на плоскости; пути Дика с пиками; последовательности вариантов ответа на тест с вопросами закрытого типа; исходы турнира на выбывание; части круга, полученные при разрезе его поверхности прямыми линиями; последовательности правильно вложенных скобок, разряженных нулями; разбиения множества.

3.1 Подходы к построению алгоритмов комбинаторной генерации

Структура многих информационных объектов может быть представлена в виде иерархической или рекурсивной зависимости. Для представления и кодирования таких информационных объектов достаточно хорошо подходит применение древовидных структур данных. В свою очередь, это приводит к возможности описания исследуемого информационного объекта с помощью формального комбинаторного множества, для которого применимы различного рода алгоритмы комбинаторной генерации. Комбинаторное множество — это конечное множество, элементы которого имеют некоторую структуру и имеется процедура построения элементов этого множества [170]. Комбинаторная генерация — раздел науки на стыке информатики, комбинаторики и дискретной математики, в рамках которого исследуются методы и алгоритмы генерации элементов комбинаторных множеств.

В своей фундаментальной монографии в области теоретических основ информатики Д.Э. Кнут [171] посвятил целый том обзорному исследованию процесса становления и развития научного направления, связанного с разработкой и применением комбинаторных алгоритмов. В указанной работе особому вниманию представлена задача обхода всех элементов заданного комбинаторного множества, которая может быть рассмотрена как перечисление (enumerating), составление списка (listing) и генерация (generating) элементов комбинаторного множества.

С другой стороны, Ф. Раски [172] рассматривает комбинаторную генерацию как отдельное научное направление, в рамках которого выделяет следующие классы решаемых задач:

1. Listing: последовательная генерация элементов рассматриваемого комбинаторного множества;

2. Ranking: ранжирование (нумерация) элементов рассматриваемого комбинаторного множества;

3. Unranking: генерация элементов рассматриваемого комбинаторного множества в соответствии с их рангами (номерами);

4. Random selection: генерация элементов рассматриваемого комбинаторного множества в случайном порядке.

Общие и универсальные методы, направленные на разработку новых алгоритмов комбинаторной генерации для широкого класса комбинаторных множеств, исследовались такими учеными, как Э.М. Рейнгольд [173], Д.Л. Крехер [174], Е. Баркуччи [175;176], С. Баччелли [177;178], А. Дель Лунго [179;180], В. Вайновски [181–183], Ф. Флажоле [184; 185], К. Мартинес и К. Мулинеро [186–190], Б.Я. Рябко и Ю.С. Медведева [191–193], В.В. Кручинин и Ю.В. Шабля [170; 194; 195], и другие.

Проведенный аналитический обзор имеющихся на сегодняшний день подходов к разработке новых алгоритмов комбинаторной генерации показал, что можно выделить следующий перечень методов, характеризующихся некоторой степенью универсальности их применения:

– метод поиска с возвратом (backtracking) [173; 174]: данный метод может быть применен только для последовательной генерации комбинаторных объектов. Метод основан на построении генерирующего дерева для заданного комбинаторного множества и получении допустимых решений на уровне l данного генерирующего дерева за счет расширения имеющегося допустимого решения на уровне $l-1$;

– ECO-метод [175–183]: данный метод может быть применен только для последовательной генерации комбинаторных объектов. Метод основан на получении множества новых объектов размера $n+1$ за счет применения ECO-правила наследования из объектов размера n;

– метод Ф. Флажоле [184–190]: данный метод может быть применен для последовательной генерации, а также ранжирования и генерации по рангу комбинаторных объектов. Метод основан на представлении обыкновенной или экспоненциальной производящей функции одной переменной для функции мощности комбинаторного множества через комбинацию допустимых комбинаторных операторов;

– метод Б.Я. Рябко [191–193]: данный метод может быть применен для ранжирования и генерации по рангу комбинаторных объектов. Метод основан на представлении комбинаторного объекта в форме слова, с последующей обработкой префиксов данного слова и применении парадигмы «Разделяй и властвуй»;

– метод на основе деревьев И/ИЛИ [170]: данный метод может быть применен для последовательной генерации, а также ранжирования и генерации по рангу комбинаторных объектов. Метод основан на представлении комбинаторного множества в форме структуры дерева И/ИЛИ количество вариантов которого совпадает со значением функции мощности.

Каждый из представленных методов характеризуется универсальностью их применения, что позволяет использовать их при построении совершенно новых алгоритмов комбинаторной генерации. Однако существует целый перечень требований и ограничений к применимости таких методов [196]:

– основной характеристикой, необходимой для разработки алгоритмов комбинаторной генерации, является функция мощности комбинаторного множества;

– часть методов (поиск с возвратом и ECO-метод) направлены только на разработку алгоритмов последовательной генерации;

– часть методов (ECO-метод и метод Флажоле) не могут быть применены для комбинаторных множеств, определяемых более чем одним параметром;

– большинство методов требуют представления комбинаторного объекта в специальной форме (например, в форме слова, последовательности, дерева).

Также при разработке алгоритмов комбинаторной генерации могут требоваться дополнительные условия, например, в виде наличия дополнительной информации, описывающей исследуемое комбинаторное множество. Кроме того, существует множество алгоритмов комбинаторной генерации, получение

которых основано на особенностях исследуемого комбинаторного множества либо на простых методах подсчета [197–199]. Поэтому методы, используемые для разработки таких алгоритмов, не могут быть универсальными (их нельзя применять для разработки новых комбинаторных алгоритмов генерации других комбинаторных наборов).

Таким образом, на сегодняшний день не существует универсального общего метода, который может быть применен для разработки новых комбинаторных алгоритмов генерации для любого заданного комбинаторного множества. Следовательно, возникает потребность в разработке новых и совершенствовании существующих методов разработки алгоритмов комбинаторной генерации. В данной главе рассматривается задача модификации и расширения возможностей применения метода для построения алгоритмов комбинаторной генерации, который основан на использовании деревьев И/ИЛИ, так как оригинальный метод:

– позволяет разрабатывать все типы алгоритмов комбинаторной генерации (алгоритмы последовательной генерации, а также ранжирования и генерации по рангу комбинаторных объектов);

– не имеет ограничений на количество параметров, определяющих комбинаторное множество;

– в качестве дополнительной информации требует только знание выражения функции мощности комбинаторного множества.

Однако для применения данного метода необходимо знать выражение функции мощности комбинаторного множества, которое должно удовлетворять только одному или нескольким следующим условиям:

– выражение функции мощности содержит в своей записи только натуральные числа (обозначим через $\mathbb{N}$ множество натуральных чисел);

– выражение функции мощности содержит в своей записи только такие алгебраические операции, как сложение (обозначим через $+$) и умножение (обозначим через $\times$);

– функция мощности определена рекурсивно (пусть R обозначает примитивно-рекурсивную функцию).

Если выражение функции мощности f комбинаторного множества A удовлетворяет приведенным выше условиям, то будем говорить, что f принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$. Требование наличия выражения функции мощности, принадлежащей алгебре $\{\mathbb{N}, +, \times, R\}$, является основным ограничением выбранного для исследования метода построения алгоритмов комбинаторной генерации. Если

требуемый вид выражения функции мощности для заданного комбинаторного множества неизвестен, то данный метод не может быть применен для разработки новых алгоритмов комбинаторной генерации.

В рамках диссертационного исследования предлагается модифицирование оригинального метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ за счет применения разработанного комплексного метода получения коэффициентов производящих функций многих переменных для решения задач поиска функций мощности комбинаторных множеств, а также за счет применения различных методов поиска для повышения быстродействия алгоритмов генерации.

3.2 Модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ

В работах [170; 194] рассматривается метод построения алгоритмов комбинаторной генерации, который основан на представлении исследуемого комбинаторного множества в форме структуры дерева И/ИЛИ количество вариантов которого совпадает со значением функции мощности. Данный метод прошел апробацию и показал возможности его использования при разработке новых алгоритмов последовательной генерации, ранжирования и генерации по рангу для широкого круга комбинаторных множеств.

Деревом И/ИЛИ называется дерево, которое содержит узлы двух типов: И-узел и ИЛИ-узел. На рисунке 3.1 представлено схематическое изображение узлов дерева И/ИЛИ. Вариантом дерева И/ИЛИ называется дерево, которое получается из данного путем отсечения у всех ИЛИ-узлов всех дуг кроме одной.

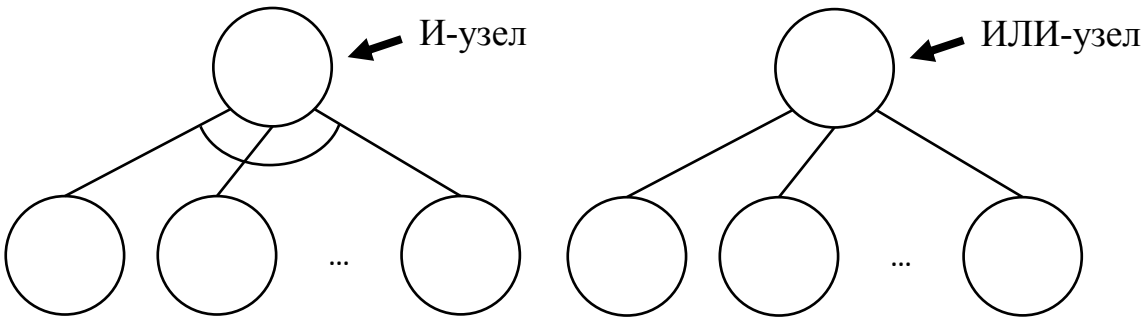


Рисунок 3.1 — Схематическое изображение узлов дерева И/ИЛИ

Если имеется информация о функции мощности f комбинаторного множества A, которая принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то можно построить структуру дерева И/ИЛИ количество вариантов которого совпадает со значением функции мощности [170]. Чтобы построить такую структуру дерева И/ИЛИ необходимо выполнить следующие действия для заданной функции мощности f:

– каждая операция сложения $+$ из f представляется ИЛИ-узлом дерева И/ИЛИ, все слагаемые записываются как сыновья данного ИЛИ-узла;

– каждая операция умножения $\times$ из f представляется И-узлом дерева И/ИЛИ, все множители записываются как сыновья данного И-узла;

– каждый коэффициент $k \in \mathbb{N}$ из f представляется ИЛИ-узлом дерева И/ИЛИ, который содержит k листьев;

– каждая рекурсивная операция из f представляется в виде схемы рекурсивной композиции дерева И/ИЛИ.

Таким образом, метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ можно записать в следующем виде [195; 196]:

Вход: Функция мощности комбинаторного множества $f \in \{\mathbb{N}, +, \times, R\}$.

Выход: Алгоритмы комбинаторной генерации

$$\texttt{RankVariant} : W(D) \to \mathbb{N}_{|W(D)|},$$

$$\texttt{UnrankVariant} : \mathbb{N}_{|W(D)|} \to W(D),$$

при этом каждый вариант v дерева И/ИЛИ D, построенного для некоторого комбинаторного множества A, должен однозначно соответствовать конкретному комбинаторному объекту $a \in A$, то есть должна быть определена биекция $A \leftrightarrow W(D)$, где $W(D)$ — множество всех вариантов v дерева И/ИЛИ D. Совокупность данной биекции с алгоритмами `RankVariant` и `UnrankVariant` представляют собой искомые алгоритмы комбинаторной генерации $\texttt{Rank}(a) : A \to \mathbb{N}$ и $\texttt{Unrank}(r) : \mathbb{N} \to A$.

Кроме того, возможна реализация алгоритма последовательной генерации за счет выполнения последовательного обхода всех вариантов дерева И/ИЛИ по аналогии с левосторонним обходом древовидных структур.

В основе алгоритмов `RankVariant` и `UnrankVariant` заложены правила вычисления для каждого варианта $v \in W(D)$ дерева И/ИЛИ некоторого уникального числа $r \in \mathbb{N}_{|W(D)|} = \{0, 1, \ldots, |W(D)| - 1\}$, называемого рангом. Для вычисления ранга r необходимо вычислить значение $l(root)$, при этом функция $l(z)$ задается следующим набором правил:

1. Если z является листом дерева И/ИЛИ, то $l(z) = 0$;
2. Если z является И-узлом дерева И/ИЛИ, то

$$l(z) = l(s_1^{(z)}) + w(s_1^{(z)}) \left(l(s_2^{(z)}) + w(s_2^{(z)}) \left(\ldots \left(l(s_{n-1}^{(z)}) + w(s_{n-1}^{(z)}) l(s_n^{(z)}) \right) \ldots \right) \right),$$

где n — количество сыновей узла z, $s_i^{(z)}$ — i-й сын узла z, $w(z)$ — количество вариантов в поддереве узла z;

3. Если z является ИЛИ-узлом дерева И/ИЛИ, то

$$l(z) = l(s_k^{(z)}) + \sum_{i=1}^{k-1} w(s_i^{(z)}),$$

где k — позиция выбранного в варианте дерева И/ИЛИ сына $s_k^{(z)}$ ИЛИ-узла z среди всех его сыновей.

Совокупность правил вычисления значения $l(z)$ представлена в виде алгоритма `RankVariant`(z, v, D) (Алгоритм 1).

Алгоритм 1: Общий алгоритм ранжирования варианта дерева И/ИЛИ

```
1  RankVariant (z, v, D)
2  begin
3      if z = лист дерева И/ИЛИ D then l := 0
4      if z = И-узел дерева И/ИЛИ D then
5          n := количество сыновей узла z
6          l := RankVariant (s_n^(z), v, D)
7          for i := n − 1 to 1 do l := RankVariant (s_i^(z), v, D) + w(s_i^(z)) · l
8      end
9      if z = ИЛИ-узел дерева И/ИЛИ D then
10         k := позиция выбранного в варианте дерева И/ИЛИ сына
             ИЛИ-узла z среди всех его сыновей
11         l := RankVariant (s_k^(z), v, D)
12         for i := 1 to k − 1 do l := l + w(s_i^(z))
13     end
14     return l
15 end
```

Совокупность правил восстановления варианта v дерева И/ИЛИ D по его рангу r представлена в виде алгоритма `UnrankVariant`(r, D) (Алгоритм 2).

Алгоритм 2: Общий алгоритм генерации по рангу варианта дерева И/ИЛИ

```
1  UnrankVariant (r, D)
2  begin
3      Поместить в стек пару (r, root)
4      Записать root в вариант v
5      while стек не пуст do
6          Вытащить из стека пару (l, z)
7          if z = И-узел дерева И/ИЛИ D then
8              n := количество сыновей узла z
9              for i := 1 to n do
10                 l_i := l  mod w(s_i^(z))
11                 Поместить в стек пару (l_i, s_i^(z))
12                 Записать s_i^(z) в вариант v
13                 l := ⌊ l / w(s_i^(z)) ⌋
14             end
15         end
16         if z = ИЛИ-узел дерева И/ИЛИ D then
17             k := 1
18             sum := 0
19             while sum + w(s_k^(z)) ⩽ l do
20                 sum := sum + w(s_k^(z))
21                 k := k + 1
22             end
23             l := l − sum
24             Поместить в стек пару (l, s_k^(z))
25             Записать s_k^(z) в вариант v
26         end
27     end
28     return v
29 end
```

Если для исследуемого комбинаторного множества A не известно выражение функции мощности f, принадлежащей алгебре $\{\mathbb{N}, +, \times, R\}$, то данный метод на основе деревьев И/ИЛИ не может быть применен для разработки алгоритмов комбинаторной генерации. Для решения данной проблемы предлагается применение математического аппарата производящих функций, а именно: применение разработанного комплексного метода получения коэффициентов производящих функций многих переменных.

Учитывая эффективность применения производящих функций для описания дискретных структур, рассмотрим комбинаторное множество A_n, которое определяется одним параметром n (например, этот параметр может характеризовать размерность каждого объекта данного комбинаторного множества). Тогда функция мощности $f(n) = |A_n|$ может быть определена через обыкновенную производящую функцию

$$F(x) = \sum_{n \geqslant 0} f(n)x^n = \sum_{n \geqslant 0} |A_n| x^n.$$

Если комбинаторное множество определяется набором из нескольких параметров $n, m, \ldots, l$, тогда функция мощности $f(n,m,\ldots,l) = |A_{n,m,\ldots,l}|$ может быть определена через производящую функцию многих переменных

$$F(x,y,\ldots,z) = \sum_{n \geqslant 0} \sum_{m \geqslant 0} \cdots \sum_{l \geqslant 0} f(n,m,\ldots,l) x^n y^m \cdots z^l =$$

$$= \sum_{n \geqslant 0} \sum_{m \geqslant 0} \cdots \sum_{l \geqslant 0} |A_{n,m,\ldots,l}| x^n y^m \cdots z^l.$$

Таким образом, если для исследуемого комбинаторного множества известна производящая функция (в том числе производящая функция многих переменных), которая определяет значения функции мощности данного комбинаторного множества, то коэффициенты производящей функции определяют значения самой функции мощности. Следовательно, получив явное выражение $f(n,m,\ldots,l)$ для коэффициентов производящей функции, становится известным явное выражение $|A_{n,m,\ldots,l}|$ для функции мощности соответствующего комбинаторного множества. Подходящим средством для решения данной задачи является разработанный комплексный метод получения коэффициентов производящих функций многих переменных.

Анализ общего алгоритма генерации по рангу варианта дерева И/ИЛИ показывает, что часть его вычислений направлена лишь на поиск позиции k выбранного в варианте дерева И/ИЛИ сына $s_k^{(z)}$ ИЛИ-узла z среди всех его сыновей (строки 17–22 Алгоритма 2). В данном случае происходит вычисление частичных сумм

$$S_k = \sum_{i=1}^{k} w(s_i^{(z)})$$

и выполняется поиск значения параметра k, для которого выполняется условие

$$S_{k-1} \leqslant l < S_k. \tag{3.1}$$

Если в структуре дерева И/ИЛИ присутствует ИЛИ-узел с количеством сыновей, которое линейно зависит от некоторого параметра n исследуемого комбинаторного множества, то для худшего случая выполнения алгоритма генерации по рангу его вычислительная сложность увеличивается в n раз. То есть только для поиска значения параметра k (без учета всех остальных действий) оценка вычислительной сложности алгоритма уже примет линейный вид $O(n)$. Пусть в алгоритме генерации по рангу оценка вычислительной сложности для расчета значения $w(s_i^{(z)})$ равна $O(W)$ и выполняется R рекурсивных вызовов, тогда итоговая оценка вычислительной сложности данного алгоритма без учета обработки И-узлов примет вид $O(nWR)$.

Для уменьшения вычислительной сложности алгоритма генерации по рангу с точки зрения поиска значения параметра k предлагаются следующие два варианта:

1. Поиск значения параметра k на основе методов приближенных вычислений. В данном случае на основе формул вычисления значения $w(s_i^{(z)})$ или частичных сумм S_k находится значение k^*, которое является приближенным решением неравенства (3.1):

$$k^* = \texttt{SolveApprox}(S_{k-1} \leqslant l < S_k).$$

Далее исследуется точность полученного приближенного решения k^* и производится поиск точного решения k в пределах заданной окрестности δ. Тогда оценка вычислительной сложности для поиска значения параметра k изменится с $O(n)$ на $O(n^* + \delta)$, где $O(n^*)$ — оценка вычислительной сложности функции получения приближенного решения `SolveApprox`. Если $n^* + \delta \ll n$, то получаем значительное уменьшение количества выполняемых операций;

2. Поиск значения параметра k на основе метода двоичного поиска. В данном случае предполагается наличие явной формулы для вычисления частичных сумм S_k. Запишем следующее неравенство для частичных сумм:

$$0 < S_1 < S_2 < \ldots < S_{n-1} < S_n.$$

Далее воспользуемся методом двоичного поиска для поиска точного решения неравенства (3.1):

$$k = \texttt{BinarySearch}(S_{k-1} \leqslant l < S_k).$$

Тогда оценка вычислительной сложности для поиска значения параметра k изменится с $O(n)$ на $O(\log_2 n)$. Если вычислительная сложность для расчета частичной суммы S_k не превышает вычислительную сложность для расчета значения $w(s_i^{(z)})$, то получаем значительное уменьшение количества выполняемых операций.

Обобщая все предлагаемые дополнения к оригинальному алгоритму, получаем модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, который можно представить в виде последовательности следующих шагов:

1. Если известно выражение функции мощности $f(n,m,\ldots,l) = |A_{n,m,\ldots,l}|$ комбинаторного множества $A_{n,m,\ldots,l}$, принадлежащее алгебре $\{\mathbb{N}, +, \times, R\}$, то переход на шаг 4;

2. Если известно выражение производящей функции многих переменных $F(x,y,\ldots,z) = \sum\limits_{n\geqslant 0}\sum\limits_{m\geqslant 0}\ldots\sum\limits_{l\geqslant 0} f(n,m,\ldots,l)x^n y^m \cdots z^l$ для последовательности значений функции мощности $f(n,m,\ldots,l)$ комбинаторного множества $A_{n,m,\ldots,l}$, то применить комплексный метод получения явных выражений коэффициентов производящих функций многих переменных. Иначе дальнейшее применение метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ невозможно;

3. Если получено выражение функции мощности $f(n,m,\ldots,l)$ комбинаторного множества $A_{n,m,\ldots,l}$, принадлежащее алгебре $\{\mathbb{N}, +, \times, R\}$, то переход на шаг 4. Иначе дальнейшее применение метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ невозможно;

4. На основе выражения функции мощности $f(n,m,\ldots,l)$ комбинаторного множества $A_{n,m,\ldots,l}$ построить структуру дерева И/ИЛИ D;

5. Определить биекцию $A_{n,m,\ldots,l} \leftrightarrow W(D)$ между элементами комбинаторного множества $A_{n,m,\ldots,l}$ и множества всех вариантов v дерева И/ИЛИ D в виде алгоритмов $\texttt{ObjectToVariant}(a, D) : A_{n,m,\ldots,l} \to W(D)$, где $a \in A_{n,m,\ldots,l}$, и $\texttt{VariantToObject}(v, D) : W(D) \to A_{n,m,\ldots,l}$, где $v \in W(D)$;

6. Определить биекцию $W(D) \leftrightarrow \mathbb{N}_{|W(D)|}$ между элементами множества всех вариантов v дерева И/ИЛИ D и конечного множества натуральных чисел $\mathbb{N}_{|W(D)|} = \{0, 1, \ldots, |W(D)| - 1\}$ с помощью алгоритмов $\texttt{RankVariant}(v, D) : W(D) \to \mathbb{N}_{|W(D)|}$, где $v \in W(D)$, и $\texttt{UnrankVariant}(r, D) : \mathbb{N}_{|W(D)|} \to W(D)$, где $r \in \mathbb{N}_{|W(D)|}$;

7. Если в структуре дерева И/ИЛИ D присутствует ИЛИ-узел с количеством сыновей, которое зависит от параметров комбинаторного множества $A_{n,m,\ldots,l}$, то попробовать уменьшить вычислительную сложность алгоритма $\texttt{UnrankVariant}(r, D)$ за счет применения методов приближенных вычислений или метода двоичного поиска для поиска выбранного сына ИЛИ-узла.

Совокупность алгоритмов $\texttt{ObjectToVariant}(a, D) : A_{n,m,\ldots,l} \to W(D)$ и $\texttt{RankVariant}(v, D) : W(D) \to \mathbb{N}_{|W(D)|}$ представляют собой алгоритм ранжирования комбинаторного объекта $\texttt{Rank}(a) : A_{n,m,\ldots,l} \to \mathbb{N}$, который позволяет кодировать комбинаторный объект $a \in A_{n,m,\ldots,l}$ уникальным рангом $r \in \mathbb{N}_{|W(D)|}$.

Совокупность алгоритмов $\texttt{UnrankVariant}(r, D) : \mathbb{N}_{|W(D)|} \to W(D)$ и $\texttt{VariantToObject}(v, D) : W(D) \to A_{n,m,\ldots,l}$ представляют собой алгоритм генерации по рангу комбинаторного объекта $\texttt{Unrank}(r) : \mathbb{N} \to A_{n,m,\ldots,l}$, который позволяет восстановить комбинаторный объект $a \in A_{n,m,\ldots,l}$ из ранга $r \in \mathbb{N}_{|W(D)|}$.

Предложенный модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ отличается от оригинального метода и его модификаций применением разработанного комплексного метода получения явных выражений коэффициентов производящих функций многих переменных для нахождения выражения функции мощности комбинаторного множества, в том числе определяемого несколькими параметрами (отражено в действиях шага 2).

Также предложенный модифицированный метод отличается применением методов приближенных вычислений и метода двоичного поиска для поиска выбранного сына ИЛИ-узла, что позволяет снижать вычислительную сложность алгоритмов генерации по рангу (отражено в действиях шага 7).

3.3 Апробация модифицированного метода построения алгоритмов комбинаторной генерации

Рассмотрим процесс разработки новых алгоритмов ранжирования и генерации по рангу на основе предложенного модифицированного метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ. Для каждого исследуемого комбинаторного множества будут представлены: краткое описание комбинаторного множества, информация о функции мощности комбинаторного множества, структура дерева И/ИЛИ, биекция между комбинаторным множеством и множеством вариантов дерева И/ИЛИ, алгоритмы комбинаторной генерации.

3.3.1 Множество сочетаний элементов множества

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для классического комбинаторного множества сочетаний из n по m ($0 \leqslant m \leqslant n$). Сочетание из n по m — это подмножество m различных элементов выбранных из множества n элементов. Общее количество всех сочетаний из n по m может быть вычислено с помощью биномиального коэффициента $\binom{n}{m}$. Существует два основных способа представления таких сочетаний. Один возможный вариант представления сочетания из n по m — возрастающая последовательность выбранных элементов множества в данном сочетании $c = (c_1, c_2, \ldots, c_m)$, где $1 \leqslant c_1 < \cdots < c_m \leqslant n$. Другой вариант представления сочетания из n по m — двоичная последовательность $b = (b_1, b_2, \ldots, b_n)$, где $b_1 + b_2 + \ldots + b_n = m$, $b_i = 1$ кодирует выбор i-го элемента и $b_i = 0$ кодирует отсутствие выбора i-го элемента в данном сочетании.

Существует множество различных алгоритмов для генерации всех сочетаний с разным упорядочиванием. С. Акл [200] провел вычислительный эксперимент, чтобы сравнить скорость таких алгоритмов, которые были разработаны до 1980 года. В ходе эксперимента на языке программирования Fortran были реализованы девять алгоритмов генерации всех сочетаний: алгоритмы Битнера, Эрлиха и Рейнгольда; Чейза; Эрлиха; Курцберга; Лемера; Лю и Танга; Мифсуда; Нейенхейс и Уилфа; Пейна и Айвза. Результаты эксперимента показали, что алгоритм Мифсуда [201] оказался самым быстрым.

Следующий этап в получении новых алгоритмов быстрой генерации сочетаний проявляется в активном использовании параллельных вычислений (см. [202–211]). Такие параллельные алгоритмы используют m процессоров и позволяют генерировать сочетания за постоянное время на каждый объект. Помимо алгоритмов, обеспечивающих быструю генерацию сочетаний, существуют исследования, направленные на уменьшение необходимого объема памяти для вычислений. Например, А. Итай [212] представил алгоритм для генерации сочетаний в лексикографическом порядке, требующий всего $O(1)$ дополнительной памяти, но непостоянного времени.

Также имеются исследования по разработке алгоритмов генерации всех сочетаний, обеспечивающих некоторые полезные свойства за счет применения специального упорядочения генерируемых объектов. Например, существуют алгоритмы с особым порядком сгенерированных сочетаний, в которых выполняется сильное свойство минимального изменения (см. алгоритмы Идеса и Маккея [213], Сяна [214], Торрес-Хименеса и Искьердо-Маркеса [215]). Основная цель этих алгоритмов — уменьшить количество изменений при формировании сочетаний, представленных в виде возрастающих последовательностей выбранных элементов в сочетании, и, как следствие, генерировать такие сочетания за постоянное время на каждый объект или в постоянном пространстве. Еще один интересный способ упорядочивания сгенерированных сочетаний предложили Ф. Раски и А. Уильямс [216]. Этот порядок, называемый cool-lex порядком, представляет собой порядок кода Грея, основанный на использовании операций сдвига префикса.

Однако бывают ситуации, в которых необходимо снова сгенерировать определенный объект, не генерируя все предшествующие ему объекты. Для этого существуют специальные алгоритмы комбинаторной генерации для ранжирования и генерации по рангу комбинаторных объектов. Такие алгоритмы ранжирования и генерации по рангу для сочетаний из n по m в лексикографическом порядке были впервые представлены Г.Д. Ноттом [217]. Затем М.К. Эр [218] предложил новые алгоритмы ранжирования и генерации по рангу с временной сложностью $O(n-m)$ и $O(n)$ (без учета времени вычисления биномиальных коэффициентов), соответственно. Эти алгоритмы короче и проще соответствующих алгоритмов Нотта из-за представления сочетаний в виде двоичных последовательностей. Следующие алгоритмы генерации по рангу для сочетаний из n по m были разработаны З. Кокосински [219]:

– UNRANKCOMB-A: убывающий лексикографический порядок, временная сложность $O(n)$ и пространственная сложность $O(nm)$;

– UNRANKCOMB-B: возрастающий лексикографический порядок, временная сложность $O(n)$ и пространственная сложность $O(nm)$;

– UNRANKCOMB-C: убывающий лексикографический порядок, временная сложность $O(m \log n)$ и пространственная сложность $O(nm)$;

– UNRANKCOMB-D: убывающий лексикографический порядок, временная сложность $O(n)$ и пространственная сложность $O(m)$.

З. Кокосински [220] также представил первый параллельный алгоритм для генерации по рангу для сочетаний. Данный параллельный алгоритм под названием UNRANKCOMB-E требует n процессоров и имеет временную сложность $O(m)$. Кроме того, этот параллельный алгоритм имеет шаг предварительной обработки для вычисления и хранения биномиальных коэффициентов с временной сложностью $O(n)$ и пространственной сложностью $O(nm)$.

Существуют также исследования по разработке алгоритмов ранжирования и генерации по рангу для сочетаний с применением некоторого общего подхода. Например, Б.Я. Рябко [191] описал методы кодирования и декодирования комбинаторных объектов, представленных в виде последовательностей, путем вычисления количества всех префиксов таких последовательностей. Применяя данный подход, были получены новые алгоритмы ранжирования и генерации по рангу для сочетаний из n по m в лексикографическом порядке. Эти алгоритмы используют двоичные последовательности для представления сочетаний, имеют временную сложность $O(n \log^3 n \log \log n)$ и пространственную сложность $O(n \log^2 n)$. Применяемый в рамках данного диссертационного исследования подход к построению алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ также был применен для разработки алгоритмов ранжирования и генерации по рангу для сочетаний [221]. Полученные комбинаторные алгоритмы используют двоичные последовательности для представления сочетаний из n по m и имеют временную сложность $O(n + m^2)$ и $O(nm)$, соответственно.

В [222] предложен способ улучшить алгоритмы ранжирования и генерации по рангу, основанные на вычислении биномиальных коэффициентов. Основная идея состоит в том, чтобы вычислить новые биномиальные коэффициенты на основе биномиальных коэффициентов, полученных на предыдущих шагах. Эффективность этого подхода была проверена на нескольких известных алгоритмах генерации по

рангу для сочетаний и на новом алгоритме генерации по рангу для сочетаний в лексикографическом порядке, основанном на факторадической системе счисления.

Кроме того, существуют исследования, посвященные частному случаю разработки новых эффективных алгоритмов ранжирования и генерации по рангу для сочетаний из n по m, когда полученные алгоритмы не зависят от n. Следовательно, такие алгоритмы эффективны, когда n велико, а m мало. Например, в [223] представлены алгоритмы ранжирования и генерации по рангу с временной сложностью $O(m^{3m+3})$. Эти алгоритмы используют несколько биекций из множества сочетаний в другие комбинаторные множества. Более того, порядок генерируемых сочетаний не является одним из известных вариантов упорядочения элементов комбинаторных множеств. В [224; 225] представлен еще один подход к генерации по рангу малых сочетаний элементов больших множеств с использованием градиентного метода оптимизации. Порядок сгенерированных сочетаний из n по m соответствует порядку вращающейся двери, который начинается с сочетания $(1, 2, \ldots, m-1, m)$ и заканчивается сочетанием $(1, 2, \ldots, n-1, n)$. Для поиска каждого элемента c_i сочетания $c = (c_1, c_2, \ldots, c_m)$ необходимо решить задачу минимизации с ограничениями с помощью градиентного метода оптимизации. Данный алгоритм генерации по рангу имеет временную сложность $O(m^2)$ при использовании одного процессора и $O(m \log m)$ при использовании не более чем $O(m/\log m)$ процессоров. Однако нет информации о каком-либо алгоритме ранжирования сочетаний в этом порядке.

Рассмотрим процесс построения алгоритма генерации по рангу для малых сочетаний элементов больших множеств с помощью разработанного модифицированного метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ.

Алгоритмы ранжирования и генерации по рангу для сочетаний в колексикографическом порядке

Существует несколько вариантов упорядочения любого списка строк или последовательностей, например, широко известные лексикографический и колексикографический порядки [171].

Лексикографический (lex) порядок — это такой порядок, в котором $a_1, a_2, \ldots, a_n < b_1, b_2, \ldots, b_m$, если $n = m$ и имеется $k \in \{1, \ldots, n\}$ удовлетворяющее $a_k < b_k$ и $a_i = b_i$ для всех $i < k$ либо если $n < m$ и $a_i = b_i$ для $i \leqslant n$.

Колексикографический (colex) порядок — это такой порядок, в котором $a_1, a_2, \ldots, a_n < b_1, b_2, \ldots, b_m$, если $a_n, \ldots, a_2, a_1 < b_m, \ldots, b_2, b_1$ в лексикографическом порядке.

Лексикографический порядок является естественным и понятным способом упорядочения. В то же время колексикографический порядок может сделать комбинаторные алгоритмы короче, быстрее, проще и естественнее [172]. Для ранжирования сочетаний $c = (c_1, c_2, \ldots, c_m)$ в колексикографическом порядке можно использовать следующую формулу [172, Формула (4.10)]:

$$\texttt{RankColex}(c) = \sum_{i=1}^{m} \binom{c_i - 1}{i}. \tag{3.2}$$

Применяя (3.2), можно быстро вычислить ранг данного сочетания из n по m при больших n, поскольку он не зависит от параметра n. Рассмотрим следующий алгоритм генерации по рангу для сочетаний в колексикографическом порядке, описанный в [172, Алгоритм 4.10]:

Алгоритм 3: Алгоритм генерации по рангу для сочетаний в колексикографическом порядке

```
1  UnrankColex(r,m)
2  begin
3      for i := m to 1 do
4          k := i
5          while r ⩾ (k choose i) do k := k + 1
6          c_i := k
7          r := r − (k−1 choose i)
8      end
9      return c
10 end
```

В качестве входных данных Алгоритм 3 для заданного сочетания из n по m принимает количество m выбранных элементов и ранг $r < \binom{n}{m}$ сочетания в колексикографическом порядке. В результате значения выбранных элементов c_i в сочетании будут получены в виде последовательности $(c_1, c_2, \ldots, c_m)$.

Основная идея этого алгоритма генерации по рангу для сочетаний в колексикографическом порядке заключается в следующем:

Сначала для случая $k = m$ ищется интервал, содержащий ранг r, среди следующих биномиальных коэффициентов:

$$\binom{m}{m} + \binom{m+1}{m} + \binom{m+2}{m} + \ldots + \binom{n}{m} = \binom{n+1}{m+1}. \tag{3.3}$$

Легко увидеть, что количество членов слева в (3.3) равно $n - m + 1$. Если известен ранг r сочетания из n по m и решение для неравенства

$$\binom{k-1}{m} \leqslant r < \binom{k}{m}, \tag{3.4}$$

тогда m-й выбранный элемент в сочетании равен k.

Далее нужно получить $(m-1)$-й выбранный элемент в сочетании путем поиска интервала, содержащего ранг

$$r := r - \binom{k-1}{m}. \tag{3.5}$$

Таким образом, решая неравенство (3.4) и применяя (3.5), можно последовательно найти значения выбранных элементов в сочетании с рангом r. При $m < n - m$ вычислительная сложность Алгоритма 3 равна $O\left(m^2 \cdot (n-m)\right)$, что было получено на основе следующей информации:

– количество итераций в цикле for (строка 2 в Алгоритме 3) равно m;

– минимальное количество сравнений в цикле while (строка 4 в Алгоритме 3) равно $n - m + 1$ (для $i = m$) и максимальное количество сравнений равно n (для $i = 1$);

– вычислительная сложность для расчета значения биномиального коэффициента $\binom{n}{m}$ равна $O(m)$ при $m < n - m$ и $O(n - m)$ при $m > n - m$.

При больших n и малых m количество интервалов вида (3.4), которые могут содержать требуемый ранг r, велико из-за зависимости от n. Цель предлагаемых улучшений Алгоритма 3 — уменьшить количество сравнений в цикле while (строка 4 в Алгоритме 3). Для этого необходимо быстро находить решение неравенства (3.4).

Аналогичный результат с точки зрения построения алгоритма ранжирования генерации по рангу для сочетаний может быть получен на основе метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ.

Для этого необходимо воспользоваться следующим выражением функции мощности комбинаторного множества:

$$\binom{n}{m} = \binom{m-1}{m-1} + \binom{m}{m-1} + \binom{m+1}{m-1} + \ldots + \binom{n-2}{m-1} + \binom{n-1}{m-1}. \quad (3.6)$$

Так как данная функция мощности принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.2).

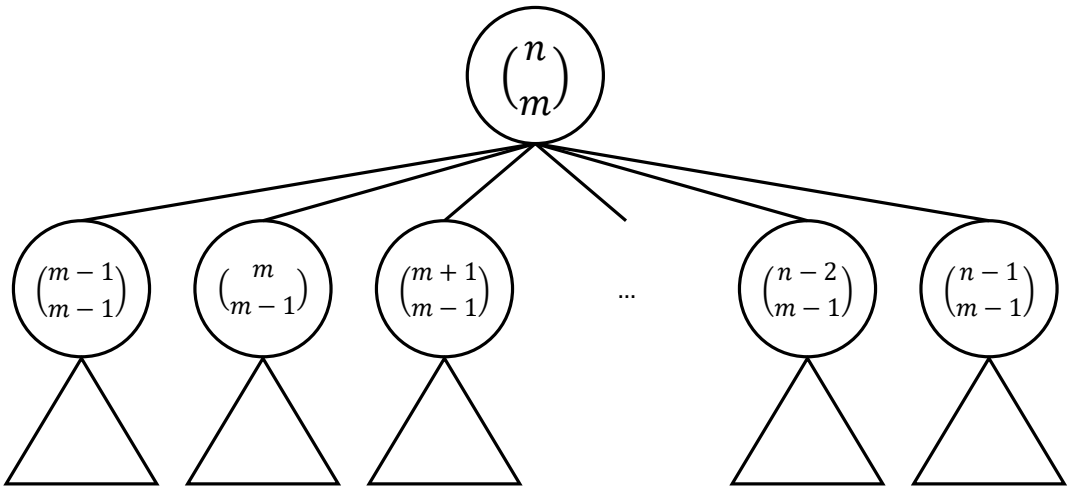


Рисунок 3.2 — Структура дерева И/ИЛИ для функции мощности (3.6)

В полученной структуре дерева И/ИЛИ присутствует ИЛИ-узел с количеством сыновей, которое зависит от параметров исследуемого комбинаторного множества сочетаний (имеется $n - m + 1$ сыновей ИЛИ-узла, помеченного $\binom{n}{m}$). Согласно шагу 7 предложенного в данном диссертационном исследовании модифицированного метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, чтобы уменьшить вычислительную сложность алгоритма генерации по рангу, применим методы приближенных вычислений для поиска выбранного сына ИЛИ-узла. Для этого воспользуемся следующей формулой для вычисления частичных сумм:

$$S_k = \sum_{i=m-1}^{k} \binom{i}{m-1} = \binom{k+1}{m}. \quad (3.7)$$

Таким образом, на основе данной формулы вычисления частичных сумм S_k можно найти значение k^*, которое является приближенным решением неравенства (3.1). Также данный подход может быть применен для решения неравенства (3.4), в таком случае получим следующий вид связи с частичными суммами из (3.7):

$$S_{k-1} \leqslant r < S_k.$$

В соответствии с предложенными дополнениями, изменим Алгоритм 3, добавив в него функцию `FindK`(r,m) для предварительного поиска значения k. Эта функция должна вычислять приближенное значение k для решения неравенства (3.4). При этом полученное значение k не должно быть больше точного значения решения данного неравенства. Таким образом, получаем новый алгоритм генерации по рангу для сочетаний в колексикографическом порядке:

Алгоритм 4: Модификация алгоритма генерации по рангу для сочетаний в колексикографическом порядке

```
1  UnrankColexNew(r,m)
2  begin
3      for i := m to 1 do
4          k := FindK(r,i)
5          while r ⩾ C(k,i) do k := k + 1
6          c_i := k
7          r := r − C(k−1,i)
8      end
9      return c
10 end
```

Далее подробно рассмотрим разработку внутренностей функции `FindK`(r,m) для Алгоритма 4.

Биномиальный коэффициент может быть представлен как

$$\binom{n}{m} = \frac{n(n-1)(n-2)\cdots(n-m+1)}{m!}. \tag{3.8}$$

Используя (3.4) и (3.8), получаем

$$r < \frac{k(k-1)(k-2)\cdots(k-m+1)}{m!}.$$

После некоторых преобразований получаем следующее неравенство:

$$\sqrt[m]{k(k-1)(k-2)\cdots(k-m+1)} > \sqrt[m]{r\ m!}. \tag{3.9}$$

Если рассмотреть неравенство о среднем арифметическом и геометрическом [226, pp. 190]

$$\frac{1+2+3+\ldots+n}{n} \geqslant \sqrt[n]{1\cdot 2\cdot 3\cdots n}$$

для чисел $k, (k-1), (k-2), \ldots, (k-m+1)$, тогда получим

$$\frac{k+(k-1)+(k-2)+\ldots+(k-m+1)}{m} \geqslant \sqrt[m]{k(k-1)(k-2)\cdots(k-m+1)}.$$

После упрощений слева получаем

$$k-\frac{m-1}{2} \geqslant \sqrt[m]{k(k-1)(k-2)\cdots(k-m+1)}. \tag{3.10}$$

Объединяя (3.9) и (3.10), находим

$$k-\frac{m-1}{2} > \sqrt[m]{r\ m!}$$

или

$$k > \sqrt[m]{r\ m!}+\frac{m-1}{2}.$$

Следовательно,

$$k \approx \left\lceil \sqrt[m]{r\ m!}+\frac{m-1}{2} \right\rceil. \tag{3.11}$$

Тогда, применяя (3.11), можем найти приближенное значение k, при котором выполняется следующее неравенство:

$$r < \binom{k}{m}. \tag{3.12}$$

Также воспользуемся следующим приближением Стирлинга [227, p. 518] для вычисления факториала в (3.11):

$$n! \approx \sqrt{2\pi n}\ \left(\frac{n}{e}\right)^n\ e^{\frac{1}{12n}-\frac{1}{360n^3}}. \tag{3.13}$$

Используя приближенную формулу (3.13) в (3.11) и сделав некоторые преобразования, получаем следующую приближенную формулу для k:

$$k \approx \left\lceil m\ e^{\frac{\log r}{m}+\frac{\log 2\pi m}{2m}+\frac{1}{12m^2}-\frac{1}{360m^4}-1}+\frac{m-1}{2} \right\rceil. \tag{3.14}$$

Эту приближенную формулу можно использовать для предварительного поиска значения k. То есть формула (3.14) может быть положена в основу функции `FindK`(r,m) (строка 3 в Алгоритме 4).

Вычислительные эксперименты

С целью проверки полученной приближенной формулы (3.14) проведем вычислительные эксперименты. Для этого установим количество выбранных элементов m в диапазоне от 10 до 200 и ранг r в диапазоне от 10^{100} до 10^{200}. При этих значениях параметров m и r имеем распределение значений параметра n в диапазоне от 332 (при $m = 170$ и $r = 10^{100}$) до $4{,}5 \cdot 10^{20}$ (когда $m = 10$ и $r = 10^{200}$). Затем вычислим разницу между значением k, полученным с помощью приближенной формулы (3.14), и точным значением k, полученным с помощью цикла while (строка 4 в Алгоритме 3). Полученные результаты проведенного вычислительного эксперимента представлены в таблице 3.1.

Таблица 3.1 — Ошибки определения значения k в зависимости от значений параметров m и r

m	r										
	10^{100}	10^{110}	10^{120}	10^{130}	10^{140}	10^{150}	10^{160}	10^{170}	10^{180}	10^{190}	10^{200}
10	0	0	0	0	0	0	0	0	0	1	1
20	0	0	0	0	0	0	0	0	0	1	1
30	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0
60	1	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0
80	1	0	0	0	0	1	0	0	0	0	0
90	1	1	0	1	0	0	0	1	0	0	0
100	1	1	1	1	1	1	0	0	0	0	0
110	2	1	1	0	1	1	1	0	0	0	0
120	2	1	1	1	1	1	0	0	1	0	1
130	3	2	2	1	1	1	1	0	0	1	0
140	3	3	2	1	1	1	1	1	1	1	0
150	4	3	3	3	2	1	2	2	1	1	1
160	4	4	3	2	2	2	2	1	1	1	1
170	5	4	4	4	3	2	3	2	2	1	2
180	5	5	5	4	3	3	2	2	2	2	2
190	7	6	5	5	3	3	3	3	2	2	2
200	7	6	6	5	5	4	3	3	3	2	2

Из таблицы 3.1 видно, что ошибки определения значения k в зависимости от значений параметров m и r являются небольшими по сравнению с значениями самих параметров. При этом обратим внимание, что увеличение ранга r уменьшает ошибку, а увеличение параметра m увеличивает ошибку. Это объясняется использованием для расчетов приближенных формул (3.11) и (3.13). Также приближенная формула (3.14) не может быть применена для сильно малых m, так как применение (3.13) к таким значениям m приводит к увеличению ошибки вычисления значения факториала. Кроме того, для таких значений m можно найти значение k, решив уравнение, полученное из (3.12):

Если $m = 1$, тогда

$$r < \binom{k}{1}.$$

В данном случае можем использовать следующее начальное условие для значения k:

$$k = r.$$

Если $m = 2$, тогда

$$r < \binom{k}{2}$$

или

$$k^2 - k - 2r > 0.$$

В данном случае можем использовать следующее начальное условие для значения k:

$$k = \left\lceil \frac{1 + \sqrt{1 + 8r}}{2} \right\rceil.$$

Для других сильно малых значений m можно использовать другие приближенные формулы для решения (3.12), например, это может быть приближенная формула (3.11). Таким образом, получаем алгоритм предварительного поиска значения k (Алгоритм 5). В этом алгоритме параметр s показывает граничное значение параметра m для применения приближенной формулы (3.14).

Алгоритм 4 с использованием Алгоритма 5 имеет временную сложность $O(m \cdot (m + \varepsilon m)) \approx O(m^2)$ (предполагая выполнение алгебраических операций с числами с временной сложностью $O(1)$). Это определяется количеством m итераций в цикле for (строка 2 в Алгоритме 4), где каждый цикл требует:

Алгоритм 5: Предварительный поиск значения k для Алгоритма 4

```
1 FindK(r,m)
2 begin
3     if r = 0 then k := m
4     else if m = 1 then  k := r
5     else if m = 2 then  k := ⌈(1+√(1+8r))/2⌉
6     else if m < s then  k := k ≈ ⌈ ∛(r m!) + (m−1)/2 ⌉
7     else k := ⌈ m e^{(log r)/m + (log 2πm)/(2m) + 1/(12m^2) − 1/(360m^4) − 1} + (m−1)/2 ⌉
8     return k
9 end
```

– предварительный поиск значения k (строка 3 в Алгоритме 4) с использованием Алгоритма 5, имеющий временную сложность $O(m)$;

– поиск точного значения k путем выполнения ε итераций в цикле while (строка 4 в Алгоритме 4);

– вычисление значения биномиального коэффициента $\binom{n}{m}$, имеющее временную сложность $O(m)$ при $m < n - m$.

Параметр ε показывает ошибку определения значения k и является существенно меньше, чем значение параметра n (см. таблицу 3.1). Следовательно, модифицированный Алгоритм 4 имеет полиномиальную сложность, зависящую только от параметра m. В случае, когда n велико, а m мало, этот алгоритм будет лучше исходного Алгоритма 3 из-за независимости от параметра n в его оценке вычислительной сложности.

Дополнительно проведем вычислительный эксперимент, направленный на сравнение полученного алгоритма генерации по рангу малых сочетаний элементов больших множеств в колексикографическом порядке с алгоритмом Парка и Мияшиты [224; 225]. Для этого устанавливаем количество выбранных элементов m в диапазоне от 10 до 100 и генерируем 20 различных сочетаний из n по m с равномерно распределенными рангами r в диапазоне 0 до $\binom{n}{m} - 1$. Алгоритмы были реализованы в системе компьютерной алгебре «Maxima» (ноутбук Intel i7-9750H, 2,6 ГГц, Windows 10, 64-разрядная версия). На рисунках 3.3–3.4 показано среднее время генерации каждого сочетания для $n = 500$ и $n = 1000$. Заметим, что оба алгоритма показывают схожие результаты, однако есть разница в способе упорядочения сочетаний и в наличии алгоритма ранжирования.

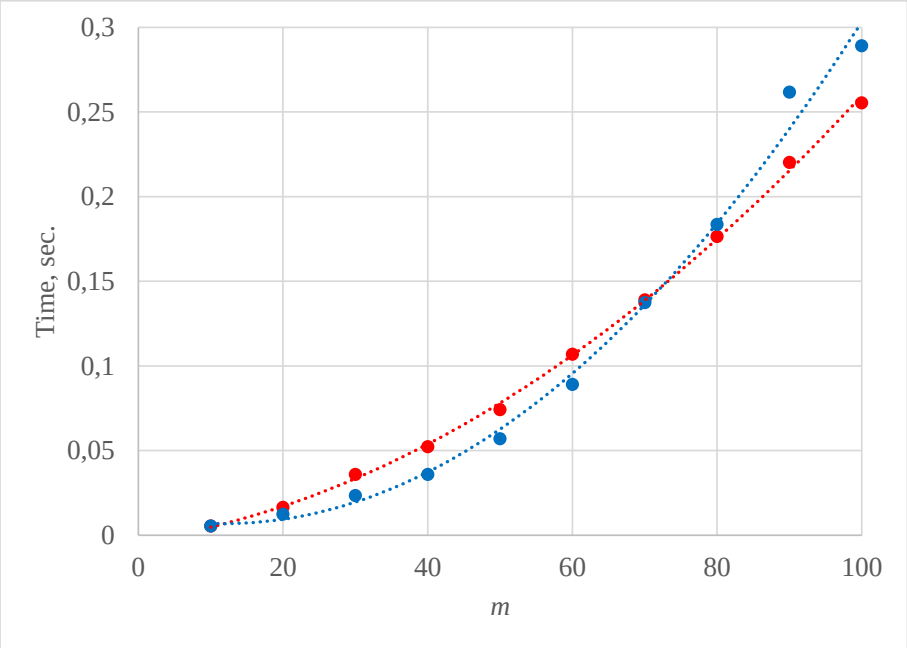


Рисунок 3.3 — Среднее время генерации сочетаний для $n = 500$:
(красная линия) Алгоритм Парка и Мияшиты
(синяя линяя) Разработанный алгоритм

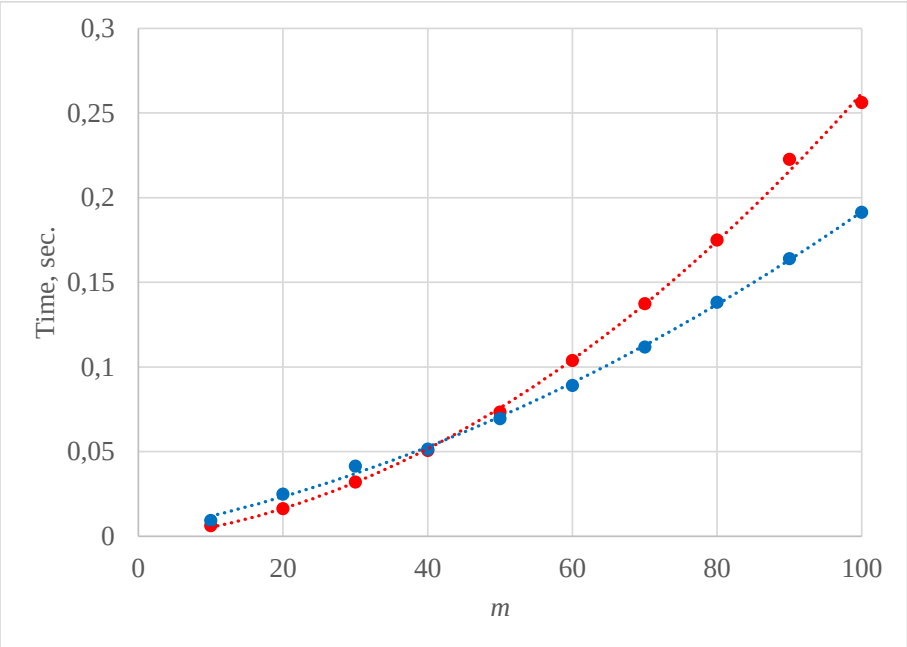


Рисунок 3.4 — Среднее время генерации сочетаний для $n = 1000$:
(красная линия) Алгоритм Парка и Мияшиты
(синяя линяя) Разработанный алгоритм

3.3.2 Множество решеточных путей на плоскости

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: самонепересекающийся решеточный путь на плоскости из точки (0,0) в точку (n,n) с шагами вида (0,1) (шаг вверх), (1,0) (шаг вправо) и $(-1,0)$ (шаг влево) [228]. При этом остановимся на частном случае такого решеточного пути, содержащего всего $2(n+2)$ шагов, из которых n шагов вверх, $n+2$ шага вправо и 2 шага влево.

Данный комбинаторный объект предлагается кодировать в виде последовательности $a = (a_1, \ldots, a_{2n+4})$, где каждое a_i представляет собой очередной выполненный шаг решеточного пути, при этом $a_i =$ "N" обозначает шаг вверх, $a_i =$ "E" — шаг вправо, $a_i =$ "W" — шаг влево.

Значения функции мощности данного комбинаторного множества формируют следующую целочисленную последовательность (последовательность $A119578$ в OEIS [94]):

$$0, 2, 18, 120, 700, 3780, 19404, 96096, 463320, 2187900, \ldots$$

На рисунке 3.5 показан пример всех возможных вариантов рассматриваемых решеточных путей при $n = 2$.

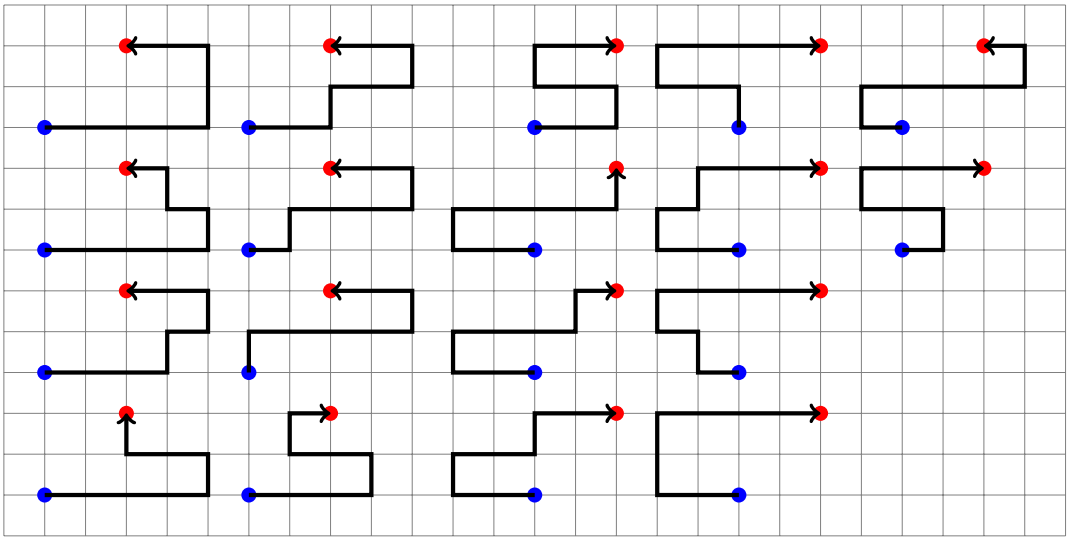


Рисунок 3.5 — Все самонепересекающиеся решеточные пути из точки (0,0) в точку (2,2) с 2 шагами вверх, 4 шагами вправо и 2 шагами влево

Далее рассмотрим обобщение в виде комбинаторного множества таких самонепересекающихся решеточных путей из точки (0,0) в точку (k,k) с двумя шагами влево для всех $k \leqslant n$.

Представление в виде структуры дерева И/ИЛИ

Известна следующая функция мощности для комбинаторного множества самонепересекающихся решеточных путей из точки (0,0) в точку (n,n) с n шагами вверх, $n+2$ шагами вправо и 2 шагами влево:

$$f_1(n) = \binom{2n}{n}\binom{n+1}{2},$$

для $n > 0$ и $f_1(0) = 0$.

Тогда функция мощности для комбинаторного множества самонепересекающихся решеточных путей из точки (0,0) в точку (k,k) с k шагами вверх, $k+2$ шагами вправо и 2 шагами влево для всех $k \leqslant n$, принимает следующий вид:

$$f_2(n) = \sum_{k=0}^{n} f_1(k) = \sum_{k=1}^{n} \binom{2k}{k}\binom{k+1}{2}. \tag{3.15}$$

Так как функция мощности (3.15) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.6).

Данная структура дерева И/ИЛИ основана на применении известной структуры дерева И/ИЛИ для комбинаторного множества сочетаний элементов. При этом поддерево узла, помеченного $\binom{2k}{k}$, показывает вариант самонепересекающегося решеточного пути из точки (0,0) в точку (k,k) с k шагами вверх и k шагами вправо, а поддерево узла, помеченного $\binom{k+1}{2}$, определяет вариант встраивания в данный решеточный путь 2 шага влево.

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде последовательности $v = (k, v_1, v_2)$, в которой:

– k определяет метку выбранного сына ИЛИ-узла, помеченного $f_2(n)$;

– v_1 соответствует варианту поддерева узла, помеченного $\binom{2k}{k}$;

– v_2 соответствует варианту поддерева узла, помеченного $\binom{k+1}{2}$.

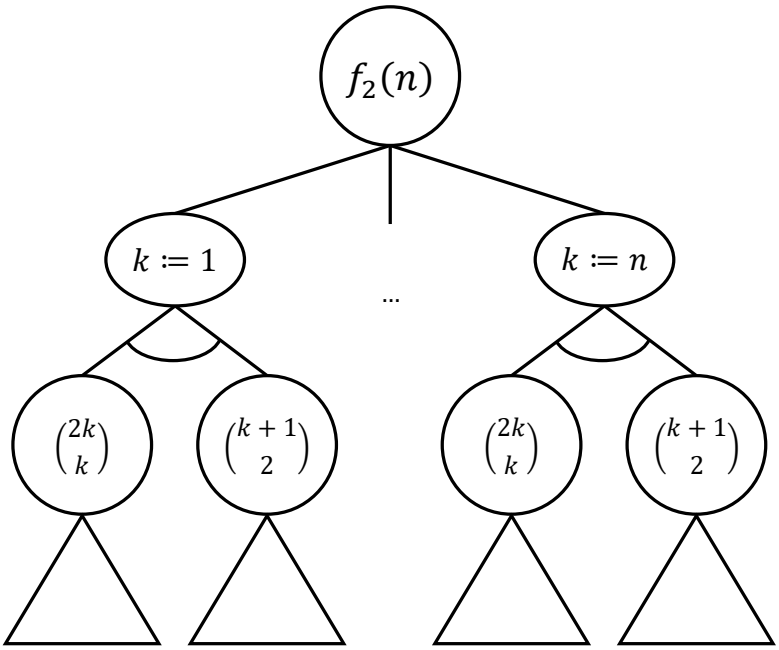


Рисунок 3.6 — Структура дерева И/ИЛИ для функции мощности (3.15)

Алгоритмы ранжирования и генерации по рангу

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества решеточных путей на плоскости (Алгоритм 6 и Алгоритм 7). В данных алгоритмах также используются разработанные ранее алгоритмы ранжирования и генерации по рангу вариантов деревьев И/ИЛИ для множества сочетаний элементов.

Алгоритм 6: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.6

```
1 RankVariant (v = (k, v_1, v_2), n)
2 begin
3 |   l_1 := RankVariant_C (v_1, 2k, k)
4 |   l_2 := RankVariant_C (v_2, k + 1, 2)
5 |   r := l_1 + l_2 * C(2k, k) + sum_{i=1}^{k-1} C(2i, i) * C(i+1, 2)
6 |   return r
7 end
```

Алгоритм 7: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.6

```
1  UnrankVariant (r, n)
2  begin
3      k := 1
4      sum := 0
5      while sum + C(2k, k) C(k+1, 2) ⩽ r do
6          sum := sum + C(2k, k) C(k+1, 2)
7          k := k + 1
8      end
9      r := r − sum
10     l1 := r  mod C(2k, k)
11     l2 := ⌊ r / C(2k, k) ⌋
12     v1 := UnrankVariant_C (l1, 2k, k)
13     v2 := UnrankVariant_C (l2, k + 1, 2)
14     v = (k, v1, v2)
15     return v
16 end
```

В полученной структуре дерева И/ИЛИ присутствует ИЛИ-узел с количеством сыновей, которое зависит от параметров исследуемого комбинаторного множества решеточных путей (имеется n сыновей ИЛИ-узла, помеченного $f_2(n)$). Следовательно, в алгоритме генерации по рангу (Алгоритм 7) выполняются вычисления частичных сумм следующего вида:

$$S_k = \sum_{i=1}^{k} \binom{2i}{i} \binom{i+1}{2}. \tag{3.16}$$

Таким образом, для худшего случая вычислительная сложность поиска выбранного сына ИЛИ-узла (определение значения параметра k в строках 3–8 Алгоритма 7) равна $O(n^2)$. Данная оценка вычислительной сложности складывается из вычислительной сложности $O(n)$ для расчета биномиального коэффициента $\binom{2n}{n}$, а также из максимального количества итераций в цикле while, которое равно n.

Согласно шагу 7 предложенного в данном диссертационном исследовании модифицированного метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, чтобы уменьшить вычислительную сложность алгоритма генерации по рангу, применим метод двоичного поиска для поиска выбранного сына ИЛИ-узла. Для этого воспользуемся следующей формулой для вычисления частичных сумм (3.16):

$$S_k = \frac{(2k+1)!}{3(k-1)!k!}.$$

Тогда получаем следующую модификацию алгоритма генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.6:

Алгоритм 8: Модификация алгоритма генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.6

```
1  UnrankVariant (r, n)
2  begin
3      k := BinarySearch(S_{k-1} ⩽ r < S_k)
4      r := r − S_{k-1}
5      l_1 := r mod C(2k, k)
6      l_2 := ⌊ r / C(2k, k) ⌋
7      v_1 := UnrankVariant_C (l_1, 2k, k)
8      v_2 := UnrankVariant_C (l_2, k + 1, 2)
9      v = (k, v_1, v_2)
10     return v
11 end
```

Применение метода двоичного поиска позволяет в среднем сократить количество требуемых операций для поиска точного решения неравенства (3.1). В результате для худшего случая вычислительная сложность поиска выбранного сына ИЛИ-узла станет равна $O(n \log_2 n)$, что является лучшим значением по сравнению с исходной версией алгоритма.

3.3.3 Множество помеченных путей Дика с подъемами на возвратных шагах

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: помеченный путь Дика длины $2n$ с m подъемами на возвратных шагах. Путь Дика длины $2n$ — это решеточный путь на плоскости, который начинается в точке (0,0), заканчивается в точке ($2n$,0) и состоит из шагов вида (1,1) (шаг по диагонали вправо вверх, называемый подъемом) и шагов вида $(1, -1)$ (шаг по диагонали вправо вниз, называемый спуском). Возвратный шаг — это спуск с уровня 1 (возвращение на уровень 0) [229]. В помеченном пути Дика длины $2n$ с m подъемами на возвратных шагах каждый спуск имеет свою метку (уникальное число от 1 до n). Если рассмотреть последовательность пометок спусков пути Дика, то в ней имеется ровно m подъемов (то есть ровно m пометок больше чем предшествующее им значение).

На рисунке 3.7 показан пример всех возможных вариантов рассматриваемых помеченных путей Дика при $n = 3$ и $m = 1$.

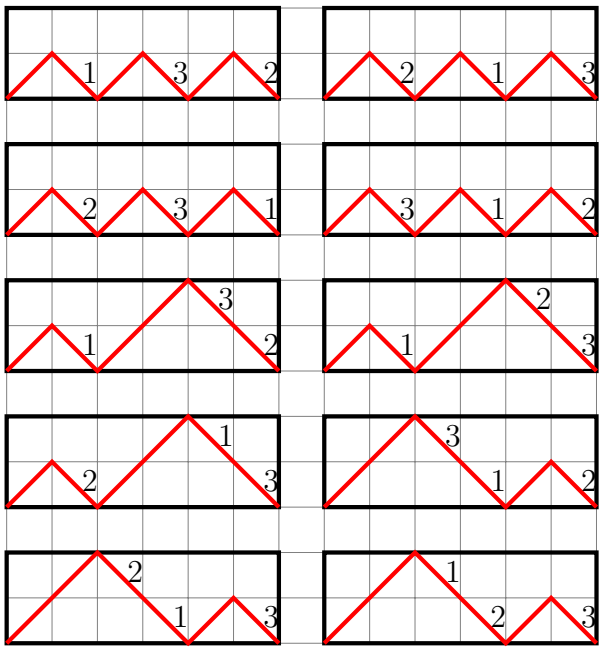


Рисунок 3.7 — Все помеченные пути Дика длины 6 с 1 подъемом на возвратных шагах

Представление в виде структуры дерева И/ИЛИ

В работе [195] представлена разработка алгоритмов комбинаторной генерации для исследуемого комбинаторного множества с помощью метода на основе деревьев И/ИЛИ. При этом функция мощности данного комбинаторного множества, принадлежащая алгебре $\{\mathbb{N}, +, \times, R\}$, была получена на основе частных выводов о виде и структуре производящей функции двух переменных

$$EC(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\frac{EC_n^m}{n!}x^n y^m = \frac{y-1}{y-e^{C(x)(y-1)}}, \tag{3.17}$$

где

$$C(x) = \frac{1-\sqrt{1-4x}}{2}$$

представляет собой производящую функцию для чисел Каталана. В результате было получено явное выражение следующего вида [196]:

$$EC_n^m = \begin{cases} 1, & \text{для } n = m = 0; \\ \sum\limits_{k=m+1}^{n} CT_n^k E_k^m P_n^{n-k}, & \text{иначе.} \end{cases}$$

Рассмотрим применение разработанного комплексного метода получения коэффициентов производящих функций многих переменных для случая производящей функции (3.17). Согласно данному методу, представим производящую функцию $EC(x,y)$ как композицию более простых и известных производящих функций следующего вида:

$$EC(x,y) = E(C(x),y),$$

где

$$E(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} e(n,m)x^n y^m = \frac{y-1}{y-e^{x(y-1)}} = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\frac{E_n^m}{n!}x^n y^m$$

представляет собой производящую функцию для чисел Эйлера первого рода [2].

Воспользуемся известным выражением для композиты производящей функции $C(x)$ [68]

$$C^{\Delta}(n,k) = \frac{k}{n}\binom{2n-k-1}{n-1}.$$

Применяя Теорему 5 для композиции $EC(C(x),y)$, получаем следующее явное выражение для ее коэффициентов:

$$EC_{n,m} = n! \sum_{k=0}^{n} e(k,m) C^{\Delta}(n,k) = \sum_{k=m+1}^{n} E_k^m \frac{(2n-k-1)!}{(k-1)!(n-k)!}.$$

Кроме того, данное выражение можно преобразовать к следующему виду:

$$EC_n^m = \sum_{k=m+1}^{n} CT_n^k C_n^k E_k^m P_{n-k}. \tag{3.18}$$

Полученное выражение (3.18) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, так как для ее составляющих известны следующие рекуррентные формулы:

– элементы транспонированного треугольника Каталана (последовательность $A033184$ в [94]) показывают количество путей Дика длины $2n$ с m возвратными шагами и могут быть вычислены на основе рекуррентного соотношения [229]

$$CT_n^m = CT_{n-1}^{m-1} + CT_n^{m+1}, \quad CT_n^0 = 0, \quad CT_n^n = 1; \tag{3.19}$$

– количество сочетаний из n по m (последовательность $A007318$ в [94]) может быть вычислено на основе рекуррентного соотношения [2]:

$$C_n^m = C_{n-1}^m + C_{n-1}^{m-1}, \quad C_n^n = C_n^0 = 1; \tag{3.20}$$

– элементы треугольника Эйлера (последовательность $A173018$ в [94]) показывают количество перестановок n элементов с m подъемами и могут быть вычислены на основе рекуррентного соотношения [230]:

$$E_n^m = (m+1)E_{n-1}^m + (n-m)E_{n-1}^{m-1}, \quad E_n^{n-1} = E_n^0 = 1; \tag{3.21}$$

– количество перестановок n элементов (последовательность $A000142$ в [94]) может быть вычислено на основе рекуррентного соотношения [2]:

$$P_n = nP_{n-1}, \quad P_0 = 1. \tag{3.22}$$

Таким образом, можно построить структуру дерева И/ИЛИ для функции мощности (3.18), которая представлена на рисунке 3.8. Данная структура дерева И/ИЛИ основана на применении известных схем рекурсивных композиций деревьев И/ИЛИ для выражений (3.19)–(3.22) [195].

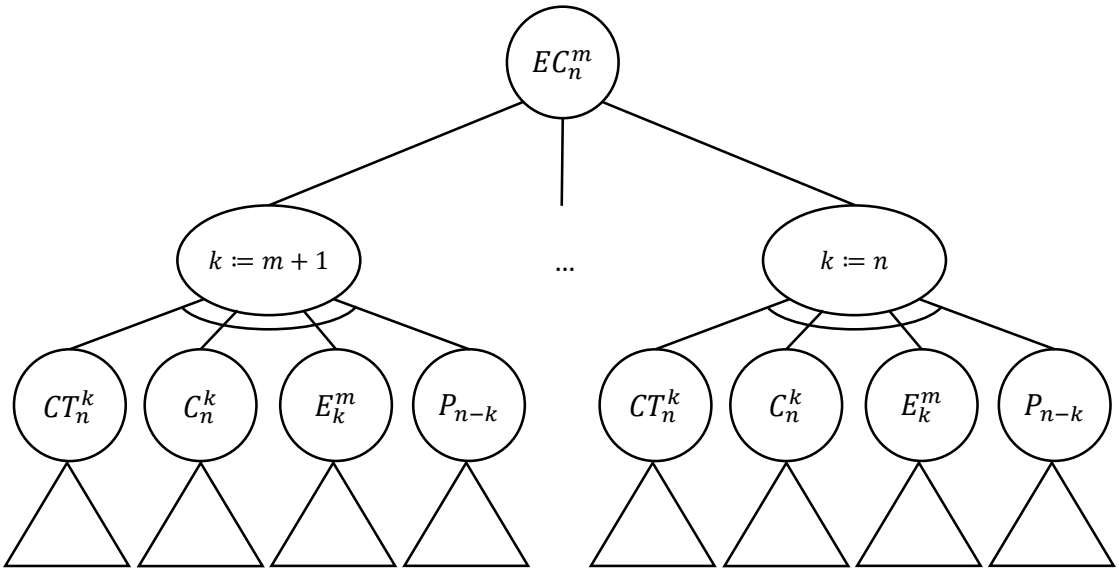


Рисунок 3.8 — Структура дерева И/ИЛИ для функции мощности (3.18)

В таком случае биекция между помеченными путями Дика длины $2n$ с m подъемами на возвратных шагах и вариантами полученного дерева И/ИЛИ определяется набором следующих правил:

– количество возвратных шагов в пути Дика определяется значением параметра k (выбранный сын ИЛИ-узла, помеченного EC_n^m);

– поддерево узла, помеченного CT_n^k, определяет вид пути Дика длины $2n$ с k возвратными шагами;

– поддерево узла, помеченного C_n^k, определяет k значений, взятых из множества n значений, которые используются в качестве пометок возвратных шагов в пути Дика, оставшиеся $n-k$ значений используются в качестве пометок оставшихся $n-k$ спусков в пути Дика;

– поддерево узла, помеченного E_k^m, определяет вид перестановки пометок k возвратных шагов в пути Дика, в которой содержится ровно m подъемов;

– поддерево узла, помеченного P_{n-k}, определяет вид перестановки пометок оставшихся $n-k$ спусков в пути Дика.

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать последовательностью $v = (k, v_1, v_2, v_3, v_4)$, в которой:

– k определяет метку выбранного сына ИЛИ-узла, помеченного EC_n^m;

– v_1 соответствует варианту поддерева узла, помеченного CT_n^k;

– v_2 соответствует варианту поддерева узла, помеченного C_n^k;

– v_3 соответствует варианту поддерева узла, помеченного E_k^m;

– v_4 соответствует варианту поддерева узла, помеченного P_{n-k}.

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества помеченных путей Дика с подъемами на возвратных шагах (Алгоритм 9 и Алгоритм 10). В данных алгоритмах также используются разработанные ранее алгоритмы ранжирования и генерации по рангу вариантов деревьев И/ИЛИ для выражений (3.19)–(3.22) [195].

В полученной структуре дерева И/ИЛИ присутствует ИЛИ-узел с количеством сыновей, которое зависит от параметров исследуемого комбинаторного множества (имеется $n - m$ сыновей ИЛИ-узла, помеченного EC_n^m). Следовательно, в алгоритме генерации по рангу (Алгоритм 10) выполняются вычисления частичных сумм следующего вида:

$$S_k = \sum_{i=m+1}^{k} CT_n^i C_n^i E_i^m P_{n-i}. \tag{3.23}$$

Также отметим, что в алгоритме генерации по рангу вычисление параметра k выполняется всего один раз, так как отсутствуют рекурсивные вызовы основной функции `UnrankVariant_EC`.

Алгоритм 9: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.8

```
1 RankVariant_EC (v = (k, v1, v2, v3, v4), n, m)
2 begin
3     l1 := RankVariant_CT (v1, n, k)
4     l2 := RankVariant_C (v2, n, k)
5     l3 := RankVariant_E (v3, k, m)
6     l4 := RankVariant_P (v4, n − k)
7     r := l1 + CT_n^k (l2 + C_n^k (l3 + E_k^m l4)) + Σ_{i=m+1}^{k−1} CT_n^i C_n^i E_i^m P_{n−i}
8     return r
9 end
```

Алгоритм 10: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.8

```
1  UnrankVariant_EC (r, n, m)
2  begin
3      k := m + 1
4      sum := 0
5      while sum + CT_n^k C_n^k E_k^m P_{n-k} ⩽ r do
6          sum := sum + CT_n^k C_n^k E_k^m P_{n-k}
7          k := k + 1
8      end
9      r := r − sum
10     l_1 := r  mod CT_n^k
11     r := ⌊ r / CT_n^k ⌋
12     l_2 := r  mod C_n^k
13     r := ⌊ r / C_n^k ⌋
14     l_3 := r  mod E_k^m
15     l_4 := ⌊ r / E_k^m ⌋
16     v_1 := UnrankVariant_CT (l_1, n, k)
17     v_2 := UnrankVariant_C (l_2, n, k)
18     v_3 := UnrankVariant_E (l_3, k, m)
19     v_4 := UnrankVariant_P (l_4, n − k)
20     v = (k, v_1, v_2, v_3, v_4)
21     return v
22 end
```

Таким образом получаем, что уменьшить вычислительную сложность исследуемого алгоритма за счет применения метода двоичного поиска для поиска точного решения неравенства (3.1), где частичные суммы вычисляются по формуле (3.23), не представляется возможным. Чтобы реализовать такую возможность, необходимо получить более простое с точки зрения вычислительной сложности выражение для расчета частичных сумм (3.23), либо нужно воспользоваться подходящими методами приближенных вычислений.

3.3.4 Множество путей Дика с пиками

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: путь Дика длины $2n$ с m пиками, образованный l путями Дика меньшей длины. Пик в пути Дика — это ситуация, когда после подъема совершается спуск.

Данный комбинаторный объект предлагается кодировать в виде последовательности $a = (a_1, \ldots, a_k)$, где каждое $a_i = (a_{i1}, a_{i2}, \ldots)$ представляет собой путь Дика, при этом $a_{ij} =$ "u" обозначает подъем, $a_{ij} =$ "d" — спуск.

На рисунке 3.9 показан пример всех возможных вариантов рассматриваемых путей Дика при $n = 4$, $m = 3$ и $l = 2$.

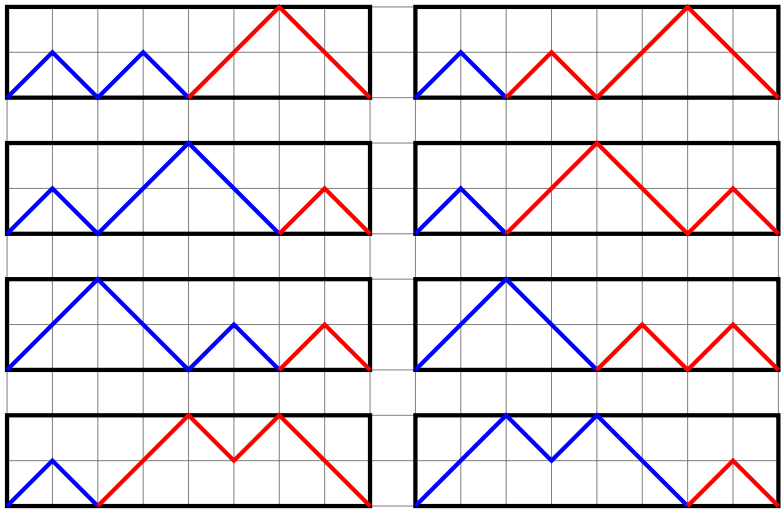


Рисунок 3.9 — Все пути Дика длины 8 с 3 пиками, образованные 2 путями Дика

Функция мощности данного комбинаторного множества определяется производящей функцией трех переменных

$$N(x,y,z) = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} N_{n,m,l}x^n y^m z^l = \frac{1}{1 - zN(x,y)} =$$

$$= \frac{2x - z\left(1 - x - xy - \sqrt{(1-x-xy)^2 - 4x^2y}\right)}{2x}, \qquad (3.24)$$

где $N(x,y)$ представляет собой производящую функцию для чисел Нараяны [58]

$$N(x,y) = \sum_{n>0}\sum_{m>0} N_{n,m}x^n y^m = \frac{1 - x - xy - \sqrt{(1-x-xy)^2 - 4x^2y}}{2x}.$$

Рассмотрим применение разработанного комплексного метода получения коэффициентов производящих функций многих переменных для случая производящей функции (3.24). Согласно данному методу, представим производящую функцию $N(x,y,z)$ как композицию более простых и известных производящих функций следующего вида:

$$N(x,y,z) = H(A(x,y,z)) = H(zN(x,y)),$$

где

$$H(x) = \sum_{n\geqslant 0} h_n x^n = \sum_{n\geqslant 0} x^n = \frac{1}{1-x}. \tag{3.25}$$

Применяя Теорему 11 для композиции $H(zN(x,y))$, получаем следующее явное выражение для ее коэффициентов:

$$N_{n,m,l} = \sum_{k=0}^{n+m+l} h_l A^{\Delta}(n,m,l,k), \tag{3.26}$$

где $A^{\Delta}(n,m,l,k)$ представляет собой композиту производящей функции $A(x,y,z)$. Рассмотрим k-ю степень производящей функции и ее коэффициенты, получим

$$A(x,y,z)^k = z^k N(x,y)^k = \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} A^{\Delta}(n,m,l,k)x^n y^m z^l =$$

$$= \sum_{n\geqslant 0}\sum_{m\geqslant 0}\sum_{l\geqslant 0} N^{\Delta}(n,m,k)\delta_{k,l}x^n y^m z^l, \tag{3.27}$$

где $N^{\Delta}(n,m,k)$ представляет собой композиту производящей функции $N(x,y)$.

Применяя (3.25) и (3.26) для вычисления (3.27), получаем

$$N_{n,m,l} = \sum_{k=0}^{n+m+l} N^{\Delta}(n,m,k)\delta_{k,l} = N^{\Delta}(n,m,l). \tag{3.28}$$

Следовательно, значения $N_{n,m,l}$ являются композитой производящей функции $N(x,y)$. Чтобы получить явную формулу для композиты производящей функции $N(x,y)$, воспользуемся следующим функциональным уравнением [58]:

$$-xy + (1-x-xy)N(x,y) - xN(x,y)^2 = 0. \tag{3.29}$$

Данное функциональное уравнение можно преобразовать к виду

$$N(x,y) = xG(N(x,y),y), \tag{3.30}$$

где производящая функция $G(x,y)$ имеет следующую форму:

$$G(x,y) = (1+x)(x+y).$$

Для уравнения (3.30) применим теорему Лагранжа, получим

$$[x^n]N(x,y)^k = \frac{k}{n}[x^{n-k}]G(x,y)^n. \tag{3.31}$$

Рассмотрим n-ю степень производящей функции $G(x,y)$

$$G(x,y)^n = (1+x)^n(x+y)^n. \tag{3.32}$$

Чтобы получить явное выражение для коэффициентов производящей функции $G(x,y)^n$, сначала воспользуемся биномом Ньютона, тогда

$$(1+x)^n = \sum_{k\geqslant 0}\binom{n}{k}x^k = \sum_{k\geqslant 0}\sum_{m\geqslant 0}\delta_{m,0}\binom{n}{k}x^k y^m, \tag{3.33}$$

$$(x+y)^n = \sum_{k\geqslant 0}\binom{n}{k}x^k y^{n-k} = \sum_{k\geqslant 0}\sum_{m\geqslant 0}\delta_{m,n-k}\binom{n}{k}x^k y^m. \tag{3.34}$$

Объединяя (3.33) и (3.34) для (3.32) и применяя правило умножения производящих функций, получим

$$G(x,y)^n = \sum_{k\geqslant 0}\sum_{m\geqslant 0}\binom{n}{m-n+k}\binom{n}{m}x^k y^m. \tag{3.35}$$

Таким образом, используя (3.35) в (3.31), получим явное выражение для композиты производящей функции $N(x,y)$, которое в то же время является явным выражением для коэффициентов производящей функции $N(x,y,z)$:

$$N^{\Delta}(n,m,l) = \frac{l}{n}\binom{n}{m-l}\binom{n}{m} = N_{n,m,l}. \tag{3.36}$$

Также на основе функционального уравнения (3.29) получим рекуррентное соотношение для значений $N_{n,m,l}$

$$N_{n,m,l} = N_{n-1,m-1,l-1} + N_{n-1,m-1,l} + N_{n-1,m,l} + N_{n-1,m,l+1} \tag{3.37}$$

для $n \geqslant m \geqslant l > 0$ и $N_{n,n,n} = 1$.

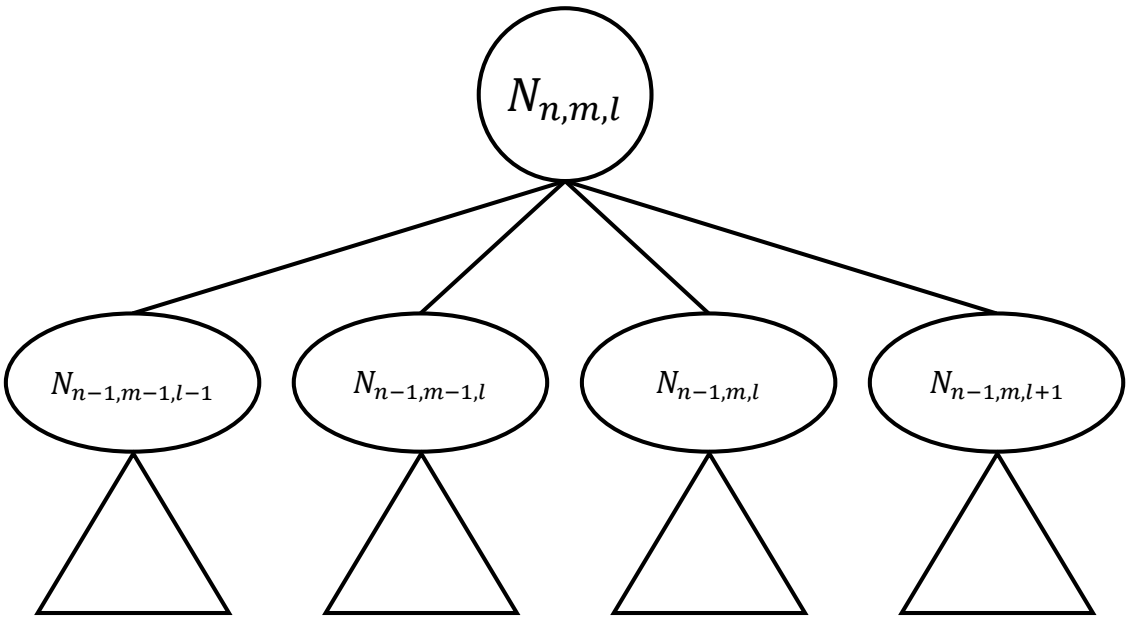


Рисунок 3.10 — Структура дерева И/ИЛИ для функции мощности (3.37)

Представление в виде структуры дерева И/ИЛИ

Так как функция мощности (3.37) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.10).

В таком случае биекция между исследуемыми путями Дика длины $2n$ с m пиками, образованными l путями Дика меньшей длины, и вариантами полученного дерева И/ИЛИ определяется набором следующих правил:

– если выбран первый сын ИЛИ-узла (поддерево узла, помеченного $N_{n-1,m-1,l-1}$), то к имеющемуся пути Дика длины $2(n-1)$ с $m-1$ пиками, образованному $l-1$ путями Дика меньшей длины, в начало добавляется путь Дика вида (”u”,”d”), тогда $a := ((\text{”}u\text{”},\text{”}d\text{”}),a_1,a_2,\ldots)$;

– если выбран второй сын ИЛИ-узла (поддерево узла, помеченного $N_{n-1,m-1,l}$), то к имеющемуся пути Дика длины $2(n-1)$ с $m-1$ пиками, образованному l путями Дика меньшей длины, в конец первого пути Дика $a_1 = (a_{11},\ldots,a_{1t})$ добавляется пик, тогда $a := ((a_{11},\ldots,a_{1t},\text{”}u\text{”},\text{”}d\text{”}),a_2,\ldots)$;

– если выбран третий сын ИЛИ-узла (поддерево узла, помеченного $N_{n-1,m,l}$), то к имеющемуся пути Дика длины $2(n-1)$ с m пиками, образованному l путями Дика меньшей длины, в первый путь Дика $a_1 = (a_{11},\ldots,a_{1t})$ добавляются шаги, которые не образуют пик, тогда $a := ((\text{”}u\text{”},a_{11},\ldots,a_{1t},\text{”}d\text{”}),a_2,\ldots)$;

– если выбран четвертый сын ИЛИ-узла (поддерево узла, помеченного $N_{n-1,m,l+1}$), то к имеющемуся пути Дика длины $2(n-1)$ с m пиками, образованному $l+1$ путями Дика меньшей длины, добавляются шаги, которые не образуют пик и объединяют первый путь Дика $a_1 = (a_{11}, \ldots, a_{1t})$ и второй путь Дика $a_2 = (a_{21}, \ldots, a_{2s})$ в один, тогда $a := ((a_{11}, \ldots, a_{1t}, "u", a_{21}, \ldots, a_{2s}, "d"), a_3, \ldots)$.

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде последовательности $v = (v_1, v_2, \ldots)$, где каждое $v_i \in \{1,2,3,4\}$ представляет собой номер выбранного сына ИЛИ-узла на i-ом уровне дерева И/ИЛИ.

Алгоритмы ранжирования и генерации по рангу

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества путей Дика с пиками (Алгоритм 11 и Алгоритм 12).

Алгоритм 11: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.10

```
1  RankVariant (v = (v_1, v_2, ...), n, m, l)
2  begin
3      if n = m = l then r := 0
4      else if v_1 = 1 then
5          r := S_0 + RankVariant ((v_2, ...), n − 1, m − 1, l − 1)
6      else if v_1 = 2 then
7          r := S_1 + RankVariant ((v_2, ...), n − 1, m − 1, l)
8      else if v_1 = 3 then
9          r := S_2 + RankVariant ((v_2, ...), n − 1, m, l)
10     else if v_1 = 4 then
11         r := S_3 + RankVariant ((v_2, ...), n − 1, m, l + 1)
12     return r
13 end
```

Алгоритм 12: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.10

```
1  UnrankVariant (r, n)
2  begin
3      if n = m = l then v := ()
4      else if r < S_1 then
5          v := concat ((1),UnrankVariant (r − S_0, n − 1, m − 1, l − 1))
6      else if r < S_2 then
7          v := concat ((2),UnrankVariant (r − S_1, n − 1, m − 1, l)
8      else if r < S_3 then
9          v := concat ((3),UnrankVariant (r − S_2, n − 1, m, l)
10     else if r < S_4 then
11         v := concat ((4),UnrankVariant (r − S_3, n − 1, m, l + 1)
12     return v
13 end
```

В данных алгоритмах используется функция конкатенации `concat`, которая выполняет склеивание двух последовательностей в одну. Также здесь для расчетов применяются частичные суммы следующего вида:

$$
\begin{aligned}
&S_0 = 0,\\
&S_1 = S_0 + N_{n-1,m-1,l-1} = N_{n-1,m-1,l-1},\\
&S_2 = S_1 + N_{n-1,m-1,l} = N_{n-1,m-1,l-1} + N_{n-1,m-1,l},\\
&S_3 = S_2 + N_{n-1,m,l} = N_{n-1,m-1,l-1} + N_{n-1,m-1,l} + N_{n-1,m,l},\\
&S_4 = S_3 + N_{n-1,m,l+1} = N_{n-1,m-1,l-1} + N_{n-1,m-1,l} + N_{n-1,m,l} + N_{n-1,m,l+1}.
\end{aligned}
$$

Оценка вычислительной сложности разработанных алгоритмов ранжирования и генерации по рангу равна $O(nm)$, что определяется наличием максимум n рекурсивных вызовов и вычислительной сложностью $O(m)$ для расчета значения $N_{n,m,l}$ на основе формулы (3.36) при поиске частичных сумм S_k. Если частичные суммы S_k будут предварительно вычислены и храниться в памяти, то вычислительная сложность данных алгоритмов примет линейный вид $O(n)$.

3.3.5 Множество последовательностей вариантов ответа на тест с вопросами закрытого типа

Вопросы открытого и закрытого типа широко применяются при проведения различных тестирований и опросов. К вопросам открытого типа относятся вопросы, требующие составления собственных вариантов ответа на них, тогда как в вопросах закрытого типа можно ответить, выбрав вариант из предложенного ограниченного списка. Закрытые тесты (тесты, содержащие только вопросы закрытого типа) широко применяются организациями различного рода: образовательными, предпринимательскими, общественными и государственными. Основная цель таких опросников — контроль или оценка знаний, умений и других характеристик на разных этапах обучения. За счет фиксированного количества возможных вариантов ответа на вопросы закрытого типа появляется возможность автоматизировать процессы хранения, передачи и обработки таких данных [231; 232]. При этом, если обрабатывается большой объем таких данных, то особенно важной становится проблема их кодирования и хранения.

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: последовательность из не менее t правильных ответов на закрытый тест, который содержит n вопросов с m вариантами ответа на каждый из них (только один вариант ответа является правильным, остальные $m-1$ являются неправильными).

Данный комбинаторный объект предлагается кодировать в виде последовательности $a = (a_1, \ldots, a_n)$, где каждое $a_i \in \{0, 1, \ldots, m-1\}$ представляет собой выбор ответа на i-й вопрос закрытого теста, при этом $a_i = 0$ соответствует случаю правильного ответа на i-й вопрос, $a_i > 0$ соответствует случаю неправильного ответа (значение a_i показывает позицию выбранного ответа среди $m-1$ неправильных ответов).

На рисунке 3.11 показан пример графического представления закрытого теста, который содержит n вопросов с m вариантами ответа на каждый из них.

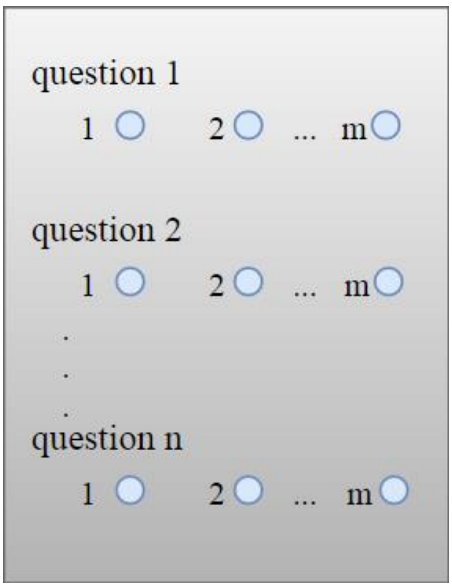


Рисунок 3.11 — Пример графического представления закрытого теста

Для рассматриваемого комбинаторного множества последовательностей вариантов ответа на закрытый тест получено следующее явное выражение для функции мощности:

$$A(n,m,t) = \sum_{k=t}^{n} C_n^k (m-1)^{n-k}, \tag{3.38}$$

где C_n^k — это количество сочетаний из n по k [2].

Используя явное выражение (3.38), можно рассчитать общее количество последовательностей вариантов ответа на закрытый тест, которые удовлетворяют указанным выше требованиям. В данной формуле значение k показывает количество правильных ответов на n вопросов. Исходя из описания комбинаторного объекта, это значение должно быть не менее t, то есть $t \leqslant k \leqslant n$. Затем для фиксированного значения k нужно определить на какие именно k вопросов были даны правильные ответы (количество возможных способов выбора k вопросов из n определяется количеством сочетаний из n по k). На оставшиеся $n-k$ вопросов нужно ответить неправильно (количество возможных вариантов неправильного ответа равно $m-1$, так как только один из m вариантов ответа является правильным).

Представление в виде структуры дерева И/ИЛИ

Так как функция мощности (3.38) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.12).

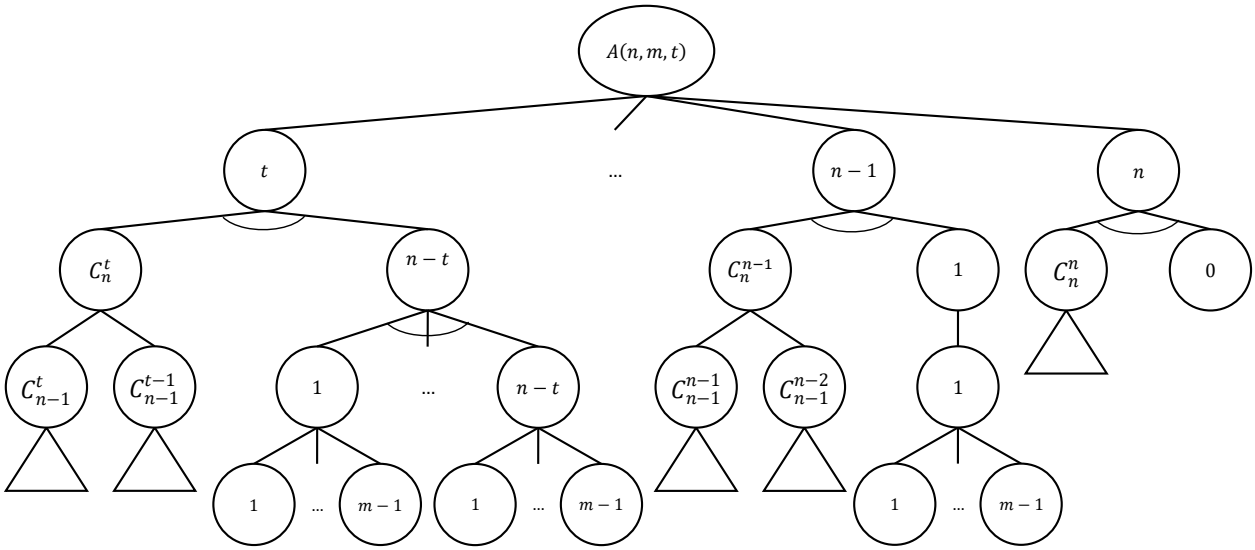


Рисунок 3.12 — Структура дерева И/ИЛИ для функции мощности (3.38)

В таком случае биекция между последовательностями из не менее t правильных ответов на закрытый тест, который содержит n вопросов с m вариантами ответа на каждый из них, и вариантами полученного дерева И/ИЛИ определяется набором следующих правил:

– количество правильных ответов на n вопросов определяется значением параметра k (выбранный сын ИЛИ-узла, помеченного $A(n,m,t)$), что также соответствует количеству нулей в последовательности $a = (a_1, \ldots, a_n)$;

– левый сын И-узла, помеченный C_n^k, определяет k вопросов из n, на которые был дан правильный ответ (для данного поддерева каждый выбранный левый сын, помеченный C_{n-1}^k, определяет ситуацию неправильного ответа на очередной вопрос, а каждый правый сын, помеченный C_{n-1}^{k-1}, определяет ситуацию неправильного ответа), что также определяет месторасположение нулей в последовательности $a = (a_1, \ldots, a_n)$;

– правый сын И-узла, помеченный $n - k$, определяет варианты выбранных неправильных ответов на оставшиеся $n - k$ вопросов (имеется $m - 1$ возможных вариантов неправильного ответа для каждого из $n - k$ вопросов).

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде последовательности $v = (k, vl, vr)$, в которой:

– k определяет метку выбранного сына ИЛИ-узла, помеченного $A(n,m,t)$;

– $vl = (v_1, v_2, \ldots)$ соответствует варианту поддерева узла, помеченного C_n^k (данный вариант представляется в виде последовательности выбранных сыновей в поддереве, где $v_i = 0$ соответствует выбору левого сына и $v_i = 1$ соответствует выбору правого сына);

– $vr = (v_1, v_2, \ldots, v_{n-k})$ соответствует варианту поддерева узла, помеченного $n - k$ (данный вариант представляется в виде последовательности выбранных листьев в поддереве, где $v_i \in \{1, \ldots, m-1\}$).

На рисунке 3.13 показан пример варианта дерева И/ИЛИ для функции мощности (3.38) при $n = 4$, $m = 5$ и $t = 2$. Данный вариант дерева И/ИЛИ кодируется последовательностью $v = (2, (0,1,0), (3,2))$.

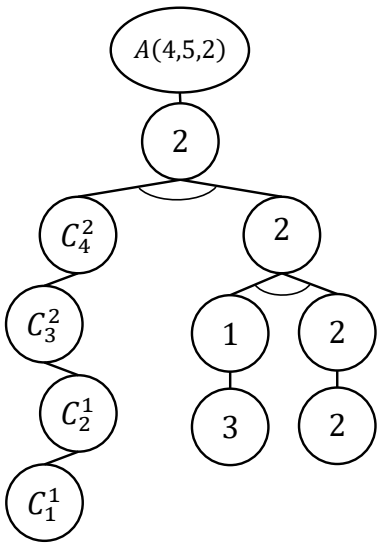


Рисунок 3.13 — Пример варианта дерева И/ИЛИ для функции мощности (3.38) при $n = 4$, $m = 5$ и $t = 2$

На основе разработанных правил биекции определим какой именно объект $a = (a_1, a_2, a_3, a_4)$ соответствует варианту дерева И/ИЛИ $v = (2, (0,1,0), (3,2))$, представленному на рисунке 3.13:

1. Первое значение 2 показывает количество правильных ответов, то есть имеется 2 нуля среди значений a_i;

2. Последовательность (0,1,0) показывает, что:

– на первый вопрос был дан неправильный ответ ($v_1 = 0$), то есть $a_1 \neq 0$;

– на второй вопрос был дан правильный ответ ($v_2 = 1$), то есть $a_2 = 0$;

– на третий вопрос был дан неправильный ответ ($v_3 = 0$), то есть $a_3 \neq 0$;

– остался один вопрос (четвертый вопрос) и осталось выбрать один вопрос, на который нужно дать правильный ответ, то есть $a_4 = 0$;

3. Последовательность (3,2) показывает, что:

– на первый вопрос, на который был дан неправильный ответ (a_1), необходимо выбрать третий неправильный вариант ответа ($v_1 = 3$), то есть $a_1 = 3$;

– на второй вопрос, на который был дан неправильный ответ (a_3), необходимо выбрать второй неправильный вариант ответа ($v_2 = 2$), то есть $a_3 = 2$.

В результате получаем, что варианту дерева И/ИЛИ $v = (2, (0,1,0),(3,2))$ соответствует объект $a = (3, 0, 2, 0)$.

Алгоритмы ранжирования и генерации по рангу

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества последовательностей вариантов ответа на тест с вопросами закрытого типа (Алгоритм 13 и Алгоритм 14).

Алгоритм 13: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.11

```
1  RankVariant (v = (k, vl, vr), n, m, t)
2  begin
3      if k = n then r := Σ_{i=t}^{k-1} C_n^i (m-1)^{n-i}
4      else
5          l_1 := RankVariant_C (vl, n, k)
6          l_2 := vr_{n-k} - 1
7          for i := n - k - 1 to 1 do l_2 := vr_i - 1 + (m-1) l_2
8          r := l_1 + C_n^k l_2 + Σ_{i=t}^{k-1} C_n^i (m-1)^{n-i}
9      end
10     return r
11 end
```

Алгоритм 14: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.11

```
1  UnrankVariant (r, n, m, t)
2  begin
3      k := t
4      sum := 0
5      while sum + C_n^k (m − 1)^{n−k} ⩽ r do
6          sum := sum + C_n^k (m − 1)^{n−k}
7          k := k + 1
8      end
9      r := r − sum
10     l_1 := r  mod C_n^k
11     l_2 := ⌊ r / C_n^k ⌋
12     vl := UnrankVariant_C (l_1, n, k)
13     for i := 1 to n − k do
14         vr_i := (l_2  mod (m − 1)) + 1
15         l_2 := ⌊ l_2 / (m−1) ⌋
16     end
17     v = (k, vl, vr)
18     return v
19 end
```

Например, используя алгоритм ранжирования для варианта дерева И/ИЛИ $v = (2, (0,1,0), (3,2))$, получим соответствующий ему ранг $r = 37$.

Оценка вычислительной сложности разработанных алгоритмов ранжирования и генерации по рангу равна $O((n-t)^2)$, что определяется расчетом значений функции мощности по формуле (3.38). Также отметим, что в алгоритме генерации по рангу (Алгоритм 14) выполняются вычисления частичных сумм следующего вида:

$$S_k = \sum_{i=t}^{k} C_n^i (m-1)^{n-i}.$$

3.3.6 Множество исходов турнира на выбывание

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: исход событий турнира на выбывание при участии в нем 2^{n-1} игроков.

Данный комбинаторный объект предлагается кодировать в виде последовательности $a = (a_1,a_2,\ldots,a_{2^{n-1}-1})$, элементы которой отражают левосторонний обход дерева победителей каждого матча в виде турнирной сетки, где каждое $a_i \in \{1,2,\ldots,2^{n-1}\}$ показывает порядковый номер участника, при этом a_1 показывает победителя финала, a_2 — победителя первого полуфинала, $a_{2^{n-2}+1}$ — победителя второго полуфинала, и т.д.

В таблице 3.2 показан пример всех возможных исходов событий турнира на выбывание при участии в нем 4 игроков (так как $2^{n-1} = 4$, то получаем $n = 3$).

Таблица 3.2 — Все исходы событий турнира на выбывание при участии в нем 4 игроков

1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1		3		1		4		2		3		2		4	
1				1				2				2			

1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1		3		1		4		2		3		2		4	
3				4				3				4			

Значения функции мощности данного комбинаторного множества формируют следующую целочисленную последовательность (последовательность $A058891$ в OEIS [94]):

$$1, 2, 8, 128, 32768, 2147483648, 9223372036854775808, \ldots$$

Значения данной последовательности задаются следующей формулой:

$$A(n) = 2^{2^{n-1}-1}.$$

Данная формула имеет сходство с формулой расчета количества последовательностей де Брейна порядка n при использовании двоичного алфавита (последовательность $A016031$ в OEIS [94]).

Представление в виде структуры дерева И/ИЛИ

Известна следующая функция мощности для комбинаторного множества исходов событий турнира на выбывание при участии в нем 2^{n-1} игроков:

$$A(n) = 2A(n-1)^2, \quad A(1) = 1. \tag{3.39}$$

Так как функция мощности (3.39) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.14).

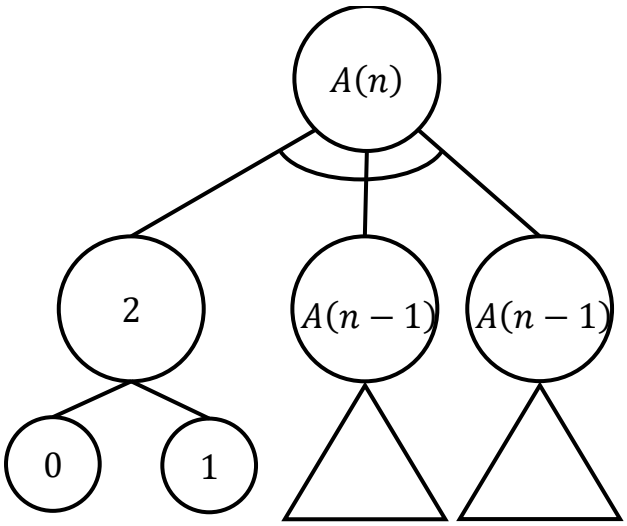


Рисунок 3.14 — Структура дерева И/ИЛИ для функции мощности (3.39)

В таком случае биекция между исходами событий турнира на выбывание при участии в нем 2^{n-1} игроков и вариантами полученного дерева И/ИЛИ определяется набором следующих правил:

– выбор левого (или правого) листа ИЛИ-узла, помеченного 2, соответствует победе первого (или второго) участника в матче между двух участников;

– поддерево левого узла, помеченного $A(n-1)$, определяет первого участника в матче между двух участников (им является один из победителей предыдущего раунда турнирной сетки);

– поддерево правого узла, помеченного $A(n-1)$, определяет второго участника в матче между двух участников (им является один из победителей предыдущего раунда турнирной сетки).

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде последовательности $v = (v_1,v_2,\ldots,v_{2^{n-1}-1})$ выбранных листов ИЛИ-узла в рамках левостороннего обхода варианта дерева И/ИЛИ, то есть $v_i = 0$ при выборе левого листа ИЛИ-узла и $v_i = 1$ при выборе правого листа ИЛИ-узла.

Алгоритмы ранжирования и генерации по рангу

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества исходов событий турнира на выбывание при участии в нем 2^{n-1} игроков (Алгоритм 15 и Алгоритм 16).

Оценка вычислительной сложности разработанных алгоритмов ранжирования и генерации по рангу равна $O(4^n)$, что определяется наличием 2^n рекурсивных вызовов и вычислительной сложностью $O(2^n)$ для расчета значения $A(n)$ на основе формулы (3.39).

Алгоритм 15: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.14

```
1  RankVariant (v = (v_1,v_2,...,v_{2^{n-1}-1}), n)
2  begin
3      if n = 1 then r := 0
4      else
5          l_1 := v_1
6          l_2 := RankVariant ((v_2,...,v_{2^{n-2}}), n − 1)
7          l_3 := RankVariant ((v_{2^{n-2}+1},...,v_{2^{n-1}-1}), n − 1)
8          r := l_1 + 2(l_2 + A(n − 1)l_3)
9      end
10     return r
11 end
```

Алгоритм 16: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.14

```
1  UnrankVariant (r, n)
2  begin
3      if n = 1 then v := ()
4      else
5          l_1 := r  mod 2
6          r := ⌊r/2⌋
7          l_2 := r  mod A(n − 1)
8          l_3 := ⌊r/A(n−1)⌋
9          vl := UnrankVariant (l_2, n − 1)
10         vr := UnrankVariant (l_3, n − 1)
11         v := concat ((l_1), vl, vr)
12     end
13     return v
14 end
```

3.3.7 Множество частей круга, полученных при разрезе его поверхности прямыми линиями

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: часть круга, полученная при разрезе его поверхности n прямыми линиями таким образом, чтобы количество получаемых частей было максимально возможным. При таком способе разреза поверхности круга каждая n-я линия создает n новых частей круга.

Данный комбинаторный объект предлагается кодировать в виде пары $a = (a_1,a_2)$, где:

– a_1 показывает порядковый номер линии, которая последней участвовала в процессе создания требуемой части круга (то есть $a_1 \in \{1,\ldots,n\}$);

– a_2 показывает порядковый номер для требуемой части круга среди n созданных данной линией новых частей круга (то есть $a_2 \in \{1,\ldots,n\}$).

Чтобы задать очередность нумерации частей круга, получаемых после проведения каждого разреза в виде прямой линии, определим на окружности круга некоторую начальную точку. Тогда новыми частями круга будут считаться те части, которые находятся за линией, если наблюдать со стороны начальной точки. Нумерация старых частей, которые уменьшились за счет проведения новой линии, сохраняется прежней. Для нумерации новых частей необходимо от начальной точки пройти вдоль окружности по направлению часовой стрелки до первого касания с новой линией, а затем новые части круга нумеруются в порядке их встречи вдоль прохождения данной линии.

Значения функции мощности данного комбинаторного множества формируют следующую целочисленную последовательность (последовательность $A000124$ в OEIS [94]):

$$1, 2, 4, 7, 11, 16, 22, 29, 37, 46, 56, 67, 79, 92, 106, 121, 137, \ldots$$

Значения данной последовательности задаются следующей формулой:

$$P(n) = \frac{n(n+1)}{2} + 1.$$

Значения $P(n)$ также называют центральными многоугольными числами (или последовательность ленивого поставщика [57; 233]), которые имеют целый перечень комбинаторных интерпретаций. Во-первых, центральные многоугольные числа относятся к исследуемой задаче размещения прямых линий на поверхности плоской фигуры. Также они используются для оценки сложности цилиндрического алгебраического разложения [234]. Кроме того, центральные многоугольные числа связаны с треугольными числами (последовательность $A000217$ в OEIS [94]), и основное различие между ними состоит в том, что центральные многоугольные числа всегда больше на 1.

Центральные многоугольные числа определяются следующей производящей функцией, которая была введена Плуффом [94]:

$$F(x) = \sum_{n \geqslant 0} P(n)\, x^n = \frac{1 - x + x^2}{(1-x)^3}.$$

Также центральные многоугольные числа определяются следующей экспоненциальной производящей функцией, которая была введена Критцером [94]:

$$E(x) = \sum_{n \geqslant 0} \frac{P(n)}{n!}\, x^n = \left(1 + x + \frac{x^2}{2}\right) e^x.$$

Представление в виде структуры дерева И/ИЛИ

Известна следующая функция мощности для комбинаторного множества частей круга, полученных при разрезе его поверхности n прямыми линиями:

$$P(n) = n + P(n-1), \quad P(0) = 1. \tag{3.40}$$

Так как функция мощности (3.40) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.15).

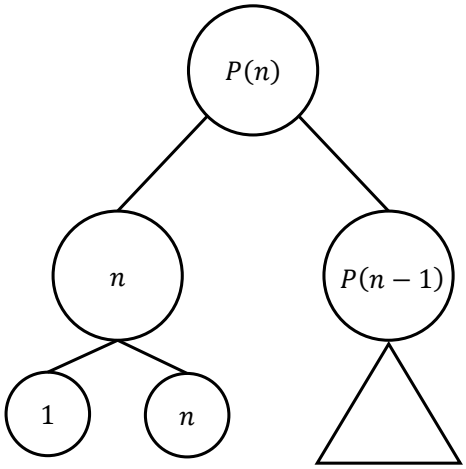


Рисунок 3.15 — Структура дерева И/ИЛИ для функции мощности (3.40)

В таком случае биекция между частями круга, полученными при разрезе его поверхности n прямыми линиями, и вариантами полученного дерева И/ИЛИ определяется набором следующих правил:

– выбор левого сына ИЛИ-узла, помеченного $P(n)$, показывает, что n-я линия последней участвовала в процессе создания требуемой части круга (то есть $v_1 = a_1$), а выбор в нем листа показывает порядковый номер для требуемой части круга среди n созданных данной линией новых частей (то есть $v_2 = a_2$);

– выбор правого сына ИЛИ-узла, помеченного $P(n)$, показывает, что необходимо исключить из дальнейшего рассмотрения все n созданных n-й линией новых частей и перейти к выбору частей, образованных $(n-1)$-й линией.

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде пары $v = (v_1, v_2)$, где:

– v_1 соответствует метке ИЛИ-узла $P(v_1)$, в рамках которого выбран его левый сын;

– v_2 соответствует метке выбранного листа.

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества частей круга, полученных при разрезе его поверхности n прямыми линиями (Алгоритм 17 и Алгоритм 18).

Алгоритм 17: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.15

```
1 RankVariant (v = (v1,v2), n)
2 begin
3 |  if n = 0 then r := 0
4 |  else
5 |  |  if n = v1 then r := v2 − 1
6 |  |  else r := RankVariant (v, n − 1)
7 |  end
8 |  return r
9 end
```

Алгоритм 18: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.15

```
1 UnrankVariant (r, n)
2 begin
3 |  if n = 0 then v := (0,0)
4 |  else
5 |  |  if r < n then v := (n,r)
6 |  |  else v := UnrankVariant (r − n, n − 1)
7 |  end
8 |  return v
9 end
```

Оценка вычислительной сложности разработанных комбинаторных алгоритмов равна $O(n)$, что определяется наличием n рекурсивных вызовов.

3.3.8 Множество последовательностей правильно вложенных скобок, разряженных нулями

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: последовательность длины n, представляющая собой последовательность правильно вложенных скобок, разряженных нулями.

Данный комбинаторный объект предлагается кодировать в виде последовательности $a = (a_1,a_2,\ldots,a_n)$, где $a_i \in \{0,"(",")"\}$.

Далее показан пример всех возможных последовательностей длины 4, представляющих собой последовательности правильно вложенных скобок, разряженных нулями:

$$(()),\quad ()(),\quad ()00,\quad (0)0,\quad (00),\quad 0()0,\quad 0(0),\quad 00(),\quad 0000.$$

Значения функции мощности данного комбинаторного множества формируют следующую целочисленную последовательность (последовательность $A001006$ в OEIS [94]):

$$1, 1, 2, 4, 9, 21, 51, 127, 323, 835, 2188, 5798, 15511, 41835, \ldots$$

Значения данной последовательности задаются следующей формулой:

$$M_n = \sum_{k=0}^{n} \frac{1}{k+1}\binom{2k}{k}\binom{n}{2k}. \tag{3.41}$$

Данный комбинаторный объект представляет собой одну из известных комбинаторных интерпретаций для чисел Моцкина.

Представление в виде структуры дерева И/ИЛИ

Известна следующая функция мощности для комбинаторного множества последовательностей длины n, представляющих собой последовательности правильно вложенных скобок, разряженных нулями:

$$M_n = M_{n-1} + \sum_{k=0}^{n-2} M_k M_{n-2-k}, \quad M_0 = M_1 = 1. \tag{3.42}$$

Так как функция мощности (3.42) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.16).

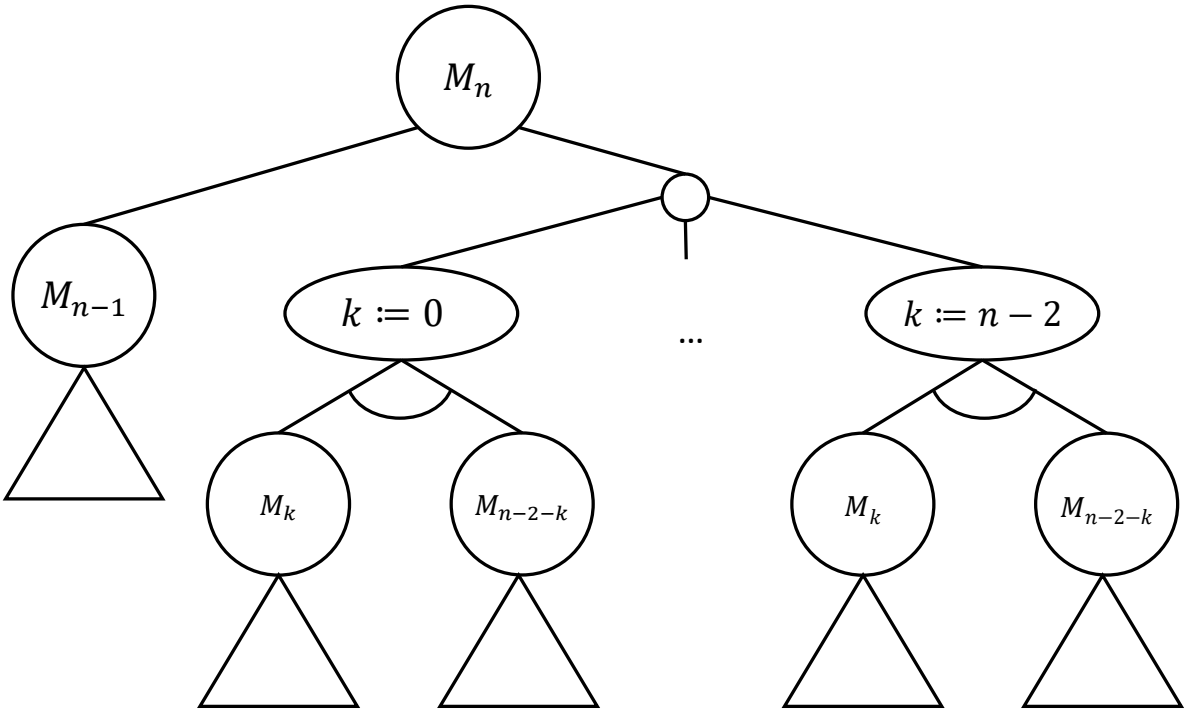


Рисунок 3.16 — Структура дерева И/ИЛИ для функции мощности (3.42)

В таком случае биекция между последовательностями длины n, представляющими собой последовательности правильно вложенных скобок, разряженных нулями, и вариантами полученного дерева И/ИЛИ определяется набором следующих правил:

– выбор левого сына ИЛИ-узла, помеченного M_n, соответствует добавлению нуля в искомую последовательность a, которая представляется в виде $a = 0b$, где b является последовательностью правильно вложенных скобок, разряженной нулями, и состоит из $n - 1$ символов;

– выбор правого сына ИЛИ-узла, помеченного M_n, соответствует добавлению пары открывающейся и закрывающейся скобок в искомую последовательность a, которая представляется в виде $a = (b)c$, где b и c также являются последовательностями правильно вложенных скобок, разряженных нулями (b определяется вариантом vl и состоит из k символов, c определяется вариантом vr и состоит из $n - 2 - k$ символов).

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде пары $v = (v_1, v_2)$, где:

– v_1 соответствует выбору левого ($v_1 = 0$) или правого ($v_1 = 1$) сына ИЛИ-узла, помеченного M_n;

– если $v_1 = 0$, то v_2 соответствует варианту поддерева узла, помеченного M_{n-1};

– если $v_1 = 1$, то v_2 представляет собой тройку $v_2 = (k, vl, vr)$, где k соответствует метке выбранного сына для правого сына ИЛИ-узла, vl соответствует варианту поддерева узла, помеченного M_k, vr соответствует варианту поддерева узла, помеченного M_{n-2-k}.

Алгоритмы ранжирования и генерации по рангу

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества последовательностей длины n, представляющих собой последовательности правильно вложенных скобок, разряженных нулями (Алгоритм 19 и Алгоритм 20).

Алгоритм 19: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.16

```
1  RankVariant (v = (v1,v2), n)
2  begin
3      if n = 0 then r := 0
4      else
5          if v1 = 0 then r := RankVariant (v2, n − 1)
6          else
7              (k,vl,vr) := v2
8              l1 := RankVariant (vl, k)
9              l2 := RankVariant (vr, n − 2 − k)
10             r := l + l1 + Mk l2 + Mn−1 + Σ_{i=0}^{k−1} Mi Mn−2−i
11         end
12     end
13     return r
14 end
```

Алгоритм 20: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.16

```
1  UnrankVariant (r, n)
2  begin
3      if n = 0 then v := ()
4      else
5          if r < M_{n-1} then v := (0,UnrankVariant (r, n − 1))
6          else
7              r := r − M_{n-1}
8              k := 0
9              sum := 0
10             while sum + M_k M_{n-2-k} ⩽ r do
11                 sum := sum + M_k M_{n-2-k}
12                 k := k + 1
13             end
14             r := r − sum
15             l_1 := r  mod M_k
16             l_2 := ⌊r / M_k⌋
17             vl := UnrankVariant (l_1, k)
18             vr := UnrankVariant (l_2, n − 2 − k)
19             v := (1,(k,vl,vr))
20         end
21     end
22     return v
23 end
```

Оценка вычислительной сложности разработанных алгоритмов ранжирования и генерации по рангу равна $O(n) \cdot O_S(n)$, что определяется наличием максимум n рекурсивных вызовов и вычислительной сложностью $O_S(k)$ для расчета значений частичных сумм вида

$$S_k = \sum_{i=0}^{k-1} M_i M_{n-2-i}.$$

3.3.9 Множество разбиений множества

Описание комбинаторного множества

Рассмотрим разработку алгоритмов комбинаторной генерации для следующего комбинаторного объекта: разбиение множества n элементов на непересекающиеся непустые подмножества.

Данный комбинаторный объект предлагается кодировать в виде последовательности $a = (a_1,a_2,\ldots)$, где каждое a_i представляет собой последовательность лексикографически упорядоченных элементов i-го подмножества, при этом сами последовательности a_i также лексикографически упорядочены.

Далее показан пример всех возможных разбиений множества 3 элементов на непересекающиеся непустые подмножества:

$$\{1\},\{2\},\{3\};\quad \{1\},\{2,3\};\quad \{2\},\{1,3\};\quad \{3\},\{1,2\};\quad \{1,2,3\}.$$

Значения функции мощности данного комбинаторного множества формируют следующую целочисленную последовательность (последовательность $A000110$ в OEIS [94]):

$$1,1,2,5,15,52,203,877,4140,21147,115975,678570,4213597,\ldots$$

Значения данной последовательности задаются следующей формулой:

$$B_n = \sum_{k=0}^{n}\sum_{i=0}^{k}\frac{(-1)^i(k-i)^n}{i!(k-i)!}. \tag{3.43}$$

Данный комбинаторный объект представляет собой одну из известных комбинаторных интерпретаций для чисел Белла. Также числа Белла могут быть вычислены через сумму чисел Стирлинга второго рода

$$B_n = \sum_{k=0}^{n} S(n,k),$$

а также быть определены через экспоненциальную производящую функцию

$$F(x) = \sum_{n\geqslant 0}\frac{B_n}{n!}\,x^n = e^{e^x-1}.$$

Представление в виде структуры дерева И/ИЛИ

Известна следующая функция мощности для комбинаторного множества разбиений множества n элементов на непересекающиеся непустые подмножества:

$$B_n = \sum_{k=0}^{n-1} C_{n-1}^k B_k, \quad B_0 = 1. \tag{3.44}$$

Так как функция мощности (3.44) принадлежит алгебре $\{\mathbb{N}, +, \times, R\}$, то на ее основе можно построить структуру дерева И/ИЛИ (рисунок 3.17).

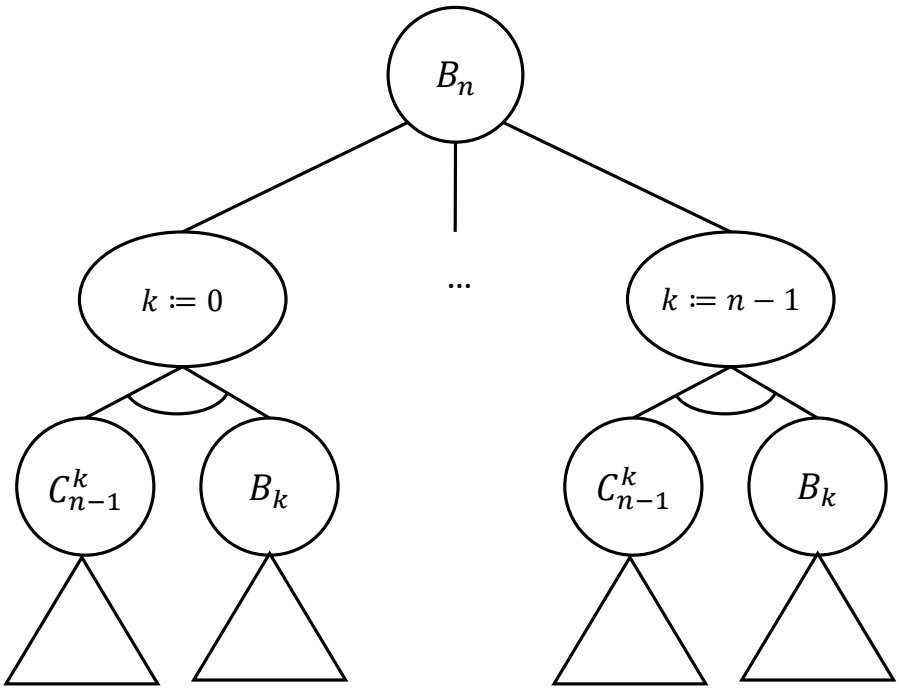


Рисунок 3.17 — Структура дерева И/ИЛИ для функции мощности (3.44)

В таком случае биекция между разбиениями множества n элементов на непересекающиеся непустые подмножества и вариантами полученного дерева И/ИЛИ определяется набором следующих правил:

– узел, помеченный k, показывает сколько элементов стоит отдельно от элемента со значением n;

– узел, помеченный C_{n-1}^k, определяет какие именно k элементов из оставшихся $n-1$ элементов множества будут стоять отдельно от элемента со значением n. Оставшиеся $n-1-k$ элементов записываются в одно подмножество с элементом со значением n;

– узел, помеченный B_k, определяет на какие подмножества будут разбиты k элементов, стоящие отдельно от элемента со значением n.

Для компактности представления каждый вариант дерева И/ИЛИ предлагается кодировать в виде последовательности $v = ((k_1, vc_1), (k_2, vc_2), \ldots)$, где:

– k_1 соответствует метке выбранного сына ИЛИ-узла, помеченного B_n;

– $vc_1 = (v_1, v_2, \ldots)$ соответствует варианту поддерева узла, помеченного $C_{n-1}^{k_1}$;

– аналогичные действия повторяются для поддерева узла, помеченного B_{k_1}, для получения оставшейся части $((k_2, vc_2), \ldots)$.

Алгоритмы ранжирования и генерации по рангу

На основе полученной структуры дерева И/ИЛИ с помощью общих алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ (Алгоритм 1 и Алгоритм 2) разработаны соответствующие алгоритмы для комбинаторного множества разбиений множества n элементов на непересекающиеся непустые подмножества (Алгоритм 21 и Алгоритм 22).

Алгоритм 21: Алгоритм ранжирования вариантов дерева И/ИЛИ, представленного на рисунке 3.17

```
1  RankVariant (v = ((k1,vc1),(k2,vc2),...), n)
2  begin
3      if n = 0 then r := 0
4      else
5          l1 := RankVariant_C (vc1, n − 1, k1)
6          l2 := RankVariant (((k2,vc2),...), k1)
7          r := l1 + C_{n−1}^{k1} l2 + Σ_{i=0}^{k1−1} C_{n−1}^{i} B_i
8      end
9      return r
10 end
```

Алгоритм 22: Алгоритм генерации по рангу вариантов дерева И/ИЛИ, представленного на рисунке 3.17

```
1  UnrankVariant (r, n)
2  begin
3      if n = 0 then v := ()
4      else
5          k := 0
6          sum := 0
7          while sum + C_{n-1}^k B_k ⩽ r do
8              sum := sum + C_{n-1}^k B_k
9              k := k + 1
10         end
11         r := r − sum
12         l_1 := r mod C_{n-1}^k
13         l_2 := ⌊ r / C_{n-1}^k ⌋
14         vc := UnrankVariant_C (l_1, n − 1, k)
15         v := concat ((k,vc), UnrankVariant (l_2, k)
16     end
17     return v
18 end
```

Оценка вычислительной сложности разработанных алгоритмов ранжирования и генерации по рангу равна $O(n) \cdot O_S(n)$, что определяется наличием максимум n рекурсивных вызовов и вычислительной сложностью $O_S(k)$ для расчета значений частичных сумм вида

$$S_k = \sum_{i=0}^{k} C_{n-1}^i B_i.$$

3.4 Выводы по главе

Основным результатом данной главы является предложенный модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, который отличается от оригинального метода и его модификаций применением разработанного комплексного метода получения явных выражений коэффициентов производящих функций многих переменных для нахождения выражения функции мощности комбинаторного множества, в том числе определяемого несколькими параметрами. Также предложенный модифицированный метод отличается применением методов приближенных вычислений и метода двоичного поиска для поиска выбранного сына ИЛИ-узла, что позволяет снижать вычислительную сложность алгоритмов генерации по рангу.

С целью апробации предложенного модифицированного метода построения алгоритмов комбинаторной генерации, были разработаны новые алгоритмы ранжирования и генерации по рангу для целого набора комбинаторных множеств. В частности, рассмотрено применение предложенных подходов к уменьшению вычислительной сложности алгоритмов генерации по рангу за счет использования методов приближенных вычислений и метода двоичного поиска. Для этого была представлена модификация алгоритма генерации по рангу для классического комбинаторного множества сочетаний из n по m в колексикографическом порядке с применением методов приближенных вычислений для поиска выбранного сына ИЛИ-узла. Вычислительная сложность модифицированного Алгоритма 4 с использованием Алгоритма 5 для предварительного поиска значения k составляет $O(m \cdot (m + \varepsilon m)) \approx O(m^2)$ (предполагая выполнение алгебраических операций с числами с временной сложностью $O(1)$). В отличие от оригинального алгоритма, полученный алгоритм можно эффективно применять при больших n и малых m, поскольку время работы этого алгоритма зависит только от параметра m. Также были разработаны новые алгоритмы ранжирования и генерации по рангу для комбинаторного множества самонепересекающихся решеточных путей на плоскости. В данном случае применение метода двоичного поиска для поиска выбранного сына ИЛИ-узла позволяет сократить в среднем количество требуемых вычислительных операций и получить вычислительную сложность $O(n \log_2 n)$, что является лучшим значением по сравнению с исходной версией алгоритма с вычислительной сложностью $O(n^2)$.

Кроме того, рассмотрено применение разработанного комплексного метода получения явных выражений коэффициентов производящих функций многих переменных для нахождения выражения функции мощности комбинаторного множества. Для этого было рассмотрено комбинаторное множество помеченных путей Дика длины $2n$ с m подъемами на возвратных шагах, функция мощности которого определяется производящей функцией двух переменных. Применяя Теорему 5 для производящей функции исследуемого комбинаторного множества, было получено явное выражение (3.18) для ее коэффициентов. Полученное выражение функции мощности комбинаторного множества принадлежит требуемой алгебре $\{\mathbb{N}, +, \times, R\}$, что позволило на ее основе построить структуру дерева И/ИЛИ и разработать алгоритмы комбинаторной генерации. Также было рассмотрено комбинаторное множество путей Дика с пиками, функция мощности которого определяется производящей функцией трех переменных. Применяя Теорему 11 для производящей функции исследуемого комбинаторного множества, было получено явное выражение (3.36) и рекуррентное соотношение (3.37) для ее коэффициентов. Полученное рекуррентное выражение функции мощности комбинаторного множества принадлежит требуемой алгебре $\{\mathbb{N}, +, \times, R\}$, что позволило на ее основе построить структуру дерева И/ИЛИ и разработать новые алгоритмы комбинаторной генерации.

Дополнительно был рассмотрен процесс разработки новых алгоритмов ранжирования и генерации по рангу с помощью метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ для следующих комбинаторных множеств: множество последовательностей вариантов ответа на тест с вопросами закрытого типа; множество исходов турнира на выбывание; множество частей круга, полученных при его разрезе прямыми линиями; множество правильных скобочных последовательностей, разряженных нулями; множество разбиений множества. Полученные результаты подтверждают универсальность и эффективность применения данного метода построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ для широкого многообразия комбинаторных множеств.

Результаты данной главы опубликованы в следующих публикациях [221; 235–240].

Глава 4. База знаний производящих функций двух переменных

Приведенный в первых главах математический инструментарий позволяет решать задачи формирования информационных объектов с помощью соответствующих правил и операций. При этом важнейшее значение в выполнении этих правил и операций играют коэффициенты k-й степени производящих функций многих переменных. Дополнительным шагом в развитии данного направления является создание соответствующей базы знаний для производящих функций двух переменных и их коэффициентов k-й степени, описываемых алгеброй биномиальных коэффициентов.

В данной главе представлены основные результаты в области разработки базы знаний производящих функций двух переменных. Рассмотрены вопросы реализации данной базы знаний в виде автоматизированной электронной энциклопедии числовых пирамид. Приведена методика использования разработанной базы знаний для задач, связанных с формированием информационных объектов на основе применения производящих функций.

4.1 Структура элементов базы знаний

Базы математических знаний являются развитием классических математических справочников и энциклопедий, что, в свою очередь, делает их важным инструментом при проведении различных исследований в математических науках и смежных областях. Более того развитие систем компьютерной алгебры требует создания баз знаний, позволяющих организовать поиск и манипулирование сложными математическими объектами. Производящие функции являются одним из таких объектов, которые находят применение в различных прикладных математических дисциплинах: перечислительная комбинаторика, статистика, теория чисел, комбинаторная генерация, теория ортогональных полиномов, анализ алгоритмов и т.д.

Существующие базы знаний направлены на получение новых знаний для конкретных чисел или числовых последовательностей, например, можно выделить проект «World!Of Numbers» [241], сервер комбинаторных объектов [242] и онлайн-энциклопедию целочисленных последовательностей OEIS [94; 243].

Рассмотрим структуру фрейма базы знаний OEIS. В каждом фрейме этой базы знаний содержится:

– название числовой последовательности;

– числовая последовательность;

– комментарии, в которых описаны математические объекты, связанные с числовой последовательностью;

– формулы, которые задают числовую последовательность (например, производящая функция числовой последовательности, явная или рекуррентная формула, уравнение и т.д.);

– список научных публикаций и интернет ресурсов, содержание которых связано с числовой последовательностью;

– исходный код программной реализации вычислений значений числовой последовательности в различных системах компьютерной математики (например, Maxima, Mathematica, Maple и другие);

– примеры использования;

– связь с другими числовыми последовательностями;

– сведения об авторах, которые внесли в базу знаний информацию о числовой последовательности или дополнили ее новыми знаниями.

Основной функцией базы знаний OEIS является поиск и редактирование числовых последовательностей, которые могут быть двух типов: линейная последовательность и числовой треугольник. Данная база знаний имеет ряд существенных недостатков:

– невозможно представить тензоры и матричные представления, имеющие 3 и более индексов;

– устаревший интерфейс, формулы записываются и представляются только в текстовом режиме;

– поиск реализован только по элементам числовой последовательности;

– отсутствуют механизмы экспорта знаний в другие форматы.

Описанные недостатки не снижают значение этой базы знаний для математических исследований и практики использования числовых последовательностей. Тем не менее на основе вышеизложенного предлагается создать базу знаний для

представления производящих функций и их коэффициентов k-й степени в двух вариантах: база знаний на основе системы компьютерной алгебры «Maxima» для расширения модулей по работе с производящими функциями и онлайн-сервис в виде автоматизированной поисковой системы по производящим функциям с расширенными возможностями поиска и генерации программного обеспечения по вычислению коэффициентов производящих функций. Переход на исследование коэффициентов степеней производящих функций многих переменных открыл новые возможности для решения задач, основанных на применении композиции производящих функций многих переменных, а также для решения смежных задач. Запишем основные соотношения для коэффициентов степеней производящих функций двух переменных и дадим соответствующие определения.

Пусть задана производящая функция вида

$$U(x,y) = \sum_{n\geqslant 0}\sum_{m\geqslant 0} u(n,m)\, x^n y^m.$$

Тогда числовой пирамидой для производящей функции $U(x,y)$ будем называть трехмерную таблицу, формируемую выражением

$$T(n,m,k) = [x^n y^m]U(x,y)^k,$$

где $T(n,m,k)$ описывается выражением, состоящим из произведения или деления биномиальных коэффициентов, а также рациональных выражений, состоящих из переменных n, m, k и констант.

Например, для производящей функции

$$U(x,y) = \frac{1}{1-x-y}$$

описываемая ею числовая пирамида будет задана формулой

$$T(n,m,k) = [x^n y^m]U(x,y)^k = \binom{n+m}{n}\binom{n+m+l-1}{n+m}.$$

Числовая пирамида состоит из коэффициентов k-й степени производящей функции $U(x,y)$. Тогда для числовой пирамиды можно записать рекуррентное выражение вида

$$T(n,m,k) = \sum_{i=0}^{n}\sum_{j=0}^{m} T(i,j,k-1)\, u(n-i,m-j).$$

Структура фрейма предлагаемой базы знаний содержит:

– десятичный четырехзначный номер производящей функции;

– явное выражение производящей функции $U_{num}(x,y)$;

– явное выражение коэффициентов $T_{num}(n,m,k)$;

– числовую пирамиду, представленную трехмерной таблицей;

– связь с другими числовыми пирамидами (взаимная, обратная и другие);

– список ссылок на источники информации из онлайн-энциклопедии целочисленных последовательностей OEIS;

– программные функции вычисления коэффициентов $T_{num}(n,m,k)$;

– программные функции получения представления в формате Latex.

Таблица 4.1 — Фреймовая модель представления знаний

Имя фрейма: <PyramidN>		
Имя слота	Тип данных	Значение слота
Слот 1: <Производящая функция>	Выражение	$U_{num}(x,y)$
Слот 2: <Явное представление>	Выражение	$T_{num}(n,m,k)$
Слот 3: <Пирамида>	Целое	Трехмерная матрица
Слот 4: <Внутренние связи>	Список	Перечень взаимных, инверсных и обратных числовых пирамид
Слот 5: <Связь с OEIS>	Список	Перечень ссылок URL
Слот 6: <Свойства>	Текст	Свойства симметричности
Слот 7: < Процедура 1>	Процедура	Построение числовой пирамиды
Слот 8: < Процедура 2>	Процедура	Получение взаимных и обратных числовых пирамид
Слот 9: < Процедура 3>	Процедура	Вычисление коэффициентов производящей функции $U_{num}(x,y)$
Слот 10: < Процедура 4>	Процедура	Процедура получения представления в формате TeX
Слот 11: < Процедура 5>	Процедура	Процедура получения представления в формате Maxima
Слот 12: < Процедура 6>	Процедура	Процедура построения числовой пирамиды
Слот 13: < Процедура 7>	Процедура	Поиск числовой пирамиды по значениям ее слота
Слот 14: < Процедура 8>	Связанная процедура	Проверка на дублирование

4.2 Методика получения числовых пирамид

Сначала определим основные соотношения и правила для получения соответствующих числовых пирамид.

Пусть для производящей функции $U(x,y)$ с известной для нее числовой пирамидой $T(n,m,k)$ задана взаимная производящая функция

$$U_r(x,y) = \frac{1}{U_{num}(x,y)}.$$

Тогда для производящей функции $U_r(x,y)$ соответствующая ей числовая пирамида будет иметь выражение

$$T_r(n,m,k) = \sum_{i=0}^{n+m} \binom{n+m+k}{i+k} \binom{i+k-1}{i} \frac{T(n,m,i)}{T(0,0,1)^{i+k}} (-1)^i. \qquad (4.1)$$

На основании формулы обращения Лагранжа запишем функциональное уравнение

$$U_A(x,y) = U(xU_A(x,y),y).$$

В результате получаем для производящей функции $U_A(x,y)$ соответствующую ей числовую пирамиду

$$T_A(n,m,k) = \frac{k}{n+k} T(n,m,n+k). \qquad (4.2)$$

Представленные формулы для взаимной (4.1) и реверсивной (4.2) числовых пирамид позволяют получить следующую схему отношений между числовыми пирамидами:

$$\begin{array}{ccccccc} \ldots & \to & T(n,m,k) & \to & T_A(n,m,k) & \to & \ldots \\ & & \updownarrow & & \updownarrow & & \\ \ldots & \leftarrow & T_r(n,m,k) & \leftarrow & T_{Ar}(n,m,k) & \leftarrow & \ldots \end{array}$$

Вертикальная стрелка означает отношение взаимности, стрелка вправо — отношение реверсивности, стрелка влево — отношение обратной реверсивности.

Числовая пирамида $T(n,m,k)$ будет называться левой по отношению к числовой пирамиде $T_A(n,m,k)$. Аналогично, числовая пирамида $T_A(n,m,k)$ будет называться правой по отношению к числовой пирамиде $T(n,m,k)$.

Кроме того, существуют инверсные отношения между диагонально расположенными числовыми пирамидами. Например, на основе применения формул (4.2) и (4.1) получим

$$T_A(n,m,k) = \frac{k}{n+k}\sum_{i=0}^{n+m}\binom{2n+m+k}{i+n+k}\binom{i+n+k-1}{i}\frac{T_r(n,m,i)}{T_r(0,0,1)^{i+n+k}}(-1)^i, \tag{4.3}$$

что также соответствует уравнению вида

$$A(x,y) = \frac{1}{U_r(x\,A(x,y),y)}.$$

Пусть заданы производящая функция $U(x,y)$ и соответствующая ей числовая пирамида $T(n,m,k)$, тогда для левой пирамиды будет верно соотношение

$$T_A(n,m,k) = k\sum_{i=0}^{n+m}\binom{2n+m-k}{i+n-k}\binom{i+n-k-1}{i-1}\frac{T(n,m,i)}{T(0,0,1)^{i+n-k}}\frac{(-1)^{i-1}}{i}, \tag{4.4}$$

что также соответствует уравнению вида

$$A_r(x,y) = U_r\left(\frac{x}{A_r(x,y)},y\right).$$

Построение соответствующих формул для схемы отношений между числовыми пирамидами относительно формальной переменной y будет выполняться аналогичным образом.

Далее рассмотрим методику получения числовых пирамид. Пусть имеется множество $G = \{g_i(x)\}_{i=1}^N$ производящих функций одной переменной, а коэффициенты k-й степени данных производящих функций описываются биномиальными коэффициентами и формируют числовые треугольники, записанные в виде множества $\{T_i(n,k)\}_{i=1}^N$. Пример такого множества приведен в таблице 4.2 (отметим, что данная таблица существенно ограничена и используется лишь в качестве наглядного примера).

Можно предложить несколько способов получения числовых пирамид производящей функции двух переменных, заключающиеся в выборе двух производящих функций $g_{num1}(x), g_{num2}(x) \in G$ и построении новой производящей функции двух переменных путем использования следующих методов:

1. На основе операции умножения производящих функций;
2. На основе операции композиции производящих функций;
3. На основе решения уравнения Лагранжа для производящих функций.

Таблица 4.2 — Пример множества производящих функций и их коэффициентов

Производящая функция $g_i(x)$	Коэффициенты k-й степени $T_i(n,k) = [x^n]g_i(x)^k$
$g_1(x) = (1+x)$	$\binom{k}{n}$
$g_2(x) = (1+x)^2$	$\binom{2k}{n}$
$g_3(x) = \frac{1}{1-x}$	$\binom{n+k-1}{n}$
$g_4(x) = \frac{1}{(1-x)^2}$	$\binom{n+2k-1}{n}$
$g_5(x) = \frac{1-\sqrt{1-4x}}{2x}$	$\frac{k}{n+k}\binom{2n+k-1}{n}$
$g_6(x) = \frac{1-2x-\sqrt{1-4x}}{2x^2}$	$\frac{k}{n+k}\binom{2n+2k}{n}$
$g_7(x) = \frac{1+\sqrt{1+4\,x}}{2}$	$\begin{cases} 1, & n=0, k=0, \\ \frac{k(-1)^{n-1}}{n}\binom{2n-k-1}{n-1}, & n>0. \end{cases}$
$g_8(x) = \sqrt{\frac{1-\sqrt{1-16\,x}}{8\,x}}$	$\begin{cases} 1, & n=0, k=0, \\ \frac{k\,4^n\binom{\frac{4\,n+k}{2}-1}{n}}{2\,n+k}, & (n+k)-\text{четное}, \\ \frac{k\binom{\frac{4\,n+k-1}{2}}{n}\binom{4\,n+k-1}{\frac{4\,n+k-1}{2}}}{(2n+k)\binom{2n+k-1}{\frac{n+k-1}{2}}}, & (n+k)-\text{нечетное}. \end{cases}$
$g_9(x) = \sqrt{1+x}$	$\begin{cases} \binom{m}{n}4^n, & 2\,m=k, \\ \frac{(-1)^{n-m}\binom{n}{m}\binom{2n}{n}}{\binom{2n}{2m}}, & 2m+1=k, n>k, \\ \frac{\binom{2m}{2n}\binom{2n}{n}}{\binom{m}{n}}, & 2m+1=k, n<k. \end{cases}$
$g_{10}(x) = \sqrt{4\,x^2+1}+2\,x$	$\begin{cases} \frac{k\binom{n+j}{n}\binom{2\,n+2\,j}{n+j}}{\binom{2\,j}{j}(n+k)}, & n+k-\text{четное}, j=\frac{n+k}{2}, \\ \frac{k\binom{n+j}{n}\binom{2\,n+2\,j}{n+j}}{\binom{2\,j}{j}(n+k)}, & n+k-\text{нечетное}, j=\frac{n+k+1}{2}. \end{cases}$
$g_{11}(x) = \frac{1}{\sqrt{1-4\,x}}$	$\begin{cases} \frac{\binom{n+j}{n}\binom{2\,n+2\,j}{n+j}}{\binom{2\,j}{j}}, & k-\text{четное}, j=\frac{k}{2}, \\ \frac{\binom{n+j}{n}\binom{2\,n+2\,j}{n+j}}{\binom{2\,j}{j}}, & k-\text{нечетное}, j=\frac{k-1}{2}. \end{cases}$
$g_{12}(x) = \frac{1-\sqrt{1+4\,x^2}}{2}$	$\begin{cases} 0, & n=0, n<2k, \\ \frac{k}{n}(-1)^{\frac{n}{2}-1}\left(1+(-1)^n\right)\binom{n-k-1}{\frac{n}{2}-1}, & n>0. \end{cases}$

Рассмотрим первый метод получения числовых пирамид. Для этого выберем две производящие функции $g_{num1}(x), g_{num2}(x) \in G$ и построим новую производящую функцию двух переменных вида

$$U(x,y) = g_{num1}(x) \cdot g_{num2}(y).$$

Тогда числовая пирамида, соответствующая производящей функции $U(x,y)$, определяется как

$$T(n,m,k) = T_{num1}(n,k) \cdot T_{num2}(m,k).$$

Например, для

$$U(x,y) = g_1(x) \cdot g_3(y) = \frac{1+x}{1-y},$$

получаем

$$T(n,m,k) = T_1(n,k) \cdot T_3(m,k) = \binom{k}{n}\binom{m+k-1}{m}.$$

Используя введенную схему отношений между числовыми пирамидами можно получить большое число разнообразных числовых пирамид. Например, на основе соотношения (4.2) найдем правую числовую пирамиду по переменной x

$$T_A(n,m,k) = \frac{k}{n+k}T(n,m,n+k) = \frac{k}{n+k}\binom{n+k}{n}\binom{m+n+k-1}{m}.$$

При этом выражение производящей функции $U_A(x,y)$, соответствующей данной числовой пирамиде, будет получено из решения функционального уравнения

$$U_A(x,y) = U(x\,U_A(x,y),y) = \frac{1+x\,U_A(x,y)}{1-y}.$$

В результате получаем

$$U_A(x,y) = \frac{1}{1-x-y}.$$

Двигаясь далее вправо по схеме отношений между числовыми пирамидами можно получить другие числовые пирамиды и соответствующие им производящие функции.

Рассмотрим второй метод получения числовых пирамид. Для этого выберем две производящие функции $g_{num1}(x), g_{num2}(x) \in G$ и построим новую производящую функцию двух переменных вида

$$U(x,y) = g_{num1}(x \cdot g_{num2}(y)).$$

Тогда числовая пирамида, соответствующая производящей функции $U(x,y)$, определяется как

$$T(n,m,k) = T_{num1}(n,k) \cdot T_{num2}(m,n).$$

Например, для

$$U(x,y) = g_1(x \cdot g_3(y)) = \frac{1+x-y}{1-y},$$

получаем

$$T(n,m,k) = T_1(n,k) \cdot T_3(m,n) = \binom{k}{n}\binom{m+n-1}{m}.$$

Рассмотрим третий метод получения числовых пирамид. Для этого выберем две производящие функции $g_{num1}(x), g_{num2}(x) \in G$ и построим новую производящую функцию двух переменных вида

$$U(x,y) = g_{num1}(x \cdot g_{num2}(y)) \cdot g_{num2}(y).$$

Тогда числовая пирамида, соответствующая производящей функции $U(x,y)$, определяется как

$$T(n,m,k) = T_{num1}(n,k) \cdot T_{num2}(m,n+k).$$

Например, для

$$U(x,y) = g_1(x \cdot g_3(y)) \cdot g_3(y) = \frac{1+x-y}{(1-y)^2},$$

получаем

$$T(n,m,k) = T_1(n,k) \cdot T_3(m,n) = \binom{k}{n}\binom{m+n+k-1}{m}.$$

В общем случае можно записать производящую функцию

$$U(x,y) = g_{num1}(x \cdot g_{num2}(y)^a)^b \cdot g_{num2}(y)^c,$$

где $a,b,c \in \mathbb{N}$.

Тогда числовая пирамида, соответствующая производящей функции $U(x,y)$, определяется как

$$T(n,m,k) = T_{num1}(n,bk) \cdot T_{num2}(m,an+ck).$$

Кроме того, двигаясь вправо по схеме отношений между числовыми пирамидами можно получить другие числовые пирамиды и соответствующие им производящие функции. За счет применения совокупности представленных правил можно получить неограниченное количество числовых пирамид. Однако, ограничив множество значений параметров a, b и c, получим конечное множество числовых пирамид. В настоящее время в базе знаний насчитывается 1502 производящие функции и соответствующие им числовые пирамиды.

4.3 Программная реализация базы знаний в виде электронной энциклопедии числовых пирамид

Далее представлены подробности о разработке программной системы поддержки базы знаний производящих функций двух переменных на основе системы компьютерной алгебры «Maxima». Структура данной программной системы представлена на рисунке 4.1.

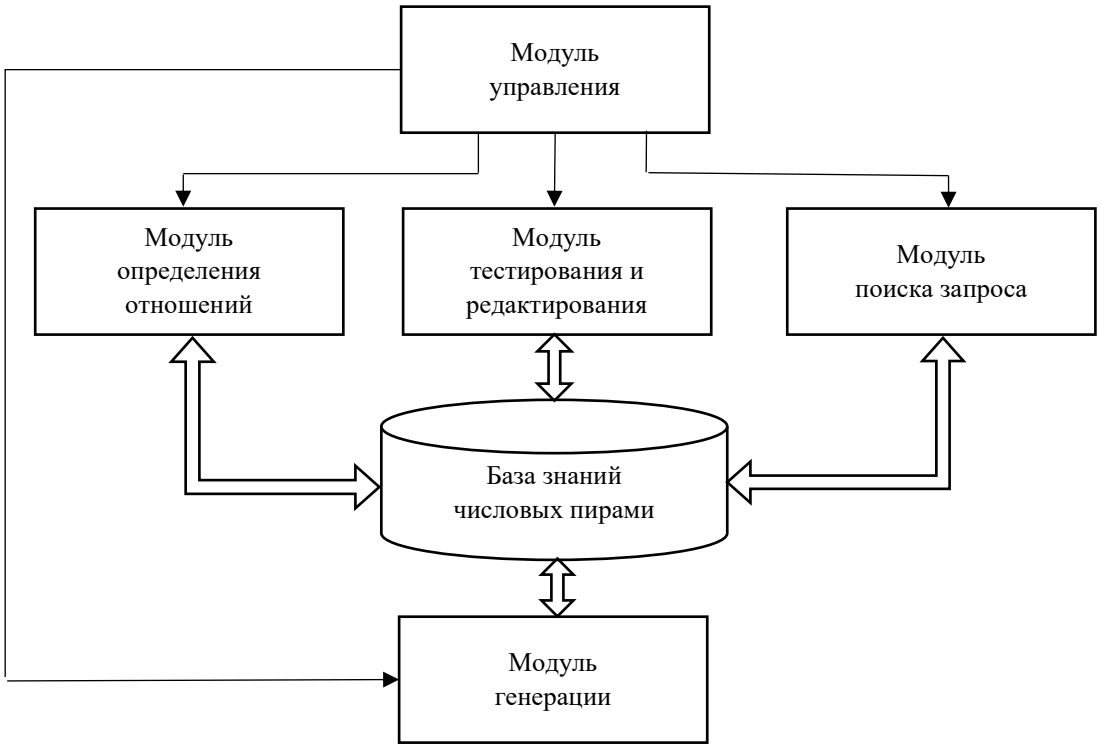


Рисунок 4.1 — Структура программной системы поддержки базы знаний

Рассмотрим описание модулей данной системы:

1. Модуль управления базой знаний числовых пирамид: обеспечивает ввод и редактирование информации о числовых пирамидах;

2. Модуль поиска запроса: производит синтаксический анализ запроса пользователя:

– при неверном запросе формируется сообщение об ошибке;

– при верном запросе производится его преобразование для дальнейшей обработки, выполняется поиск подходящей числовой пирамиды в базе знаний и формируется список таких числовых пирамид (если в результате поиска список пуст, то формируется соответствующее сообщение);

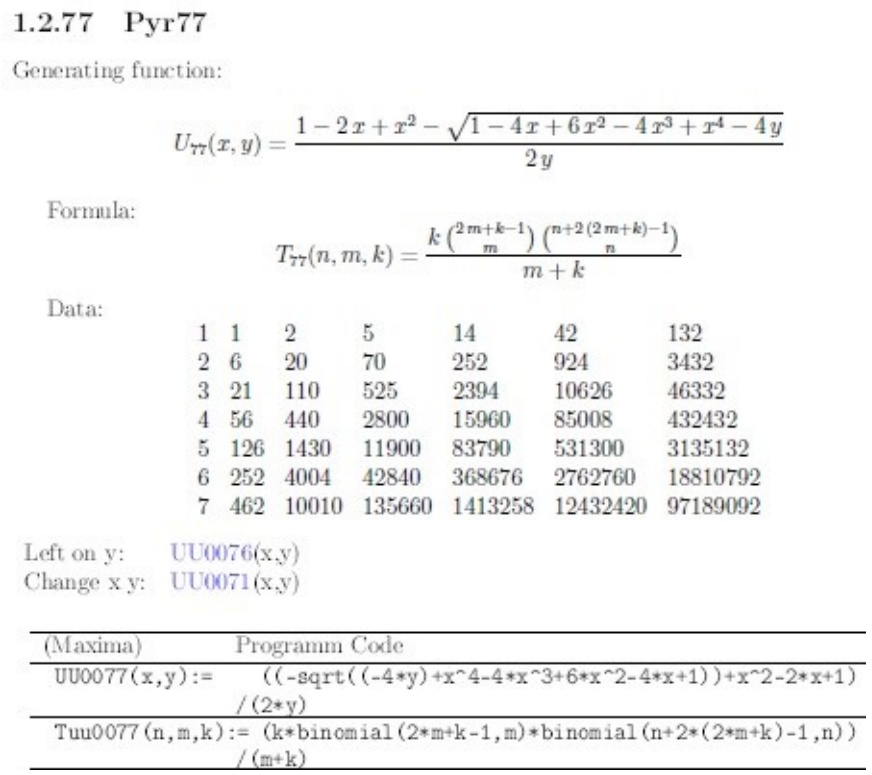
3. Модуль определения отношений: для заданной числовой пирамиды производит вычисление взаимной, инверсной, реверсивной и обратной реверсивной числовых пирамид, а также выполняет поиск их в базе знаний (также в рамках данного модуля вычисляются свойства числовой пирамиды, такие как симметричность и т.п.);

4. Модуль тестирования и редактирования: производит по запросу пользователя редактирование элементов базы знаний и их тестирование, которое проводится в двух видах:

– проверка на дублирование элементов базы знаний, так как дублирование не допустимо;

– проверка на соответствие производящей функции и ее трехмерного матричного представления (k-я степень производящей функции раскладывается в ряд Тейлора, извлекается массив значений коэффициентов разложения, производится вычисление трехмерной матрицы по явной формуле и сравниваются два полученных представления);

5. Модуль генерации: производит построение производящих функций и их матричных представлений, а также записывает их в формате Latex.



1.2.77 Pyr77

Generating function:

$$U_{77}(x,y)=\frac{1-2x+x^2-\sqrt{1-4x+6x^2-4x^3+x^4-4y}}{2y}$$

Formula:

$$T_{77}(n,m,k)=\frac{k\binom{2m+k-1}{m}\binom{n+2(2m+k)-1}{n}}{m+k}$$

Data:

1	1	2	5	14	42	132
2	6	20	70	252	924	3432
3	21	110	525	2394	10626	46332
4	56	440	2800	15960	85008	432432
5	126	1430	11900	83790	531300	3135132
6	252	4004	42840	368676	2762760	18810792
7	462	10010	135660	1413258	12432420	97189092

Left on y: UU0076(x,y)
Change x y: UU0071(x,y)

(Maxima)	Programm Code
UU0077(x,y):=	((-sqrt((-4*y)+x^4-4*x^3+6*x^2-4*x+1))+x^2-2*x+1) /(2*y)
Tuu0077(n,m,k):=	(k*binomial(2*m+k-1,m)*binomial(n+2*(2*m+k)-1,n)) /(m+k)

Рисунок 4.2 — Пример вывода страницы с информацией о числовой пирамиде под номером 77

На рисунке 4.2 приведен пример вывода страницы с информацией о числовой пирамиде под номером 77. На ней отображены:

– формула производящей функции $U_{77}(x,y)$;

– явная формула коэффициентов $T_{77}(n,m,k)$;

– таблица коэффициентов разложения производящей функции $U_{77}(x,y)$;

– ссылки на связанные числовые пирамиды (левая числовая пирамида под номером 76 и числовая пирамида под номером 71, полученная путем перемены местами x и y);

– текст программ на языке Maxima.

Созданная база знаний носит ограниченный характер и направлена на использование только внутри системы компьютерной алгебры «Maxima». Для широкого использования этой базы знаний дополнительно разработана ее онлайн-версия с расширенными возможностями поиска и генерацией различных выходных форматов. Также предлагаются механизмы развития базы знаний путем внесения изменений научным сообществом математиков и информатиков. Для реализации этой идеи был разработан транслятор, который по описанию производящей функции и ее коэффициентов формирует Latex-представление с последующим его преобразованием в функции языков Python, Mathematica и HTML.

Электронная энциклопедия числовых пирамид (ЭЭЧП) разработана в форме веб-сайта, наподобие онлайн-энциклопедии целочисленных последовательностей OEIS. Рассмотрим модель представления данных, которая состоит из следующих атрибутов:

1. Идентификатор числовой пирамиды num;

2. Формула производящей функции $U_{num}(x,y)$;

3. Явная формула коэффициентов $T_{num}(n,m,k)$;

4. Программа вычисления коэффициентов числовой пирамиды по формуле $T_{num}(n,m,k)$;

5. Программа вычисления коэффициентов числовой пирамиды через разложение в ряд производящей функции $U_{num}(x,y)$;

6. Представление производящей функции $U_{num}(x,y)$ в виде mathml-текста;

7. Представление явной формулы коэффициентов $T_{num}(n,m,k)$ в виде mathml-текста;

8. Представление производящей функции $U_{num}(x,y)$ в виде программы на языке Maxima;

9. Представление явной формулы коэффициентов $T_{num}(n,m,k)$ в виде программы на языке Maxima;

10. Представление производящей функции $U_{num}(x,y)$ в виде программы на языке Mathematica;

11. Представление явной формулы коэффициентов $T_{num}(n,m,k)$ в виде программы на языке Mathematica;

12. Список связанных пирамид (взаимная, инверсная, реверсивная, инверсная, обратная реверсивная);

13. Список ссылок на последовательности онлайн-энциклопедии целочисленных последовательностей OEIS;

14. Список свойств пирамиды;

15. Представление производящей функции $U_{num}(x,y)$ в виде Latex-текста;

16. Представление явной формулы коэффициентов $T_{num}(n,m,k)$ в виде Latex-текста.

На рисунке 4.3 представлена структура программной системы ЭЭЧП.

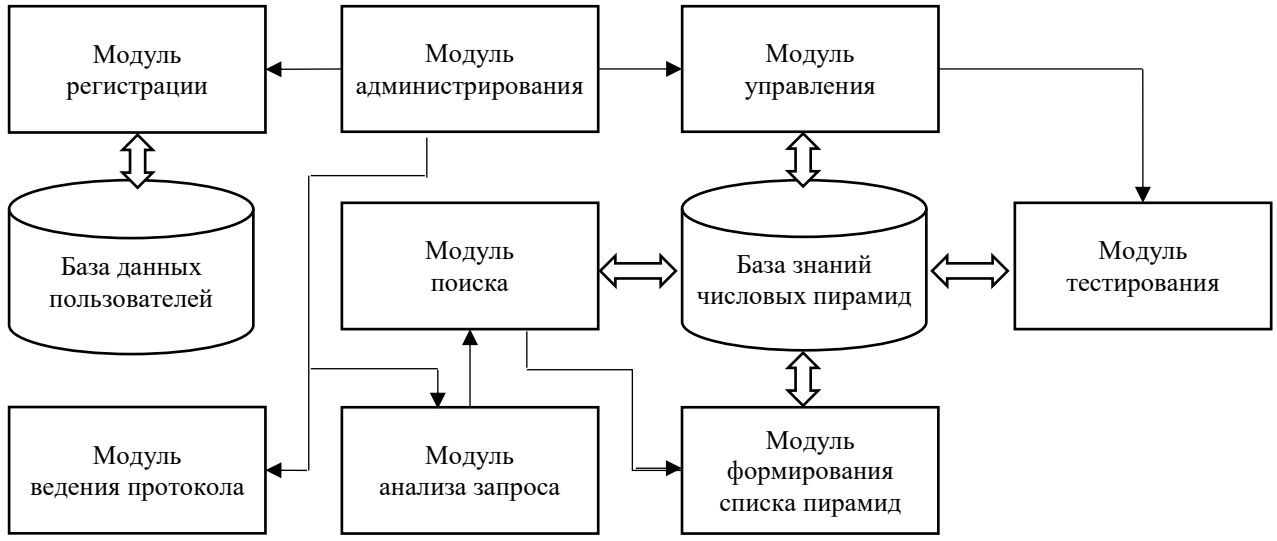


Рисунок 4.3 — Структура программной системы электронной энциклопедии числовых пирамид

Структура программной системы ЭЭЧП состоит из базы данных пользователей и базы знаний числовых пирамид, а также следующих модулей системы:

1. Модуль администрирования: обеспечивает координацию и установку модулей системы, базы данных и базы знаний, а также организует права доступа к системе;

2. Модуль регистрации: обеспечивает регистрацию и вход в систему;

3. Модуль управления базой знаний числовых пирамид: обеспечивает ввод и редактирование данных и программ для числовых пирамид;

4. Модуль анализа запроса: производит синтаксический анализ запроса пользователя, при неверном запросе формирует сообщение об ошибке, при верном запросе производит его преобразование для дальнейшей обработки и передает его в модуль поиска;

5. Модуль поиска: производит поиск подходящей числовой пирамиды в базе знаний и формирует список таких числовых пирамид (если в результате поиска список пуст, то формирует соответствующее сообщение, иначе передает полученный список в модуль формирования списка пирамид);

6. Модуль формирования списка пирамид: обеспечивает преобразование списка числовых пирамид в выходное представление и передает его пользователю;

7. Модуль ведения протокола: осуществляет запись протокола в файл протокола (структура протокола содержит информацию о пользователе, запросе и работе модулей системы);

8. Модуль тестирования: обеспечивает тестирование базы знаний при внесении в нее новых числовых пирамид (проверяется соответствие производящей функции и явной формулы числовой пирамиды, отсутствие копий этой числовой пирамиды под другими номерами, проверяются ссылки на другие числовые пирамиды).

Язык запросов описывается регулярной грамматикой и содержит следующие команды:

1. F:<выражение производящей функции числовой пирамиды>;
2. T:<выражение явной формулы коэффициентов числовой пирамиды>;
3. D:<список строк со значениями коэффициентов числовой пирамиды>;
4. <идентификатор числовой пирамиды>.

Рассмотрим процедуру поиску на следующих примерах:

Для поиска числовой пирамиды по заданной производящей функции воспользуемся запросом «F:-(sqrt(-4*x*y-4*x+1)-1)/(2*x*y+2*x)», в итоге получим следующую запись (рисунок 4.4):

Рисунок 4.4 — Пример запроса на поиск числовой пирамиды по производящей функции

Для поиска числовой пирамиды по заданной явной формуле коэффициентов воспользуемся запросом «T:binomial(n+m,n)/(n+m+1)*binomial(2*n+2*m,n+m)», в итоге получим следующую запись (рисунок 4.5):

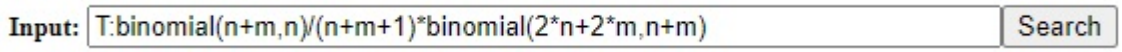


Рисунок 4.5 — Пример запроса на поиск числовой пирамиды по явной формуле

Для поиска числовой пирамиды по заданному списку строк со значениями коэффициентов воспользуемся запросом «D: 1,1,1; 1,2,3», в итоге получим следующую запись (рисунок 4.6):

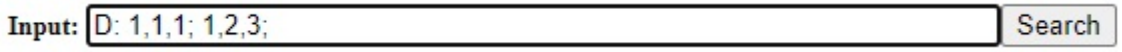


Рисунок 4.6 — Пример запроса на поиск числовой пирамиды по списку строк со значениями коэффициентов

Кроме того, допустимы запросы на поиск числовых пирамид с произвольными символами, например, запрос «D: 1,1,*; 1,2,*», где звездочки означают пропуск сравнения. Также можно выполнить поиск числовой пирамиды по идентификатору. На рисунке 4.7 приведен пример вывода фрагмента страницы электронной энциклопедии числовых пирамид для числовой пирамиды под номером 77. На ней отображено:

1. Формула производящей функции $U_{77}(x,y)$;
2. Явная формула коэффициентов $T_{77}(n,m,k)$;
3. Таблица коэффициентов разложения производящей функции $U_{77}(x,y)$;
4. Ссылки на связанные числовые пирамиды (левая числовая пирамида под номером 76 и числовая пирамида под номером 71, полученная путем перемены местами x и y);
5. Текст программ на языке Maxima;
6. Текст программ на языке Mathematica.

Программная реализация данной электронной энциклопедии числовых пирамид основана на использовании языка программирования Python, библиотеки Sympy и веб-фреймворка Flask. Веб-фреймворк Flask обеспечивает системную часть серверного приложения на основе CGI интерфейса с применением базы данных SQLAlchemy. Библиотека Sympy обеспечивает символьные вычисления по аналогии с системами компьютерной алгебры, например, были использованы следующие функциональные возможности библиотеки:

1. Получение математических выражений из строки символов;
2. Получение функций языка Python из математических выражений;
3. Разложение функции в ряд Тейлора;
4. Преобразование математических выражений в выражения языков Maxima, Mathematica, Latex;
5. Функции эквивалентных преобразований.

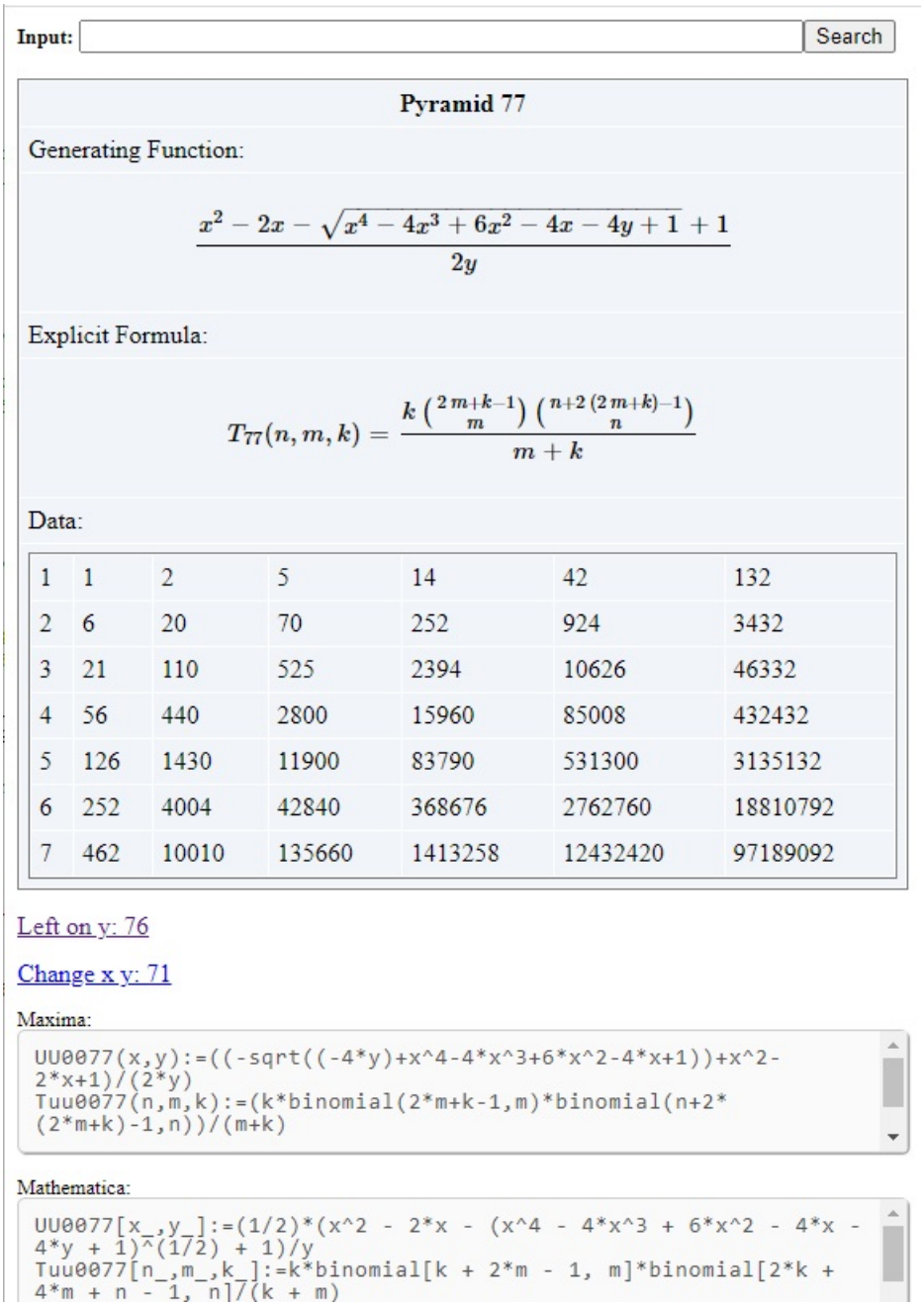


Рисунок 4.7 — Пример вывода фрагмента страницы электронной энциклопедии числовых пирамид

4.4 Методика использования базы знаний производящих функций двух переменных

В данном разделе рассмотрено использование разработанной базы знаний производящих функций двух переменных при решении следующих задач: оперирование производящими функциями двух переменных и получение явных выражений для коэффициентов композиции производящих функций двух переменных, коэффициентов взаимной и обратной производящих функций двух переменных, коэффициентов логарифмических производных производящих функций, а также их степеней. В дополнение, рассмотрена обратная задача, направленная на получение производящих функций для явных выражений, описывающих их коэффициенты.

4.4.1 Получение явных выражений коэффициентов композиции производящих функций двух переменных

Рассмотрим решение задачи, направленной на получение явных выражений коэффициентов композиции производящих функций двух переменных. Для этого воспользуемся разработанными методами получения явных выражений для разных вариантов композиций производящих функций (на основе правил из таблицы 2.1), а также методами получения явных выражений коэффициентов степеней композиций производящих функций двух переменных (на основе правил из таблицы 2.2).

Для наглядности приведем часть таблицы, в которой в одном столбце записаны примеры вариантов композиций производящих функций одной и двух переменных, а в другом столбце — явные выражения для коэффициентов k-й степени данных композиций. При подстановке $k = 1$ получается явное выражение для коэффициентов самой композиции производящих функций.

Для вычисления композиции $G(x,y) = H(A(x,y),y)$ необходимо, чтобы выполнялось следующее условие для внутренней функции: $A(0,0) = 0$. Поскольку в разработанной базе знаний содержится информация о большом количестве производящих функций двух переменных, в том числе с $A(0,0) \neq 0$, то для выполнения требуемого условия можно воспользоваться следующими способами:

1. Домножить производящую функцию на моном $x^a y^b$, где $a,b \in \mathbb{N}$, $a,b \geqslant 0$;
2. Вычесть из производящей функции свободный член $A(0,0)$.

Таблица 4.3 — Формулы для определения коэффициентов степеней композиции производящих функций

№	Композиция	Явная формула
1	$G(x,y)^k = H(A(x),y)^k$	$g(n,m,k) = \sum\limits_{q=0}^{n} A^{\Delta}(n,q)h(q,m,k)$
2	$G(x,y)^k = H(x,B(y))^k$	$g(n,m,k) = \sum\limits_{r=0}^{m} B^{\Delta}(m,r)h(n,r,k)$
3	$G(x,y)^k = H(A(x,y))^k$	$g(n,m,k) = \sum\limits_{q=0}^{n+m} A^{\Delta}(n,m,q)h(q,k)$
4	$G(x,y)^k = H(A(x,y),y)^k$	$g(n,m,k) = \sum\limits_{q=0}^{n+m}\sum\limits_{r=0}^{m} A^{\Delta}\left(n,m-r,q\right)h(q,r,k)$
5	$G(x,y)^k = H(x,B(x,y))^k$	$g(n,m,k) = \sum\limits_{q=0}^{n}\sum\limits_{r=0}^{n+m-q} B^{\Delta}\left(n-q,m,r\right)h(q,r,k)$
6	$G(x,y)^k = H(A(x),B(y))^k$	$g(n,m,k) = \sum\limits_{q=0}^{n}\sum\limits_{r=0}^{m} A^{\Delta}(n,q)\,B^{\Delta}(m,r)\,h(q,r,k)$
7	$G(x)^k = H(x,B(x))^k$	$g(n,k) = \sum\limits_{q=0}^{n}\sum\limits_{r=0}^{n-q} B^{\Delta}(n-q,r)\,h(q,r,k)$

Например, рассмотрим производящую функцию двух переменных

$$G(x,y) = \frac{x\,y-1}{x\,y+x^2+x-1},$$

у которой свободный член не равен 0, и представим ее в виде композиции производящих функций

$$G(x,y) = \frac{1}{1-\frac{x+x^2}{1-x\,y}} = H(A(x,y)),$$

где

$$H(x) = \frac{1}{1-x},$$

$$A(x,y) = \frac{x+x^2}{1-x\,y}.$$

В базе знаний формируем запрос на поиск по производящей функции

$$U(x,y) = \frac{1+x}{1-y}.$$

Получаем пирамиду под номером 17 (рисунок 4.8).

Generating function:	$U_{17}(x,y) = \frac{1+x}{1-y}$
Formula:	$T_{17}(n,m,k) = \binom{k}{n}\binom{m+k-1}{m}$
Data:	1 1 1 1 1 1 1
	1 1 1 1 1 1 1
	0 0 0 0 0 0 0
	0 0 0 0 0 0 0
	0 0 0 0 0 0 0
	0 0 0 0 0 0 0
	0 0 0 0 0 0 0

Рисунок 4.8 — Информация из базы знаний о пирамиде под номером 17

Тогда для коэффициентов k-й степени производящей функции

$$xU(x,y) = x\frac{1+x}{1-y}$$

будет верна формула

$$T_{17}(n-k,m,k) = \binom{k}{n-k}\binom{m+k-1}{m}.$$

Далее выполним умножение переменной y на x, то есть рассмотрим случай

$$A(x,y) = xU(x,xy),$$

получим

$$A^{\Delta}(n,m,k) = T_{17}(n-m-k,m,k) = \binom{k}{n-k}\binom{m+k-1}{m}.$$

Откуда на основании правила композиции производящих функций двух переменных под номером 3 из таблицы 4.3 при $k = 1$ найдем

$$g(n,m) = \sum_{k=0}^{n+m} A^{\Delta}(n,m,k) = \sum_{k=0}^{n-m}\binom{k}{n-m-k}\binom{m+k-1}{m}.$$

Таким образом, получено явное выражение для числового треугольника последовательности $A055830$ [94], описывающей класс путей на решетке [244].

4.4.2 Получение явных выражений коэффициентов взаимной производящей функции двух переменных

Рассмотрим решение задачи, направленной на получение явных выражений коэффициентов взаимной производящей функции двух переменных и их степеней.

Пусть задана производящая функция вида

$$G(x,y) = \frac{x - \sqrt{x^2 - 2\,x^2\,y + 2\,\sqrt{1 - 4\,x}\,x^2\,y}}{2\,x^2\,y}.$$

Найдем коэффициенты k-й степени взаимной производящей $A(x,y)$, удовлетворяющей уравнению

$$A(x,y) = \frac{1}{G(x,y)}.$$

Для получения явного выражения для коэффициентов числовой пирамиды, соответствующей взаимной производящей функции $A(x,y)$, можно воспользоваться формулой (4.1).

Далее, с использованием разработанной базы знаний, разложим данную производящую функцию $G(x,y)$ в ряд Тейлора в точке $x = 0$ и $y = 0$. В результате получим следующую матрицу значений коэффициентов данного разложения:

$$\begin{aligned}
&1\\
&+ (1 + y + \cdots)\ x\\
&+ \left(2 + 2\,y + 2\,y^2 + \cdots\right)\ x^2\\
&+ \left(5 + 5\,y + 6\,y^2 + 5\,y^3 + \cdots\right)\ x^3\\
&+ \left(14 + 14\,y + 18\,y^2 + 20\,y^3 + 14\,y^4 + \cdots\right)\ x^4 + \cdots
\end{aligned}$$

Заметим, что данная функция описывает числовой треугольник, где $G(0,0) = 1$. Представим данную функцию как $G(x,y) = H\left(x, \frac{y}{x}\right)$. Используя разработанную базу знаний, можно получить соответствующую данному разложению производящую функцию и матричное представление в виде числовой пирамиды под номером 69 (рисунок 4.9).

Generating function:	$U_{69}(x,y) = \frac{x-\sqrt{x^2-2\,x\,y+2\,\sqrt{1-4\,x}\,x\,y}}{2\,x\,y}$						
Formula:	$T_{69}(n,m,k) = \frac{k\binom{2\,m+k-1}{m}\binom{2\,n+m+k-1}{n}}{n+m+k}$						
Data:	1	1	2	5	14	42	132
	1	2	6	20	70	252	924
	2	5	18	70	280	1134	4620
	5	14	56	240	1050	4620	20328
	14	42	180	825	3850	18018	84084
	42	132	594	2860	14014	68796	336336
	132	429	2002	10010	50960	259896	1319472

Рисунок 4.9 — Информация из базы знаний о пирамиде под номером 69

Тогда для производящей функции $G(x,y)$ будет верно выражение

$$G(x,y)^k = \sum_n \sum_m T_G(n,m,k)\, x^n\, y^m = \sum_n \sum_m T_{69}(n-m,m,k)\, x^n\, y^m.$$

Применяя формулу (4.1) к $T_G(n,m,k)$, получаем

$$T_A(n,m,k) = \sum_{i=0}^{n+m} \binom{n+m+k}{i+k} \binom{i+k-1}{i} \frac{T_G(n,m,i)}{T_G(0,0,1)^{i+k}} (-1)^i =$$

$$= \sum_{i=0}^{n+m} \binom{n+m+k}{i+k} \binom{i+k-1}{i} \binom{2m+i-1}{m} \binom{2n-m+i-1}{n-m} \frac{k(-1)^i}{n+i}.$$

4.4.3 Получение явных выражений коэффициентов обратной производящей функции двух переменных

Рассмотрим решение задачи, направленной на получение явных выражений коэффициентов обратной производящей функции двух переменных и их степеней.

Пусть задана производящая функция вида

$$G(x,y) = \frac{x}{1-x-x^2(1+y)}.$$

Найдем коэффициенты k-й степени обратной производящей функции $A(x,y)$, удовлетворяющей уравнению

$$G(A(x,y),y) = x.$$

Введем производящую функцию вида $G(x,y) = xG_x(x,y)$, тогда

$$A(x,y) = \frac{x}{G_x(A(x,y),y)}.$$

Теперь выполним подстановку $A(x,y) = xA_x(x,y)$, получим

$$A_x(x,y) = \frac{1}{G_x(xA_x(x,y),y)}.$$

Также введем обратную производящую функцию

$$G_r(x,y) = \frac{1}{G_x(x,y)} = 1 - x - x^2(1+y) = 1 - x(1+x+xy),$$

тогда получим

$$A_x(x,y) = G_r(xA_x(x,y),y).$$

Для производящей функции $1 + x + xy$ найдем в базе знаний представление в виде числовой пирамиды под номером 37 (рисунок 4.10).

Generating function:	$U_{37}(x,y) = 1 + x + x\,y$
Formula:	$T_{37}(n,m,k) = \binom{k}{n}\binom{n}{m}$
Data:	1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Рисунок 4.10 — Информация из базы знаний о пирамиде под номером 37

Откуда на основе биномиальной теоремы получаем

$$T_{G_r}(n,m,k) = \sum_{i=0}^{n+m} T_{37}(n-i,m,i)\binom{k}{i}(-1)^i = \sum_{i=0}^{n}\binom{i}{n-i}\binom{n-i}{m}\binom{k}{i}(-1)^i.$$

Тогда на основе формулы (4.2) коэффициенты $A_x(x,y)^k$ будут равны

$$T_{A_x}(n,m,k) = \frac{k}{n+k}T_{G_r}(n,m,n+k) = \frac{k}{n+k}\sum_{i=0}^{n}\binom{i}{n-i}\binom{n-i}{m}\binom{n+k}{i}(-1)^i.$$

4.4.4 Получение производящих функций для явных выражений, описывающих их коэффициенты

Рассмотрим решение задачи, направленной на получение производящих функций для явных выражений, описывающих их коэффициенты. Данная задача является обратной предыдущим, то есть по виду функции коэффициентов производящей функции требуется определить выражение для производящей функции.

Пусть задано явное выражение вида

$$g(n,m) = \sum_{k=0}^{m} \binom{n+m-k+2}{k} \binom{n+m-k}{n}.$$

Произведем ряд преобразований с целью получить одну из композиционных формул. Сначала заменим порядок суммирования с k на $m-k$, тогда

$$g(n,m) = \sum_{k=0}^{m} \binom{n+k+2}{m-k} \binom{n+k}{n}.$$

Также с учетом преобразований индекса k получаем

$$g(n,m) = \sum_{k=n}^{n+m} \binom{k+2}{n+m-k} \binom{k}{n} = \sum_{k=0}^{n+m} \binom{k+2}{n+m-k} \binom{k}{k-n}.$$

Введем новую переменную $j = n+m-k$ и используем символ Кронекера, тогда

$$g(n,m) = \sum_{k=0}^{n+m} \sum_{j=0}^{m} \delta_{j,n+m-k} \binom{k+2}{j} \binom{n+m-j}{m-j}.$$

Сравним полученный результат с правилом композиции производящих функций двух переменных под номером 4 из таблицы 4.3 при $k=1$, тогда получим

$$g(n,m) = \sum_{k=0}^{n+m} \sum_{j=0}^{m} A^{\Delta}(n,m-j,k) h(k,j),$$

где

$$A^{\Delta}(n,m,k) = \delta_{k,n+m} \binom{n+m}{m},$$

$$h(n,m) = \binom{n+2}{m}.$$

Используя разработанную базу знаний, для $A^{\Delta}(n,m,k)$ найдем числовую пирамиду под номером 1, которой соответствует производящая функция

$$A(x,y) = x + y.$$

Также с помощью базы знаний через запрос $T = binomial(n+2,m)$ найдем числовую пирамиду под номером 42 (рисунок 4.11).

Тогда искомая производящая функция будет равна

$$G(x,y) = H(A(x,y),y) = U_{42}(x+y,y) = \frac{(1+y)^2}{1-(x+y)(1+y)}.$$

Generating function:	$U_{42}(x,y) = \frac{(1+y)^2}{1-x\,(1+y)}$
Formula:	$T_{42}(n,m,k) = \binom{n+k-1}{n}\binom{n+2\,k}{m}$
Data:	1 2 1 0 0 0 0
	1 3 3 1 0 0 0
	1 4 6 4 1 0 0
	1 5 10 10 5 1 0
	1 6 15 20 15 6 1
	1 7 21 35 35 21 7
	1 8 28 56 70 56 28

Рисунок 4.11 — Информация из базы знаний о пирамиде под номером 42

4.4.5 Получение явных выражений коэффициентов логарифмических производных производящих функций

Рассмотрим решение задачи, направленной на получение явных выражений коэффициентов логарифмических производных производящих функций.

Пусть дано функциональное уравнение вида

$$A(x,y) = G(x\,A(x,y),y),$$

а также известно следующее явное выражение коэффициентов k-й степени производящей функции $G(x,y)$:

$$T_G(n,m,k) = [x^n\,y^m]G(x,y)^k.$$

Тогда коэффициенты логарифмической частной производной производящей функции $A(x,y)$ по формальной переменной x будут выражаться через коэффициенты $T_G(n,m,k)$ как

$$T_{lx}(n,m) = [x^n y^m]\frac{\partial \log(A(x,y))}{\partial x} = [x^n y^m]\frac{1}{A(x,y)}\frac{\partial A(x,y)}{\partial x} = T_G(n,m,n).$$

С другой стороны, пусть дано функциональное уравнение вида

$$A(x,y) = G(x,y\,A(x,y)),$$

а также известно явное выражение коэффициентов k-й степени производящей функции $G(x,y)$:

$$T_G(n,m,k) = [x^n\, y^m]G(x,y)^k.$$

Тогда коэффициенты логарифмической частной производной производящей функции $A(x,y)$ по формальной переменной y будут выражаться через коэффициенты $T_G(n,m,k)$ как

$$T_{ly}(n,m) = [x^n y^m]\frac{1}{A(x,y)}\frac{\partial A(x,y)}{\partial y} = T_G(n,m,m).$$

В качестве примера найдем коэффициенты логарифмической производной производящей функции $U_{139}(x,y)$ (рисунок 4.12).

Далее необходимо перейти по связанной ссылке на фрейм в базе знаний с информацией о пирамиде под номером 59 и получить следующее явное выражение коэффициентов k-й степени производящей функции $U_{59}(x,y)$:

$$T_{59}(n,m,k) = \frac{k\binom{2k}{n}\binom{k+m}{m}}{k+m}.$$

Тогда для искомой логарифмической производной получим

$$G(x,y) = \frac{\partial \log(U_{59}(x,y))}{\partial x} = \frac{1}{U_{59}(x,y)}\frac{\partial U_{59}(x,y)}{\partial x},$$

и ее коэффициенты будут иметь следующее явное выражение:

$$g(n,m) = T_{59}(n,m,n) = \frac{n\binom{2\,n}{n}\binom{m+n}{m}}{m+n} = \binom{2n}{n}\binom{n+m-1}{m}.$$

Generating function:	$U_{139}(x,y) = \frac{1-2x-y-\sqrt{1-4x-(2-4x)y+y^2}}{2x^2}$						
Formula:	$T_{139}(n,m,k) = \frac{k\binom{n+m+k}{m}\binom{2n+2k}{n}}{n+m+k}$						
Data:	1	1	1	1	1	1	1
	2	4	6	8	10	12	14
	5	15	30	50	75	105	140
	14	56	140	280	490	784	1176
	42	210	630	1470	2940	5292	8820
	132	792	2772	7392	16632	33264	60984
	429	3003	12012	36036	90090	198198	396396
Right on y:	UU0138(x,y)						
Left on x:	UU0059(x,y)						
Left on y:	UU0060(x,y)						
Change x y:	UU0359(x,y)						

Рисунок 4.12 — Информация из базы знаний о пирамиде под номером 139

Далее рассмотрим случай, когда левая числовая пирамида для заданной производящей функции не известна, но известна числовая пирамида для исходной производящей функции. Тогда можно воспользоваться следующими формулами, основанными на применении формулы (4.4):

$$T_{lx}(n,m) = \begin{cases} T(0,0,1)^n & n = 0\ , m = 0, \\ n \sum\limits_{j=1}^{n+m} \binom{n+m}{j} \frac{T(n,m,j)}{T(0,0,1)^j} \frac{(-1)^{j-1}}{j}, & n > 0 \text{ или } m > 0, \end{cases}$$

$$T_{ly}(n,m) = \begin{cases} T(0,0,1)^n & n = 0\ , m = 0, \\ m \sum\limits_{j=1}^{n+m} \binom{n+m}{j} \frac{T(n,m,j)}{T(0,0,1)^j} \frac{(-1)^{j-1}}{j}, & n > 0 \text{ или } m > 0, \end{cases}$$

Сравнивая приведенные выше формулы для частных логарифмических производных, можно увидеть, что они отличаются лишь множителями m и n. Если члены $T_{lx}(n,m)$ поделить на n, а члены $T_{ly}(n,m)$ поделить на m, то получим члены разложения композиции производящих функций вида

$$\log(U(x,y)),$$

где $U(x,y)$ — производящая функция, описывающая числовую пирамиду $T(n,m,k)$.

Тогда можем записать явную формулу коэффициентов композиции производящих функций как

$$[x^n y^m] \log(U(x,y)) = \sum_{j=1}^{n+m} \binom{n+m}{j} \frac{T(n,m,j)}{T(0,0,1)^j} \frac{(-1)^{j-1}}{j}.$$

Рассмотрим пример. Пусть дана производящая функция под номером 130 (рисунок 4.13).

Generating function:	$U_{130}(x,y) = \frac{1}{-4\,y+(1-x+y)^2}$						
Formula:	$T_{130}(n,m,k) = \frac{\binom{m+k}{k-1}\binom{n+k-1}{k-1}\binom{n+m+2\,k-1}{k-1}\binom{2\,(n+m)+2\,k}{2\,m+1}}{2\binom{2\,k-2}{k-1}\binom{2\,m+2\,k}{2\,k-2}}$						
Data:	1	2	3	4	5	6	7
	2	10	28	60	110	182	280
	3	28	126	396	1001	2184	4284
	4	60	396	1716	5720	15912	38760
	5	110	1001	5720	24310	83980	248710
	6	182	2184	15912	83980	352716	1248072
	7	280	4284	38760	248710	1248072	5200300

Рисунок 4.13 — Информация из базы знаний о пирамиде под номером 130

Тогда коэффициенты разложения композиции функций

$$\log(U_{130}(x,y)) = \log\left(\frac{1}{-4\,y+(1-x+y)^2}\right)$$

будут равны выражению

$$\sum_{k=1}^{n+m} (-1)^{k-1} \; \frac{\binom{m+k}{k-1}\binom{n+k-1}{k-1}\binom{n+m+2\,k-1}{k-1}\binom{2\,(n+m)+2\,k}{2\,m+1}}{2\,k\binom{2\,k-2}{k-1}\binom{2\,m+2\,k}{2\,k-2}} \; \binom{n+m}{k}.$$

4.5 Выводы по главе

В данной главе представлены основные результаты, связанные с созданием базы знаний производящих функций двух переменных, основанной на фреймовой модели. Данная база знаний реализована в виде электронной энциклопедии числовых пирамид, преимуществом которой является автоматизированный процесс поиска хранящихся записей. Полученная база знаний, содержащая записи о 1502 производящих функций и их коэффициентов, является мощным инструментом для тестирования модулей программных систем компьютерной алгебры, выполняющих преобразования производящих функций и их коэффициентов.

Предложена методика использования разработанной базы знаний производящих функций двух переменных, которая позволяет решать широкий круг задач оперирования производящими функциями двух переменных и их коэффициентами для формирования информационных объектов. Например, использование базы знаний может обеспечить построение алгоритмов комбинаторной генерации для более сложных комбинаторных объектов, определяемых производящими функциями многих переменных, без проведения дополнительных вычислений.

Результаты данной главы опубликованы в следующих публикациях [245;246].

Глава 5. Программное обеспечение для анализа и генерации критериев простоты числа

В данной главе представлены основные результаты в области разработки методов генерации критериев простоты числа. Излагаются теоретические основы построения таких критериев за счет использования методов оперирования коэффициентами степеней производящих функций. Показано получение критериев простоты числа на основе композиции логарифмической и обыкновенной производящих функций, композиции экспоненциальной и обыкновенной производящих функций, а также получение рекуррентных критериев простоты. Разработано соответствующее программное обеспечение, позволяющее создавать большой набор новых критериев простоты числа, в зависимости от выбранных производящих функций, и проводить анализ сгенерированных критериев простоты числа.

5.1 Метод построения критериев простоты числа на основе аппарата степеней производящих функций

Многие современные криптографические системы строятся на базе использования простых чисел. Поэтому алгоритмы генерации простых чисел и методы проверки сформированного числа на простоту являются важными инструментами при создании таких криптографических систем. Так, например, в известной криптографической системе с открытым ключом RSA потребность в выборе простых чисел имеет основополагающую позицию и от выбора простых чисел во многом определяется стойкость шифрования [247–249].

Несмотря на большое количество исследований в данной области, на сегодняшний день задача поиска и генерации простых чисел является не решенной. Поэтому исследования, способствующие развитию методов и инструментов определения простоты числа, имеют как теоретическое значение, так и практическое значение.

Процесс генерации простого числа обычно реализуется путем выполнения следующих действий:

1. Задается произвольное натуральное число, для которого заранее не известно является ли оно простым или составным;

2. Заданное число поступает на вход алгоритма проверки простоты числа (тест простоты числа), который определяет простое это число или составное.

Данные действия повторяются пока не будет получено простое число.

Существует два класса тестов простоты числа, которые выделены на основе критерия достоверности полученного результата:

1. Детерминированные тесты — достоверно дают ответ о том, что проверяемое число является простым или составным, но такие тесты обладают огромной вычислительной сложностью;

2. Вероятностные тесты — результат выполнения теста простоты числа является достоверным лишь с некоторой вероятностью, но время проверки гораздо меньше в сравнении с детерминированными тестами.

В практических задачах чаще используют вероятностные тесты простоты числа, поскольку они выигрывают по временным характеристикам по сравнению с детерминированными, особенно при работе с большими числами. Однако, в таком случае, возникает показатель вероятности ошибки построения псевдопростого числа. Под псевдопростым числом понимается составное число, которое в ходе проведения вероятностного теста простоты числа было ошибочно определено как простое число.

В литературе можно найти описания множества алгоритмов проверки числа на простоту, например, отметим работы следующих ученых: А.А. Балабанов [250], О.Н. Василенко [251], А.В. Черемушкин [252], П. Рибенбойм [253] и другие.

Под критерием простоты числа понимается утверждение, которому должны удовлетворять простые числа [254]. Если целое число n не удовлетворяет условиям критерия простоты числа, то данное число n гарантированно будет составным, иначе число n является простым с некоторой вероятностью ошибки. Зачастую используют совокупность критериев простоты числа, чтобы уменьшить вероятность получения псевдопростого числа. Во многих вероятностных тестах простоты числа в качестве основополагающего критерия простоты применяется малая теорема Ферма [255]. Поэтому исследование и поиск новых критериев простоты числа для улучшения характеристик вероятностных тестов простоты числа являются актуальными задачами.

Рассмотрим свойства композиции производящих функций, направленные на получение новых критериев простоты числа. Данное исследование базируется на применении композиции обыкновенной производящей функции с целыми коэффициентами и логарифмической производящей функции.

Согласно книге А.М. Роберта [256], логарифмическая производящая функция определяется как:

Определение 23. *Логарифмическая производящая функция есть обыкновенная производящая функция следующего вида*

$$R(x) = \sum_{n>0} r(n)x^n = \sum_{n>0} \frac{a(n)}{n} x^n, \tag{5.1}$$

где $a(n)$ — целочисленная последовательность.

Логарифмическая производящая функция представляет собой обыкновенную производящую функцию с целыми коэффициентами $a(n)$, деленными на их порядковый номер n.

На основе свойств композиции обыкновенных и логарифмических производящих функций, становится возможным построение различных критериев простоты числа для вероятностных тестов простоты числа.

Теорема 26. *Пусть задана композиция обыкновенной производящей функции $F(x) = \sum\limits_{n>0} f(n)\, x^n$, где $f(n)$ — целочисленная последовательность для $n > 0$, и логарифмической производящей функции $R(x) = \sum\limits_{n>0} r(n)\, x^n = \sum\limits_{n>0} \frac{a(n)}{n}\, x^n$, где $a(n)$ — целочисленная последовательность для $n > 0$, которая имеет вид $G(x) = R(F(x)) = \sum\limits_{n>0} g(n)\, x^n$. Тогда значение выражения*

$$ng(n) = n\sum_{k=1}^{n} F^{\Delta}(n,k)r(k) = n\sum_{k=1}^{n} \frac{F^{\Delta}(n,k)a(k)}{k} \tag{5.2}$$

будет целым для всех $n \in \mathbb{N}$.

Доказательство:

Согласно (2.2), коэффициенты производящей функции $G(x)$ равны

$$g(n) = \sum_{k=1}^{n} F^{\Delta}(n,k)r(k) = \sum_{k=1}^{n} \frac{F^{\Delta}(n,k)a(k)}{k}. \tag{5.3}$$

Производная производящей функции $G(x)$ есть функция вида

$$G'(x) = g(1) + 2g(2)x + 3g(3)x^2 + \ldots + ng(n)x^{n-1} + \ldots = \sum_{n>0} ng(n)x^{n-1}.$$

С другой стороны, получим следующее выражение производной производящей функции $G(x)$:

$$G'(x) = \left(R\left(F(x)\right)\right)' = F'(x)R'\left(F(x)\right).$$

Так как значения $f(n)$ формируют целочисленную последовательность, тогда для целых чисел n значения $nf(n)$ также будут целочисленными. Следовательно, коэффициенты производящей функции $F'(x) = \sum_{n>0} nf(n)x^{n-1}$ будут целыми.

Значения $a(n)$ формируют целочисленную последовательность. Следовательно, коэффициенты производящей функции $R'(x) = \sum_{n>0} nr(n)x^{n-1} = \sum_{n>0} a_n x^{n-1}$ также будут целыми.

Коэффициенты композиции производящих функций с целыми коэффициентами являются целыми. Производящие функции $F(x)$ и $R'(x)$ являются производящими функциями с целыми коэффициентами. Следовательно, коэффициенты композиции производящих функций $R'(F(x))$ будут целыми.

Коэффициенты произведения производящих функций с целыми коэффициентами являются целыми. Производящие функции $F'(x)$ и $R'(F(x))$ являются производящими функциями с целыми коэффициентами. Следовательно, коэффициенты производящей функции $G'(x) = F'(x)R'(F(x)) = \sum_{n>0} ng(n)x^{n-1}$ будут целыми.

Таким образом, на основе вышеизложенного получаем, что последовательность значений $ng(n)$ будет целочисленной. □

Из Теоремы 26 получим следующее новое свойство коэффициентов композиции производящих функций.

Следствие 10. *Значение функции коэффициентов* (5.3) *без n-го элемента*

$$\sum_{k=1}^{n-1} \frac{F^{\Delta}(n,k)a(k)}{k} \tag{5.4}$$

есть целое для любого простого числа n. Обратное утверждение неверно.

Доказательство:

Рассмотрим подробнее выражение (5.2). Значение композиты $F^{\Delta}(n,n)$ равно $f(1)^n$. Значения $f(n)$ формируют целочисленную последовательность и n целое, тогда последовательность значений $f(1)^n$ будет также целочисленной. По условию Теоремы 26 последовательность $a(n)$ является целочисленной. Тогда значение n-го элемента в сумме (5.2) равно целому выражению

$$n\frac{F^{\Delta}(n,n)a(n)}{n} = f(1)^n a(n).$$

Последовательности $ng(n)$ и $f(1)^n a(n)$ являются целочисленными последовательностями. Следовательно, будет целым значение выражения

$$ng(n) - f(1)^n a(n) = n\sum_{k=1}^{n-1}\frac{F^{\Delta}(n,k)a(k)}{k}.$$

Коэффициенты $f(n)$ являются целыми, тогда значение композиты $F^{\Delta}(n,k)$ также будет целым. Поскольку n простое, $n > k$, а также $F^{\Delta}(n,k)$ и $a(k)$ целые, то выражение (5.4) будет целым. Таким образом, значение выражения (5.4) является целым для любого простого числа n. □

Поскольку выражение (5.4) является целым для любого простого числа n, то его можно использовать для распознавания простых чисел от составных [257]. Для составного числа n значение выражения (5.4) может быть как целым, так и не целым. Но, если значение выражения (5.4) нецелое число, то n гарантированно составное число.

Интегрируя почленно производящую функцию $B(x) = \sum\limits_{n\geqslant 0} b(n)x^n$ по формальной переменной x, получим следующую производящую функцию:

$$\int B(x)dx = b(0)x + b(1)\frac{x^2}{2} + b(2)\frac{x^3}{3} + \ldots + b(n)\frac{x^{n+1}}{n+1} + \ldots = \sum_{n>0}\frac{b(n-1)}{n}x^n. \quad (5.5)$$

Теорема 27. *Пусть заданы обыкновенные производящие функции с целыми коэффициентами* $F(x) = \sum\limits_{n>0} f(n)\,x^n$ *и* $B(x) = \sum\limits_{n\geqslant 0} b(n)\,x^n$, *а также композиция производящих функций* $G(x) = R(F(x)) = \sum\limits_{n>0} g(n)\,x^n$, *где* $R(x) = \int B(x)dx$. *Тогда значение выражения*

$$ng(n) = n\sum_{k=1}^{n}\frac{F^{\Delta}(n,k)b(k-1)}{k} \quad (5.6)$$

будет целым для всех $n \in \mathbb{N}$.

Доказательство:

Рассмотрим последовательность $a(n)$, где $a(n) = b(n-1)$ для любого целого $n = 1, 2, \ldots$. Используя (5.5) и (5.1), получим следующую логарифмическую производящую функцию:

$$\int B(x)dx = \sum_{n>0} \frac{b(n-1)}{n} x^n = \sum_{n>0} \frac{a(n)}{n} x^n = \sum_{n>0} r(n)x^n = R(x).$$

Согласно Теореме 26, значение выражения

$$ng(n) = n \sum_{k=1}^{n} \frac{F^{\Delta}(n,k)a(k)}{k} = n \sum_{k=1}^{n} \frac{F^{\Delta}(n,k)b(k-1)}{k}$$

будет целым для всех $n \in \mathbb{N}$. □

Следствие 11. *Значение выражения*

$$\sum_{k=1}^{n-1} \frac{F^{\Delta}(n,k)b(k-1)}{k} \tag{5.7}$$

будет целым для любого простого числа n. Обратное утверждение неверно.

Полученные свойства композиции производящих функций могут применяться для построения новых критериев простоты числа. Для этого будем использовать совокупность обыкновенной и логарифмической производящих функций $F(x) = \sum\limits_{n>0} f(n)\, x^n$ и $R(x) = \sum\limits_{n>0} r(n)x^n = \sum\limits_{n>0} \frac{a(n)}{n} x^n$, где $f(n)$ и $a(n)$ являются целочисленными последовательностями. Тогда метод построения критериев простоты числа заключается в следующем:

1. Необходимо получить явную формулу коэффициентов композиции производящих функций вида $G(x) = R\left(F(x)\right) = \sum\limits_{n>0} g(n)\, x^n$;

2. Согласно Теореме 5.2 выражение $ng(n)$ является целым для всех $n \in \mathbb{N}$, поэтому необходимо произвести операцию дифференцирования;

3. По возможности упростить полученное выражение, а именно получить выражение, зависящее только от n и без дополнительного суммирования по k. Для упрощения можно воспользоваться онлайн-энциклопедией целочисленных последовательностей [94]. К сожалению, не всегда получается упростить полученное выражение, что влияет только на вычислительную сложность расчета данного выражения, поэтому такие выражения будут иметь теоретический интерес. С другой стороны, можно использовать методы приближенных вычислений, но в данной работе такой подход не рассматривался;

4. Необходимо привести полученное выражение к виду выражения (5.4), выполнив деление на n и вычитание n-го члена суммы.

В итоге после всех преобразований получается выражение, которое зависит только от n и которое будет целым для простых n, то есть получаем критерий простоты числа для вероятностных тестов простоты числа. В зависимости от параметров композиции, а именно от выбора логарифмической производящей функции и композиты производящей функции внутренней функции композиции, получаемые критерии простоты числа будут иметь разные числовые и вероятностные характеристики, а также оценки вычислительной сложности.

5.2 Критерии простоты числа на основе композиций производящих функций

5.2.1 Критерии простоты числа на основе композиции логарифмической и обыкновенной производящих функций

Используя полученный метод на основе свойств композиции логарифмической и обыкновенной производящей функции построены различные критерии простоты числа для вероятностных тестов простоты числа. Рассмотрим следующие примеры композиции производящих функций и получим соответствующие критерии простоты числа:

Пример 23. *Пусть задана композиция производящих функций* $G(x) = R(F(x))$, *где*

$$F(x) = \frac{bx}{1-ax} = \sum_{n>0} ba^{n-1}x^n \tag{5.8}$$

является обыкновенной производящей функцией с целыми коэффициентами, a *и* b *— целые числа,* $R(x) = \int B(x)dx$, *а также задана обыкновенная производящая функция с целыми коэффициентами*

$$B(x) = \frac{1}{1-x} = \sum_{n\geqslant 0} b(n)x^n = \sum_{n\geqslant 0} x^n.$$

Согласно [106], композита производящей функции (5.8) *равна*

$$F^{\Delta}(n,k,a,b) = \binom{n-1}{k-1} a^{n-k}b^k.$$

Используя (5.7), *получим следующий критерий простоты числа: значение выражения*

$$\sum_{k=1}^{n-1} \binom{n-1}{k-1} \frac{a^{n-k}b^k}{k} = \frac{(a+b)^n - a^n - b^n}{n} \tag{5.9}$$

будет целым для любого простого числа n (a,$b \in \mathbb{Z}$).

После преобразования получим следующий вид критерия простоты числа: если n — *простое число,* a,$b \in \mathbb{Z}$, *то*

$$(a+b)^n - a^n - b^n \equiv 0 \pmod{n}. \tag{5.10}$$

Поскольку условие

$$a^n \equiv a \pmod{n}$$

справедливо для любого простого числа n *и целого числа* a, *тогда преобразуем* (5.10) *в следующий критерий простоты: если* n — *простое число, то*

$$(a+b)^n \equiv a+b \pmod{n}. \tag{5.11}$$

Подставляя c *вместо* $(a+b)$ *в формулу* (5.11), *получим малую теорему Ферма: если* n — *простое число, то для любого* $c \in \{1, 2, \ldots, n-1\}$ *справедливо*

$$c^{n-1} \equiv 1 \pmod{n}.$$

Малая теорема Ферма имеет важное значение в алгоритмической теории чисел как основа для многих известных алгоритмов проверки на простоту: тест на простоту на малой теореме Ферма, тест на простоту Соловея-Штрассена, тест Рабина-Миллера, тест на простоту AKS, и другие [255]. Первые три теста на простоту являются вероятностными и имеют полиномиальную сложность, широко применяются на практике, тест на простоту AKS является детерминированным с полиномиальной сложностью.

Если подставить $a = 1$ *и* $b = 1$ *в* (5.8) *для* (5.6), *то получим*

$$\begin{aligned} ng(n) &= \{1, 3, 7, 15, 31, 63, 127, 255, 511, 1023, \ldots\} \\ &= 2^n - 1 \\ &= M_n, \end{aligned}$$

где M_n — *числа Мерсенна [253] (последовательность* A000225 *в [94]).*

Таким образом, используя Следствие 10, получим следующий известный критерий простоты числа: если n — *простое число, то*

$$M_n \equiv 1 \pmod{n}.$$

Пример 24. *Пусть задана композиция вида* $G(x) = R(F(x))$, *где*

$$F(x) = ax + bx^2 \tag{5.12}$$

является обыкновенной производящей функцией с целыми коэффициентами, a *и* b *— целые числа,* $R(x) = \int B(x)dx$,

$$B(x) = \frac{1}{1-x} = \sum_{n\geqslant 0} b(n)x^n = \sum_{n\geqslant 0} x^n.$$

Согласно [106], композита производящей функции (5.12) *равна*

$$F^{\Delta}(n,k,a,b) = \binom{k}{n-k} a^{2k-n} b^{n-k}.$$

Используя (5.7), *получим следующий критерий простоты числа: значение выражения*

$$\sum_{k=1}^{n-1} \binom{k}{n-k} \frac{a^{2k-n}b^{n-k}}{k} = \frac{\left(\frac{a+\sqrt{a^2+4b}}{2}\right)^n + \left(\frac{a-\sqrt{a^2+4b}}{2}\right)^n - a^n}{n} \tag{5.13}$$

будет целым для любого простого числа n (a,$b \in \mathbb{Z}$).

Рассмотрим частные случаи параметров a *и* b.

Пусть $a = 1$ *и* $b = 1$ *для* (5.12) *в* (5.6). *Тогда*

$$\begin{aligned} ng(n) &= \{1,3,4,7,11,18,29,47,76,\ldots\} \\ &= \left(\frac{1+\sqrt{5}}{2}\right)^n + \left(\frac{1-\sqrt{5}}{2}\right)^n \\ &= L_n, \end{aligned}$$

где L_n *— числа Люка [258] (последовательность A*000032 *в [94]).*

Получим следующий известный критерий простоты числа: если n *— простое число, то*

$$L_n \equiv 1 \pmod{n}.$$

Пусть $a = 2$ *и* $b = 1$ *для* (5.12) *в* (5.6). *Тогда*

$$\begin{aligned} ng(n) &= \{2,6,14,34,82,198,478,1154,2786,\ldots\} \\ &= (1+\sqrt{2})^n + (1-\sqrt{2})^n \\ &= Q_n, \end{aligned}$$

где Q_n *— числа Пелля–Люка [259] (последовательность A*002203 *в [94]).*

Получим следующий известный критерий простоты числа [260]: если n — простое число, то

$$Q_n \equiv 2 \pmod{n}.$$

Пусть $a = 1$ и $b = 2$ для (5.12) в (5.6). Тогда получим

$$\begin{aligned} ng(n) &= \{1, 5, 7, 17, 31, 65, 127, 257, 511, 1025, \ldots\} \\ &= \left(\frac{1+\sqrt{5}}{2}\right)^n + \left(\frac{1-\sqrt{5}}{2}\right)^n \\ &= j_n, \end{aligned}$$

где j_n — числа Якобсталя–Люка [261] (последовательность A014551 в [94]).

Тогда получим следующий известный критерий простоты числа: если n — простое число, то

$$j_n \equiv 1 \pmod{n}.$$

Далее рассмотрим последовательность Люка [253; 262]. Пусть P и Q — целые числа больше 0. Корни полинома $X^2 - PX + Q$ будут равны

$$\left.\begin{matrix}\alpha \\ \beta\end{matrix}\right\} = \frac{P \pm \sqrt{P^2 - 4Q}}{2}.$$

Последовательность $(V_n(P,Q))_{n\geqslant 0}$ называется последовательностью Люка, ассоциированной с парой (P,Q), где

$$V_n(P,Q) = \alpha^n + \beta^n = \left(\frac{P+\sqrt{P^2-4Q}}{2}\right)^n + \left(\frac{P-\sqrt{P^2-4Q}}{2}\right)^n.$$

Пусть $a = P$ и $b = -Q$ для выражения (5.13). Тогда получим следующий критерий простоты числа: значение выражения

$$\frac{V_n(P,Q) - P^n}{n}$$

будет целым для любого простого числа n.

После преобразования получим следующий вид критерия простоты числа: если n — простое число, то

$$V_n(P,Q) \equiv P \pmod{n}.$$

В качестве производящей функции $B(x)$ для Теоремы 27 возможно применение производящих функций полиномов степени n с целыми коэффициентами.

Пример 25. *Пусть задана композиция производящих функций* $G(x) = R(F(x))$, *где* $F(x)$ *есть производящая функция* (5.8), $R(x) = \int B(x)dx$, *а также задана обыкновенная производящая функция с целыми коэффициентами*

$$B(x) = \sum_{n\geqslant 0} b(n)x^n,$$

где $b(n) = T_n(t)$ *для* $t \in \mathbb{Z}$ *и* $T_n(t)$ *представляет собой выражение для полинома Чебышева первого рода [263; 264].*

Явная формула для полиномов Чебышева первого рода [263] равна

$$T_n(t) = \frac{\left(t + \sqrt{t^2-1}\right)^n + \left(t - \sqrt{t^2-1}\right)^n}{2}.$$

Используя (5.7), *получим следующий критерий простоты числа: значение выражения*

$$\sum_{k=1}^{n-1} \binom{n-1}{k-1} \frac{a^{n-k}b^k}{k} \frac{\left(t + \sqrt{t^2-1}\right)^{k-1} + \left(t - \sqrt{t^2-1}\right)^{k-1}}{2}$$

будет целым для любого простого числа n (a,b,$t \in \mathbb{Z}$).

Например, для $t = 1$ *имеем* $T_n(t) = 1$, *тогда получим критерий простоты числа, приведенный в* (5.9).

Далее рассмотрим композицию производящих функций $G(x) = R(F(x))$, *где* $F(x)$ *есть производящая функция* (5.12), $R(x) = \int B(x)dx$, *а также задана обыкновенная производящая функция с целыми коэффициентами*

$$B(x) = \sum_{n\geqslant 0} b(n)x^n,$$

где $b_n = A_n(t)$ *для* $t \in \mathbb{Z}$ *и* $A_n(t)$ *представляет собой выражение для полинома Эйлера первого рода [265; 266].*

Явная формула для полиномов Эйлера первого рода [265] равна

$$A_n(t) = \sum_{m=0}^{n} A_{n,m}t^n = \sum_{m=0}^{n}\sum_{k=0}^{m+1} (-1)^k \binom{n+1}{k} (m+1-k)^n t^m.$$

Используя (5.7), *получим следующий критерий простоты числа: значение выражения*

$$\sum_{k=1}^{n-1} \binom{k}{n-k} \frac{a^{2k-n}b^{n-k}}{k} A_{k-1}(t)$$

будет целым для любого простого числа n (a,b,$t \in \mathbb{Z}$).

5.2.2 Критерии простоты числа на основе композиции экспоненциальной и обыкновенной производящих функций

Рассмотрим композицию экспоненциальной и обыкновенной производящих функций и получим соответствующие критерии простоты.

Теорема 28. *Пусть задана экспоненциальная производящая функция*

$$E(x) = \sum_{n>0} e(n)\frac{x^n}{n!},$$

где $e(n) \in \mathbb{Z}$*, и ее композита* $E^{\Delta}(n,k)$*. Тогда выражение*

$$\frac{n!}{k!}E^{\Delta}(n,k)$$

является целым для $k \leqslant n$*.*

Доказательство:

Рассмотрим композицию экспоненциальных производящих функций вида $A(E(x))$, где $A(x) = \sum\limits_{n\geqslant 0} a(n)\frac{x^n}{n!}$, $E(x) = \sum\limits_{n\geqslant 0} e(n)\frac{x^n}{n!}$, а также $a(n), e(n) \in \mathbb{Z}$. Как известно [58] композиция экспоненциальных производящих функций есть экспоненциальная производящая функция

$$G(x) = A(E(x)) = \sum_{n\geqslant 0} g(n)\frac{x^n}{n!},$$

где $g(n) \in \mathbb{Z}$. Тогда, используя композиционную формулу (2.2), можно записать

$$\frac{g(n)}{n!} = \sum_{k=1}^{n} E^{\Delta}(n,k)\frac{a(k)}{k!}.$$

Отсюда выражение

$$\sum_{k=1}^{n} \frac{n!}{k!}E^{\Delta}(n,k)a(k)$$

является целым.

Теперь методом от противного покажем, что выражение

$$\frac{n!}{k!}E^{\Delta}(n,k)a(k) \tag{5.14}$$

является целым. Положим, что при некоторых значениях k выражение (5.14) не является целым, а сумма является целой. Тогда, выбирая одно такое значение

k^*, для которого значение выражения (5.14) не является целым, и, подобрав такую производящую функцию $A(x)$, для которой $a(k^*) = 0$, получим, что сумма будет нецелой, — получается противоречие. Таким образом, выражение (5.14) является целым.

Кроме того, приравнивая все $a(k) = 1$, получим, что

$$\frac{n!}{k!}E^{\Delta}(n,k)$$

является целым для $k \leqslant n$. □

Следствие 12. *Пусть задана композита экспоненциальной производящей функции $E^{\Delta}(n,k)$. Тогда выражение*

$$\sum_{k=2}^{n-1} E^{\Delta}(n,k)\frac{(n-1)!}{k!} \tag{5.15}$$

является целым для всех простых n.

Доказательство:

Рассмотрим доказательство для следующих случаев:

– для $k = 1$ композита равна $E^{\Delta}(n,1) = \frac{e(n)}{n!}$. Откуда значение выражения

$$E^{\Delta}(n,k)\frac{(n-1)!}{k!} = \frac{e(n)}{n}$$

будет нецелым для $k = 1$.

– для $k = n$ композита равна $E^{\Delta}(n,n) = e(1)^n$. Откуда значение выражения

$$E^{\Delta}(n,k)\frac{(n-1)!}{k!} = \frac{e(1)^n}{n}$$

будет нецелым для $k = n$.

– для $1 < k < n$ согласно (2.1) композита равна

$$E^{\Delta}(n,k) = \sum_{\pi_k \in C_n} \frac{e(\lambda_1)e(\lambda_2)\dots e(\lambda_k)}{\lambda_1!\lambda_2!\dots\lambda_k!}.$$

Поскольку π_k есть такая композиция, что $\lambda_1 + \lambda_2 + \ldots + \lambda_k = n$ и $k > 1$, то не существует λ_i такой, что $\lambda_i = n$. Откуда значение выражения

$$\sum_{\pi_k \in C_n} \frac{e(\lambda_1)e(\lambda_2)\dots e(\lambda_k)}{\lambda_1!\lambda_2!\dots\lambda_k!}\frac{(n-1)!}{k!}$$

будет целым для любого простого n и $1 < k < n$.

Тогда значение выражения

$$\sum_{k=2}^{n-1} E^{\Delta}(n,k)\frac{(n-1)!}{k!}$$

будет целым для всех простых n. □

Выражение (5.15) можно записать в виде

$$\frac{1}{n}\left(n!\sum_{k=1}^{n} E^{\Delta}(n,k)\frac{1}{k!} - e(n) - e(1)^n\right),$$

где

$$g(n) = n!\sum_{k=1}^{n} E^{\Delta}(n,k)\frac{1}{k!}$$

представляют собой коэффициенты композиции экспоненциальных производящих функций вида $\exp(E(x))$.

В общем случае, для композиции производящих функций

$$G(x) = A(E(x)) = \sum_{n\geqslant 0} g(n)\frac{x^n}{n!},$$

где

$$A(x) = \sum_{n\geqslant 0} a(n)\frac{x^n}{n!}, \qquad E(x) = \sum_{n\geqslant 1} e(n)\frac{x^n}{n!},$$

выражение

$$\frac{1}{n}\left(g(n) - e_n a(1) - e(1)^n a(n)\right) \tag{5.16}$$

будет целым для всех простых n.

Пример 26. *Рассмотрим композицию производящих функций вида*

$$G(x) = \exp(\exp(x) - 1) = \sum_{n\geqslant 0} g(n)\frac{x^n}{n!}.$$

Композита производящей функции $E(x) = \exp(x) - 1$ *равна*

$$E^{\Delta}(n,k) = \frac{k!}{n!}\left\{{n \atop k}\right\}.$$

Тогда значение выражения

$$\frac{1}{n}\left(n!\sum_{k=1}^{n}\frac{k!}{n!}\left\{{n \atop k}\right\}\frac{1}{k!} - 1 - 1^n\right) = \frac{1}{n}\left(\sum_{k=1}^{n}\left\{{n \atop k}\right\} - 2\right)$$

будет целым для всех простых n.

Поскольку $n > 0$, то получим известный результат для чисел Белла: если n — простое число, то

$$(B_n - 2) \equiv 0 \mod n. \tag{5.17}$$

Здесь B_n есть числа Белла [2; 57]. Выражение (5.17) является частным случаем сравнения Тушара [267] для $k = 0$:

$$B_{n+k} \equiv B_{k+1} + B_k \pmod{n}$$

для простых чисел n.

Пример 27. *Рассмотрим производящую функцию для чисел Уппулури-Карпентера [268] (последовательность A000587 в [94])*

$$G(x) = e^{1-e^x} = \frac{1}{0!} + \frac{-1}{1!}x + \frac{0}{2!}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \frac{-2}{5!}x^5 + \ldots = \sum_{n\geqslant 0} \tilde{B}_n \frac{x^n}{n!}$$

как композицию экспоненциальных производящих функций $G(x) = A(E(x))$, где

$$A(x) = e^x = \frac{1}{0!} + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \frac{1}{5!}x^5 + \ldots = \sum_{n\geqslant 0} \frac{1}{n!}x^n$$

и

$$E(x) = 1 - e^x = \frac{-1}{1!}x + \frac{-1}{2!}x^2 + \frac{-1}{3!}x^3 + \frac{-1}{4!}x^4 + \frac{-1}{5!}x^5 + \ldots = \sum_{n>0} \frac{-1}{n!}x^n.$$

Подставляя значения в (5.16), получим, что значения выражения

$$\frac{g_n - e_n a_1 - e_1^n a_n}{n} = \frac{\tilde{B}_n - (-1)\cdot 1 - (-1)^n \cdot 1}{n} = \frac{\tilde{B}_n + 1 - (-1)^n}{n}$$

будут целыми для всех простых n.

Откуда получим следующий критерий простоты числа: если n — простое число, то

$$\tilde{B}_n \equiv (-1)^n - 1 \mod n.$$

Теорема 29. *Для композиции производящих функций вида $G(x) = B\left(E(x)\right) = \sum\limits_{n\geqslant 0} \frac{g(n)}{n!}\, x^n$, где $B(x) = \sum\limits_{n\geqslant 0} b(n)\, x^n$ является обыкновенной производящей функцией с $b(n) \in \mathbb{Z}$ и $E(x) = \sum\limits_{n>0} \frac{e(n)}{n!}\, x^n$ является экспоненциальной производящей функцией с $e(n) \in \mathbb{Z}$, значение выражения*

$$\frac{g(n) - e(n)b(1)}{n} \tag{5.18}$$

будет целым для всех простых n.

Доказательство:

Применяя композиционную формулу (2.2) для $G(x) = B\left(E(x)\right)$, получим

$$g(n) = n! \sum_{k=1}^{n} E^{\Delta}(n,k) b(k). \tag{5.19}$$

Поскольку $E(x)$ является экспоненциальной производящей функцией с целыми коэффициентами $e(n)$, то значение выражения

$$\frac{n!}{k!} E^{\Delta}(n,k)$$

будет целым для k,$n \in \mathbb{Z}$ и $k \leqslant n$. Тогда значения $g(n)$ в (5.19) будут целыми для $n \in \mathbb{N}$.

Рассмотрим частный случай для суммы в (5.3) при $k = 1$:

$$n! E^{\Delta}(n,k) b(k) = n! E^{\Delta}(n,1) b(1) = n! b(1) \sum_{\pi_1 \in C_n} \frac{e(\lambda_1)}{\lambda_1!} = n! b(1) \frac{e_n}{n!} = e(n) b(1).$$

Поскольку $e(i) \in \mathbb{Z}$ и $b(i) \in \mathbb{Z}$, тогда $e(n)b(1)$ будет целым для $n \in \mathbb{N}$ и значения выражения

$$g(n) - e(n)b(1) = n! \sum_{k=2}^{n} E^{\Delta}(n,k) b(k) = \tag{5.20}$$

$$= n! \sum_{k=2}^{n} b(k) \sum_{\pi_1 \in C_n} \frac{e(\lambda_1) e(\lambda_2) \cdots e(\lambda_k)}{\lambda_1! \lambda_1! \cdots \lambda_k!}$$

будут целыми для $n \in \mathbb{Z}$.

Если $k > 1$, то $\lambda_i < n$ $(i = \overline{1,k})$. Для простых n получим

$$\gcd(n, \lambda_1! \lambda_1! \cdots \lambda_k!) = 1.$$

Откуда для простых n можем разделить (5.20) на n и значения полученного выражения

$$\frac{g(n) - e(n)b(1)}{n}$$

также будут целыми. $\square$

Следствие 13. *Для композиции производящих функций вида* $G(x) = B\left(E(x)\right) = \sum\limits_{n\geqslant 0} \frac{g(n)}{n!}\, x^n$, *где* $B(x) = \sum\limits_{n\geqslant 0} b(n)\, x^n$ *является обыкновенной производящей функцией с* $b(n) \in \mathbb{Z}$ *и* $E(x) = \sum\limits_{n>0} \frac{e(n)}{n!}\, x^n$ *является экспоненциальной производящей функцией с* $e(n) \in \mathbb{Z}$, *справедливо: если* n — *простое число, то*

$$g(n) - e(n)b(1) \equiv 0 \mod n.$$

Пример 28. *Рассмотрим производящую функцию для «зигзаг» чисел Эйлера [30] (последовательность A*000111 *в [94])*

$$G(x) = \frac{1}{1-\sin x} = \frac{1}{0!} + \frac{1}{1!}x + \frac{2}{2!}x^2 + \frac{5}{3!}x^3 + \frac{16}{4!}x^4 + \frac{61}{5!}x^5 + \ldots = \sum_{n\geqslant 0} A_{n+1}\frac{x^n}{n!}$$

как композицию $G(x) = B\left(E(x)\right)$ *обыкновенной производящей функции*

$$B(x) = \frac{1}{1-x} = 1 + x + x^2 + x^3 + x^4 + x^5 + \ldots = \sum_{n\geqslant 0} x^n$$

и экспоненциальной производящей функции

$$E(x) = \sin x = \frac{1}{1!}x + \frac{0}{2!}x^2 + \frac{-1}{3!}x^3 + \frac{0}{4!}x^4 + \frac{1}{5!}x^5 + \ldots = \sum_{n>0} \frac{((-1)^{n-1}+1)(-1)^{\frac{n-1}{2}}}{2n!}x^n.$$

Подставляя значения в (5.18), *получим*

$$\frac{g(n) - e(n)b(1)}{n} = \frac{A_{n+1} - \frac{((-1)^{n-1}+1)(-1)^{\frac{n-1}{2}}}{2}}{n}.$$

Для нечетных n, *получим*

$$\frac{A_{n+1} - \frac{((-1)^{n-1}+1)(-1)^{\frac{n-1}{2}}}{2}}{n} = \frac{A_{n+1} - (-1)^{\frac{n-1}{2}}}{n}.$$

Согласно Теореме 29, значения выражения

$$\frac{A_{n+1} - (-1)^{\frac{n-1}{2}}}{n}$$

будут целыми для всех нечетных простых n. *Откуда получим следующий критерий простоты числа: если* n — *нечетное простое число, то*

$$A_{n+1} \equiv (-1)^{\frac{n-1}{2}} \mod n.$$

Пример 29. *Рассмотрим производящую функцию для чисел Фубини (последовательность A*000670 *в [94])*

$$G(x) = \frac{1}{2-e^x} - 1 = \frac{1}{1!}x + \frac{3}{2!}x^2 + \frac{13}{3!}x^3 + \frac{75}{4!}x^4 + \frac{541}{5!}x^5 + \ldots = \sum_{n>0} F_n \frac{x^n}{n!}$$

как композицию $G(x) = B\left(E(x)\right)$ *обыкновенной производящей функции*

$$B(x) = \frac{x}{1-x} = x + x^2 + x^3 + x^4 + x^5 + \ldots = \sum_{n>0} x^n$$

и экспоненциальной производящей функции

$$E(x) = e^x - 1 = \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \frac{1}{5!}x^5 + \frac{1}{6!}x^6 + \ldots = \sum_{n>0} \frac{1}{n!}x^n.$$

Тогда получим

$$G(x) = B\left(E(x)\right) = \frac{e^x - 1}{1 - (e^x - 1)} = \frac{e^x - 1}{2 - e^x} = \frac{1}{2 - e^x} - 1.$$

Подставляя значения в (5.18)*, получим, что значения выражения*

$$\frac{g(n) - e(n)b(1)}{n} = \frac{F_n - 1 \cdot 1}{n} = \frac{F_n - 1}{n}$$

будут целыми для всех простых n*. Откуда получим следующий критерий простоты числа: если* n *— простое число, то*

$$F_n \equiv 1 \mod n.$$

5.2.3 Рекуррентные критерии простоты числа

В качестве критерия простоты числа могут выступать и выражения, генерирующие последовательности, которые позволяют выявлять простые числа. Эта задача является смежной для генерации простых чисел. В основе решения данной задачи может лежать использование последовательностей целых чисел, удовлетворяющих линейным рекуррентным уравнениям различных порядков.

В 2008 году Е.С. Роуланд [269] получил рекуррентную последовательность, состоящую только из простых чисел, что делает возможным построение рекуррентного генератора простых чисел. Последовательность задается следующим образом: пусть $a_1 = 7$ и для всех $n > 1$ $a_n = a_{n-1} + \text{НОД}(n,a_{n-1})$, то для всех $n > 1$ выражение $b_n = a_n - a_{n-1}$ принимает только 1 и простые значения. Недостатки данного метода генерации заключаются в следующем:

– генерация происходит при помощи дополнительной операции отыскания наибольшего общего делителя;

– генерируются простые числа не подряд;

– множественные повторения простых чисел, особенно числа 1;

– неизвестно генерируются ли все простые числа.

В данном разделе рассмотрим рекуррентные критерии простоты числа, которые представляют собой генерирующие последовательности, поскольку, чтобы построить n-е по порядку число, необходимо знать предыдущие значения.

Для построения рекуррентных выражений, определяющих простоту числа, рассмотрим следующую композицию производящих функций:

$$G(x) = R\left(F(x)\right) = \sum_{n>0} g(n)\, x^n,$$

где внешней функцией является производящая функция $R(x) = \log\left(\frac{1}{1-x}\right)$, а внутренней — обыкновенная производящая функция $F(x)$ с целыми коэффициентами. Выше было показано, что для любых значений $n > 0$, значения выражения $ng(n)$ являются целыми числами. Поэтому для последовательности, задаваемой целочисленной функцией $ng(n)$ возможно построение рекуррентных функций, также однозначно определяющих заданную последовательность. Приводя полученную рекуррентную функцию к виду критерия простоты числа, в результате получаем рекуррентную функцию, которая генерирует последовательность чисел. Причем при простых значениях n, элемент последовательности однозначно будет целым. Обратное утверждение неверно.

Рассмотрим пару примеров, характеризующих описанные утверждения.

Пример 30. *Используя известную рекуррентную формулу для чисел Люка [258]*

$$L(n) = L(n-1) + L(n-2),$$

получим следующее рекуррентное выражение

$$\frac{L(n-1) + L(n-2) - 1}{n}.$$

Генерируя последовательность чисел от 1 *до* n

$$0,1,1,\frac{3}{2},2,\frac{17}{6},4,\frac{23}{4},\frac{25}{3},\frac{61}{5},18,\frac{107}{4},40,\frac{421}{7},\frac{1363}{15},\frac{1103}{8},210,\frac{577}{18},492,\ldots$$

видно, что целые значения принимают только элементы при простых порядковых номерах. В общем виде существуют и псевдопростые числа n*, элементы при которых также являются целыми.*

Пример 31. *Используя известную рекуррентную формулу для чисел Мерсенна*

$$M(n) = M(n-1) + 2M(n-2) + 2$$

получим следующее рекуррентное выражение

$$\frac{M(n-1) + 2M(n-2) + 1}{n}.$$

Генерируя последовательность чисел от 1 *до* n

$$0,1,2,\frac{7}{2},6,\frac{31}{3},18,\frac{127}{4},\frac{170}{3},\frac{511}{5},186,\frac{2047}{6},630,\frac{8191}{7},\frac{10922}{5},\frac{32767}{8},7710,\frac{131071}{9},\ldots$$

видно, что целые значения принимают только элементы при простых порядковых номерах. В общем виде существуют и псевдопростые числа n*, элементы при которых также являются целыми (такие числа в данной последовательности будут числа Сарруса).*

Используя полученные рекуррентные выражения возможно построение вероятностных генераторов простых чисел. В зависимости от подставляемых производящих функций генераторы простых чисел имеют различную точность определения простоты. Используя данный подход, можно находить множество чисел от 1 до n, в котором находятся все простые числа в этом интервале и некоторые составные числа, удовлетворяющие условию генерации, количество которых определяется композицией имеющихся производящих функций. Преимущество использования рекуррентных генераторов заключаются в том, что при знании некоторых необходимых элементов $a(k)$, определенной размерности, возможно дальнейшее построение простых чисел в интервале от k до n. То есть если хранить необходимое количество элементов последовательности $a(n)$, заданной большой размерности, то с легкостью можно отыскать множество простых и псевдопростых чисел больших, чем заданные значения n. В таком случае вся сложность генерации будет заключаться в операции деления на порядковый номер n и хранении нижней границы значений $a(n)$ необходимого порядка.

Применение рекуррентных генераторов простых чисел целесообразно для построения простых чисел в заданном интервале с некоторой допустимой погрешностью. Либо для построения или уменьшения размера последовательности чисел, необходимых для отыскания простых чисел с помощью алгоритмов проверки простоты числа. То есть строится некоторая последовательность чисел, для которых впоследствии применяются некоторые тесты простоты числа до тех пор, пока не найдется среди них простое число.

5.3 Программное обеспечение для анализа и генерации критериев простоты числа

5.3.1 Генератор критериев простоты числа

Разработанный метод построения критериев простоты числа на основе аппарата степеней производящих функций позволяет создавать большой набор новых критериев простоты числа, в зависимости от выбранных производящих функций. Данный процесс является трудоемким, поскольку многие расчеты делаются вручную или с использованием математических пакетов с ручным анализом, и однотипным с точки зрения проводимых операций, поэтому возможно автоматизировать данный процесс за счет разработки специального программного обеспечения. В рамках выполнения данной работы процесс создания новых критериев простоты числа был частично автоматизирован за счет разработки специализированного программного обеспечения — генератора критериев простоты натурального числа на основе свойств композиции производящих функций (Primality Criterion Generator — «PCG»).

Основываясь на использовании метода построения критериев простоты числа на основе аппарата степеней производящих функций и применении функциональных возможностей дополнительного программного обеспечения (система компьютерной верстки TeX, система компьютерной алгебры Maxima) сформулирован алгоритм работы генератора критериев простоты числа, который заключается в выполнении следующей последовательности действий:

Входом является производящая функция $F(x)$, параметры производящей функции $F(x)$, композита $F^{\Delta}(n,k)$, функция коэффициентов b_n внешней в композиции производящей функции;

Выходом является критерий простоты числа и его запись в формате на языке математических пакетов;

1. В программу «PCG» загружается информация из файла-библиотеки композит, который содержит: перечень производящих функций $F(x)$, вычисленные значения композит $F^{\Delta}(n,k)$;

2. Выбирается из загруженного списка доступных производящая функция $F(x)$ и выбираются используемые параметры функции $F(x)$;

3. С помощью встроенного ядра программы Maxima выполняются вычисления, и в результате в программе «PCG» отображаются изображения математических формул $F(x)$, f_n и $F^{\Delta}(n,k)$, необходимых для построения критерия простоты;

4. Вводится функция коэффициентов b_n внешней в композиции производящей функции;

5. С помощью встроенного ядра программы Maxima выполняются вычисления, и в результате в программе «PCG» отображаются изображения математических формул $B(x)$ и b_n, необходимых для построения критерия простоты;

6. Автоматически проверяется то, что производящие функции $F(x)$ и $B(x)$ подготовлены и соответствуют требованиям метода построения критериев простоты числа. Если они не соответствует, то возврат к этапу 1;

7. С помощью встроенного ядра программы Maxima выполняется вычисление критерия простоты числа;

8. Полученный критерий простоты числа и его запись на языке математических пакетов отображаются для пользователя в программе «PCG». Также отображаются значения двух целочисленных последовательностей, с помощью которых можно попробовать вручную упростить выражение критерия простоты числа;

9. Если критерий простоты числа упрощать не надо, то критерий простоты числа готов и его можно сохранить для дальнейшего использования. Если требуется упростить критерий простоты числа, то пользователю необходимо ввести формулу, генерирующую одну из двух предложенных в программе «PCG» целочисленных последовательностей;

10. Программа «PCG» проверяет соответствие указанной пользователем формулы целочисленной последовательности. Если формула соответствует, то вычисляется и отображается для пользователя значение упрощенного выражения критерия простоты числа. Если формула не соответствует, то возврат к этапу 9.

Графический пользовательский интерфейс главной формы программы «PCG» состоит из нескольких функциональных модулей, отраженных в рабочих областях на рисунке 5.1.

Каждый модуль предназначен для выполнения конкретной задачи:

1. Модуль меню, функции которого направлены на сохранение полученных результатов, изменение настроек и справки;

2. Модуль, функции которого направлены на введение и отображение данных, касающихся производящей функции $F(x)$;

3. Модуль, функции которого направлены на введение и отображение данных, касающихся производящей функции $B(x)$;

4. Модуль, функции которого направлены на запуск процесса построения критерия простоты числа;

5. Модуль результатов, функции которого направлены на отображение полученного критерия простоты числа в виде изображения математической формулы;

6. Модуль результатов, функции которого направлены на отображение полученного критерия простоты числа в виде записи в формате, применяемом в математических пакетах;

7. Модуль отображения полученных целочисленных последовательностей, с помощью которых можно упростить критерий простоты числа;

8. Модуль упрощения полученного критерия простоты числа;

9. Модуль отображения упрощенного критерия простоты числа.

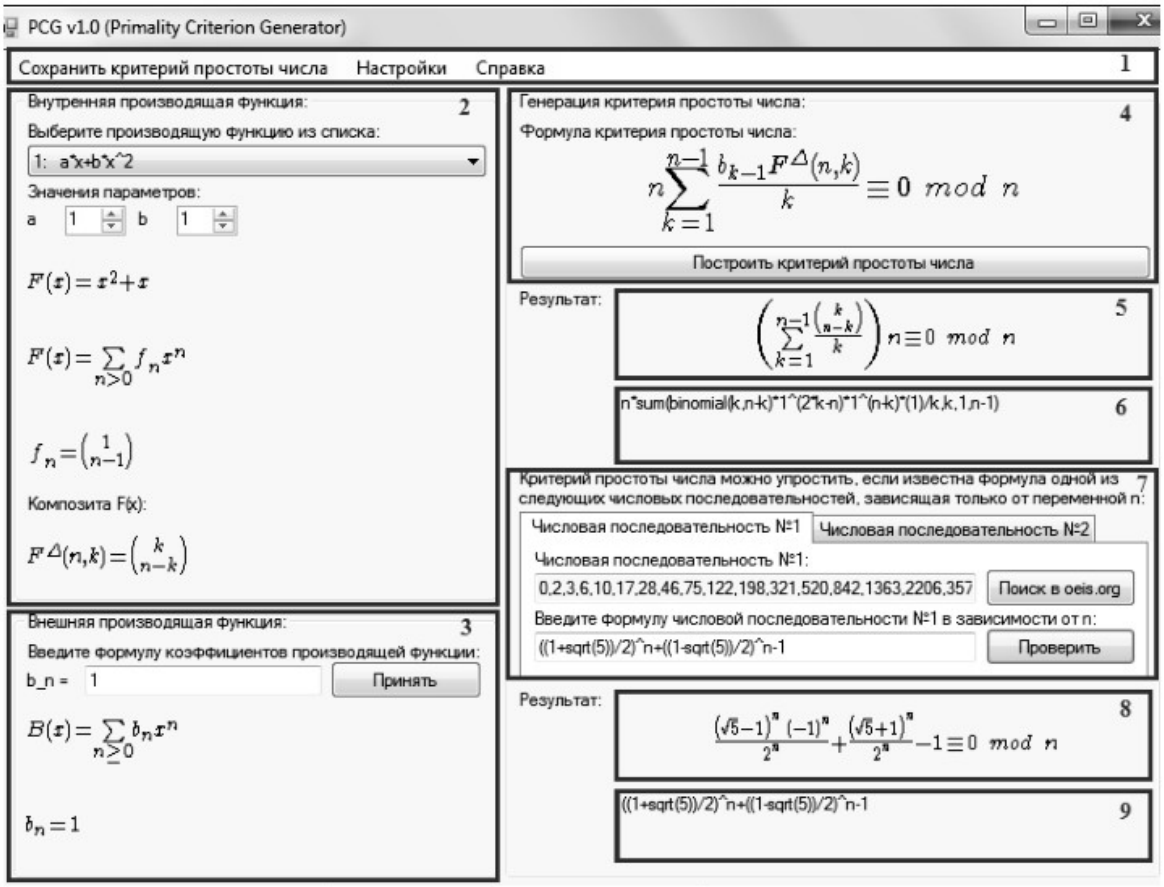


Рисунок 5.1 — Графический пользовательский интерфейс главной формы программы «PCG»

В качестве апробации использование данного программного обеспечения позволило сформировать более 700 критериев простоты числа для конкретных производящих функций. Например, на рисунке 5.1 показана генерация критерия простоты числа для производящих функций, рассмотренных в Примере 24. Для этого на вход программы были поданы:

– производящая функция $F(x) = ax + bx^2$;

– параметры $a = 1$, $b = 1$;

– производящая функция $B(x) = \sum_{n\geqslant 0} b_n x^n = \sum_{n\geqslant 0} x^n$.

В результате выполненных вычислений программой «PCG» получены:

– критерий простоты числа $n \sum_{k=1}^{n-1} \frac{\binom{k}{n-k}}{k} \equiv 0 \pmod{n}$;

– числовая последовательность [0,2,3,6,10,17,28,46,75, . . .].

При использовании модуля упрощения, программа предлагает упрощенный критерий простоты числа: если n — простое число, то

$$L_n \equiv 1 \qquad (\mathrm{mod}\ n).$$

5.3.2 Программное обеспечение для анализа и сравнения критериев простоты числа

Применение разработанного генератора критериев простоты числа позволило получить обширное количество различных критериев простоты числа за счет вариаций как самих производящих функций, так и их параметров. Поэтому возникла потребность в автоматизации процесса анализа полученных критериев простоты числа для сокращения времени проведения соответствующих исследований.

Разработанное программное обеспечение для анализа и сравнения критериев простоты числа состоит из следующих функциональных модулей (рис. 5.2):

1. Модуль ввода критериев простоты числа. Заключается в выборе критерия простоты (либо из имеющегося списка существующих критериев простоты числа, реализованных в самой программе, либо путем ввода нового критерия простоты, полученного при исследовании свойств композиции производящих функций);

2. Модуль проверки одного числа или интервала чисел. При проверке критерий простоты числа применяется либо для одного заданного пользователем числа, либо для указанного интервала натуральных чисел. Проверка одного числа необходима для точечной проверки простоты заданного числа на основе выбранного критерия простоты числа, а проверка интервала чисел используется для анализа указанного критерия простоты числа;

3. Модуль сравнения критериев простоты числа, позволяет обеспечить возможность сравнения двух различных критериев простоты числа с последующим проведением анализа данных критериев на одинаковых входных данных;

4. Модуль вывода, обеспечивает вывод сравнительной таблицы. При проведении анализа критерия простоты числа происходит вычисление параметров исследуемого критерия (время выполнения, подсчет числа совершенных ошибок в сравнении с другим критерием простоты, вероятность ошибки), что отображается в виде сравнительной таблицы. Данная таблица позволяет наглядно представить лучшие стороны сравниваемых критериев простоты числа относительно друг друга;

5. Модуль комбинирования двух критериев простоты числа, обеспечивает комбинирование двух разных критериев простоты. Использование двух разных критериев простоты на одних и тех же входных параметрах позволяет перекрыть «слепые зоны» каждого отдельного критерия простоты числа, что способствует уменьшению количества ошибок данных критериев за счет увеличения времени работы. Наличие такой функции в программе позволяет проводить более глубокий анализ на предмет точности критерия простоты.

Указанный набор функциональных возможностей программы делает возможным ее использование при решении задач, связанных с поиском эффективного способа проверки чисел на простоту.

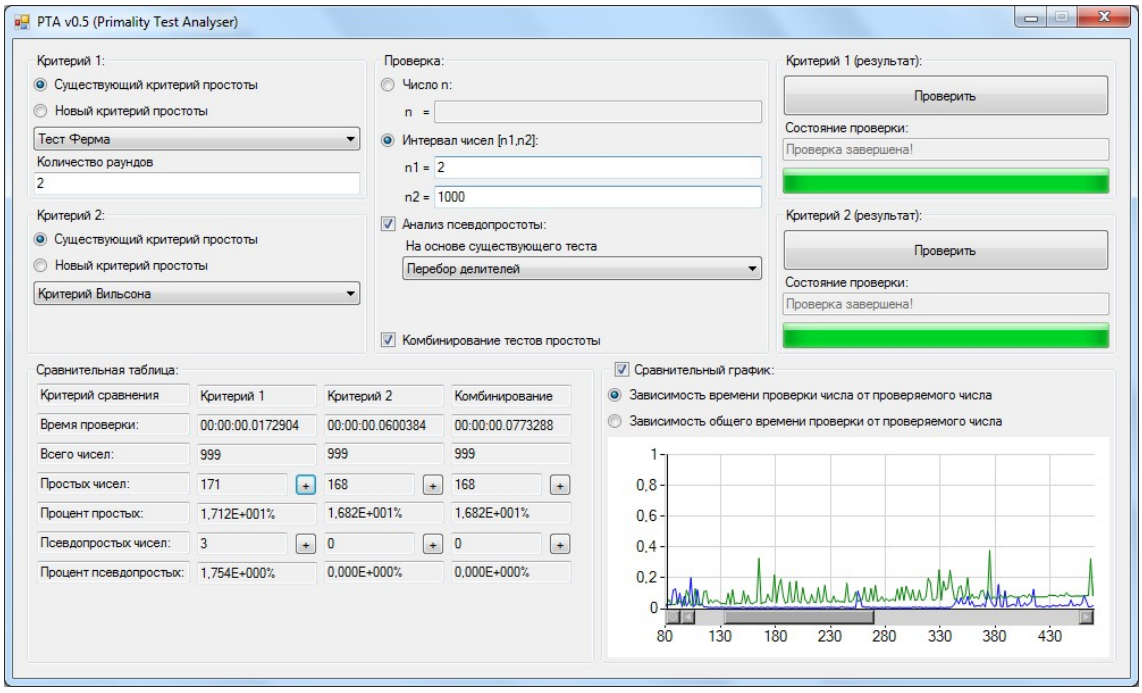


Рисунок 5.2 — Графический пользовательский интерфейс главной формы программы «PTA»

5.4 Выводы по главе

В данной главе изложен метод построения критериев простоты числа, который заключается в использовании свойств композиции производящих функций для формирования соответствующих выражений, справедливых для простых чисел. Применение данного метода позволяет после всех преобразований получить выражение, зависящее только от n, которое будет целым для простых n, т.е. критерий простоты числа для алгоритмов вероятностной проверки чисел на простоту. В зависимости от параметров композиции, а именно от внешней в композиции производящей функции и от композиты внутренней в композиции производящей функции, критерии простоты числа будут иметь разные числовые и вероятностные характеристики, а также оценки вычислительной сложности.

С применением разработанного метода получены критерии простоты числа на основе композиции логарифмической и обыкновенной производящих функций, композиции экспоненциальной и обыкновенной производящих функций, а также рекуррентные критерии простоты. Получены критерии простоты, связанные с числами Люка, Фибоначчи, Белла, полиномами Эйлера и другие. Показана взаимосвязь полученного метода с существующими алгоритмами (в частности, получен критерий простоты числа на основе малой теоремы Ферма).

Кроме того, на основе предложенного метода разработано специализированное программное обеспечение — генератор критериев простоты числа (Primality Criterion Generator). Использование данного программного обеспечения позволило сформировать более 700 критериев простоты числа для конкретных производящих функций. Также в дополнение реализован инструментарий для анализа и сравнения тестов и критериев простоты числа в виде специализированного программного обеспечения (Primality Test Analyser). Разработанное программное обеспечение позволяет значительно облегчить процесс анализа и сравнения тестов и критериев простоты числа за счет автоматизации данного процесса и представления итоговой информации в удобной для понимания и сравнения форме.

Благодаря полученным результатам определены теоретические основы для дальнейшего построения новых инструментальных средств, основанных на алгоритмах проверки чисел на простоту.

Результаты данной главы опубликованы в следующих публикациях [68; 270–279].

Глава 6. Программное обеспечение для математических пакетов

В данной главе представлены основные результаты в области разработки программного обеспечения для вычисления коэффициентов степеней производящих функций и для генерации по рангу элементов комбинаторных множеств в виде библиотек к математическим пакетам Maxima и Mathematica. С точки зрения автоматизации процессов вычисления коэффициентов степеней производящих функций созданы библиотеки к математическим пакетам, реализующие разработанные во второй главе методы получения явных выражений коэффициентов степеней производящих функций для одномерного, двумерного и трехмерного случаев, а также методы вычисления явных выражений полиномов и их композит. Кроме того, проведено сравнение предлагаемого подхода, основанного на применении композит, с существующими в математических пакетах решениями данной задачи. С точки зрения решения второй задачи создано программное обеспечение, направленное на автоматизацию вычислительных процессов в рамках работы разработанных алгоритмов комбинаторной генерации. Также в данной главе представлено описание итогов внедрения полученных результатов диссертационного исследования.

6.1 Библиотека методов получения явных выражений коэффициентов степеней производящих функций

Производящие функции являются неотъемлемой частью описания сложных информационных объектов, поэтому вычисление коэффициентов производящих функций требует автоматизированного решения. В современных системах компьютерной алгебры имеются программные модули для работы с производящими функциями. Однако они имеют ограниченные возможности по выполнению операций получения явных выражений композиции или обращения производящих функций и решения функциональных уравнений. Зачастую все это сводится к наличию в математическом пакете только инструмента по разложению функций в ряд Тейлора по одной переменной.

Кроме того, существуют частные решения данной задачи, но они направлены на небольшое количество операций, связанных с вычислением коэффициентов производящих функций. Например, для выполнения операции композиции производящих функций одной переменной известна реализация модуля к системе компьютерной алгебры Maxima [280], однако выполнение требуемых операций для производящих функций многих переменных в данном модуле не реализовано. Другим примером служит библиотека, представленная в [281], целью которой являлось нахождение символьных решений для рекуррентных соотношений и дифференциальных уравнений, основанных на производящих функциях.

Исследование математического аппарата производящих функций многих переменных, проведенное во второй главе, позволяет решить указанные задачи по получению явных выражений коэффициентов производящих функций многих переменных и найти подходы и алгоритмы для реализации этих операций в системах компьютерной алгебры. Важнейшее значение при выполнении этих операций играют коэффициенты k-й степени производящих функций многих переменных, поэтому создание соответствующего инструментального программного средства является актуальной задачей.

Описанный во второй главе метод, включая одномерный, двумерный и трехмерный случаи, был реализован в качестве библиотеки для математического пакета Mathematica, которая автоматизирует процесс нахождения коэффициентов производящих функций с использованием композит. Разработанная библиотека по работе с производящими функциями содержит:

1. Перечень композит для заданных производящих функций;
2. Правила вычисления композит;
3. Функции для представления информационных объектов на основе композит производящих функций.

Перечень композит для заданных производящих функций содержит 150 программных функций для вычисления композит производящих функций полиномиального, рационального, тригонометрического, гиперболического, логарифмического, экспоненциального и иррационального видов. Например, для композиты производящей функции $F(x) = xe^x$ применяется функция: `CompositaxExp[n,k]`, где n — номер элемента формального степенного ряда; k — показатель степени производящей функции. Данная функция вычисляет коэффициент при x^n для k-й степени производящей функции $F(x)$. Так, в качестве результата для `CompositaxExp[7,1]`, будет возвращено значение $\frac{1}{720}$.

Правила вычисления композит содержат 26 программных функций для вычисления коэффициентов и композит суммы, умножения, композиции, взаимных и обратных производящих функций, а также вывода найденных коэффициентов и композит производящих функций в формате таблиц и математических выражений.

Раздел, связанный с применением композит, содержит функции для:

– нахождения композит производящих функций, заданных уравнением;

– нахождения коэффициентов диагоналей треугольника композит;

– решения рекуррентных уравнений с производящими функциями;

– поиска критериев простоты числа на основе свойств композиции обыкновенных производящих функций с целыми коэффициентами.

Существующие решения в современных математических пакетах для одномерного случая, в отличие от многомерного, позволяют вычислять коэффициенты производящих функций за счет встроенных инструментов. Например, для разложения в ряд используются следующие встроенные функции, реализующие разложение функции в ряд Тейлора:

1. Mathematica: функция `Series` [282];
2. Maple: функция `taylor` [283];
3. Maxima: функция `taylor` [284];
4. MATLAB: функция `taylor` [285].

Для начала рассмотрим случай производящих функций одной переменной, поскольку данный процесс сравним с разложением функции в ряд Тейлора. Сравнить между собой приведенные выше инструменты по разложению функции в ряд Тейлора с точки зрения теоретических оценок вычислительной сложности не представляется возможным, поскольку исходный код данного программного обеспечения является закрытым. Поэтому сравнение проводилось эмпирически по времени вычисления и по затраченной оперативной памяти [165; 286]. Для этого были сформированы 3 группы производящих функций, сгруппированных по сложности их конструкций.

Первая группа производящих функций — это следующие 10 производящих функций, состоящих из двух простейших функций и одной операции над ними:

1. $\operatorname{tg}(\sin(x))$;
2. $\dfrac{x}{1-x}(\cos(x)-1)$;
3. $\dfrac{1-\sqrt{1-4x}}{2}+\arctan(x)$;
4. $\ln(1+x)e^x$;

5. $\sin\left(3x - 2x^2 + x^3\right)$;
6. $\cosh(x)\dfrac{x}{1-x}$;
7. $x\sqrt{1-x} + \dfrac{x}{(1-x)^2}$;
8. $\cos(x) + \sec(x) - 2$;
9. $\dfrac{x}{1-x-x^2}\arctan(x)$;
10. $\sqrt{1-x}\ln(1+x)$.

Вторая группа производящих функций — это следующие 10 производящих функций, состоящих из трех простейших функций и двух операций над ними:

1. $\operatorname{tg}(\sin(x)) + \arctan(x)$;
2. $\sin(x) + \dfrac{x\left(\cos(x)-1\right)}{1-x}$;
3. $\dfrac{1-\sqrt{1-4x}}{2}\ln(1+x) + (2x + 3x^2)$;
4. $\sin\left(5x + 3x^2 + x^3\right) + \dfrac{x}{1-x}$;
5. $\dfrac{x}{1-x}\cosh(x) + \arctan(x)$;
6. $\cos(x) + \sec(x) + xe^x - 2$;
7. $\dfrac{x}{1-x}\left(\cos(x)-1\right) + x\sqrt{1-4x}$;
8. $xe^x\ln(1+x) + \dfrac{x}{1-x-x^2}$;
9. $\left(11x - 4x^2 - 2x^3\right)\left(\sin(x) + \dfrac{1-\sqrt{1-4x}}{2}\right)$;
10. $(e^x - 1)\left(\dfrac{1-\sqrt{1-4x}}{2} + \dfrac{x}{1-x}\right)$.

Третья группа производящих функций — это следующие 10 производящих функций, состоящих из четырех функций и трех операций над ними:

1. $\operatorname{tg}(\sin(x)) + \dfrac{1-\sqrt{1-4x}}{2} + \arctan(x)$;
2. $\dfrac{\sin(x)}{1-\sin(x)} + \left(x\sqrt{1-x}\ln(1+x)\right)$;
3. $\left(xe^x + x\sqrt{1-4x}\right)\left(\left(\dfrac{x}{1-x} - x^2\right) + \left(12x - 3x^2\right)\right)$;
4. $\left(\dfrac{x}{1-x}xe^x\right) + \left(\dfrac{x}{1-x-x^2}\ln(1-x)\right)$;

5. $\dfrac{1-\sqrt{1-4x}}{2}\left(1-\sqrt[4]{1-x}\right)+\dfrac{x+x^2}{(1-x)^3}+\left(-17x-3x^2+10x^3\right)$;
6. $\left((e^x-1)+\arctan(x)\right)\left(\ln(1+x)+x^2\right)$;
7. $(\cos(x)+\sec(x)-2)+\left(\dfrac{x}{1-x-x^2}+\dfrac{x+x^2}{(1-x)^3}\right)$;
8. $\dfrac{x}{1-x}\left(\cosh(x)-1\right)+(e^x-1)\left(-7x+x^2+4x^3\right)$;
9. $\dfrac{x}{1-x}\left(\cos(x)-1\right)+\left(\dfrac{x}{1-x}-x^2\right)\left(3x-2x^2\right)$;
10. $\dfrac{x}{1-x-x^2}\arctan(x)+x\sqrt{1-4x}\ln(1+x)$.

Для каждой производящей функции из каждой группы были вычислены соответствующие им коэффициенты, при этом измерялось время вычислений и затраченная оперативная память. Вычисления выполнялись с помощью разных инструментов для всех значений n в интервале от 10 до 100 с шагом 10. В результате были вычислены средние значения времени вычислений и затраченной оперативной памяти по каждой из групп.

Для всей тестовой базы наименьшее среднее время вычислений показал метод, используемый в Mathematica, а наибольшее — метод, используемый в MATLAB. Метод, используемый в разработанной библиотеке, затрачивает не самое малое и не самое большое количество времени на вычисления относительно других методов, поэтому данный метод может использоваться для нахождения коэффициентов производящих функций. Методы, используемые в Maxima, Maple, MATLAB и разработанной библиотеке, имеют квадратичную зависимость времени вычисления коэффициентов производящих функций от параметра n. Независимость от параметра n для метода, используемого в Mathematica, может быть объяснена выделением большего количества оперативной памяти для вычислений и распараллеливанием вычислительных процессов.

Сравнение с точки зрения затрачиваемой оперативной памяти показало, что метод, используемый в Mathematica, затрачивает большее количество оперативной памяти на вычисления относительно других методов. Методы, используемые в разработанной библиотеке и Maple, имеют линейную зависимость затрачиваемой оперативной памяти на вычисления коэффициентов производящих функций от параметра n, а методы, используемые в Mathematica, Maxima и MATLAB, имеют квадратичную зависимость.

Таким образом, для нахождения коэффициентов производящих функций при небольших значениях n можно использовать любой из рассмотренных методов, в том числе и с помощью разработанной библиотеки. Если для пользователя важна скорость вычислений при больших n, то предпочтительнее будет использовать программное обеспечение Mathematica. Если же необходимо проводить вычисление коэффициентов производящих функций параллельно с другими вычислениями или на слабых машинах с ограниченными ресурсами, то наиболее предпочтительно выглядит использование разработанной библиотеки.

Из проведенного сравнения видно, что использование библиотеки для вычисления коэффициентов производящих функций оправдано при определенных условиях. Также можно сказать, что реализованный в библиотеке метод нахождения коэффициентов производящих функций, основанный на композитах производящих функций, не уступает другим рассмотренным методам, потому что тратит наименьшее количество оперативной памяти на вычисления и не самое большое количество времени на вычисления.

Для производящих функций многих переменных вычисление коэффициентов стандартными способами возможно только для простых производящих функций. Например, вычисление коэффициентов производящей функции $F(x,y) = e^{x+y}$ соразмерно как по времени, так и по объему затрачиваемой памятью. Однако при усложнении производящих функций за счет композиции с другими производящими функциями, стандартными средствами затруднительно посчитать соответствующие коэффициенты в отличие от подхода, основанного на разработанных во второй главе методах. Например, на рисунке 6.1 отражен сбой программы при попытке вычислить коэффициенты производящей функции двух переменных стандартными средствами системы компьютерной алгебры Maxima.

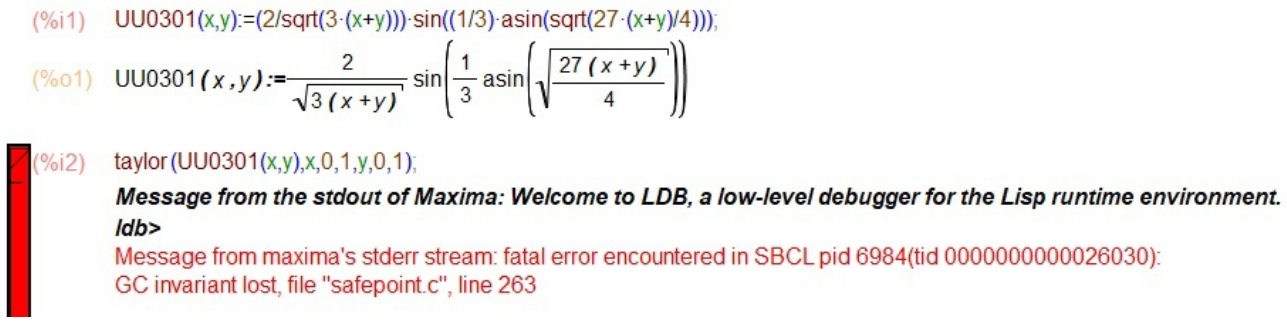


Рисунок 6.1 — Сбой программы Maxima при попытке вычислить коэффициенты производящей функции двух переменных

6.2 Библиотека для вычисления полиномов и их композит

В процессе анализа современных систем компьютерной алгебры было отмечено, что они содержат усеченный перечень функций вычисления явных формул для различных полиномов. Например, Mathematica содержит функции вычисления полиномов Чебышева, Гегенбауэра, Эрмита, Якоби, Лежандра и Белла, в то время как Maple содержит функции вычисления полиномов Бернулли, Чебышева, Эйлера, Фибоначчи, Эрмита, Якоби и Белла.

С другой стороны, в многих систем компьютерной алгебры отсутствуют функции вычисления таких полиномов как полиномы Наруми, Мотта, Малера, Лерча, Петерса и другие. Поэтому применение представленного во второй главе комплексного метода формирования информационных объектов, основанного на k-й степени производящих функций, для определения соответствующих полиномов позволило автоматизировать данный процесс за счет разработки библиотеки для вычисления полиномов и их композит. Разработка библиотеки основывалась на введенных правилах для работы с композитами производящих функций, а реализация данной библиотеки обеспечила автоматизацию соответствующих вычислительных процессов.

Разработанная библиотека содержит 23 программные функции для вычисления полиномов Абеля, Бернулли, Бесселя, Чебышева, Эйлера, Эйлера-Фробениуса, Гегенбауэра, Эрмита, Гумберта, Якоби, Лагерра, Лежандра, Лерча, Малера, Мейкснера, Мотта, Наруми, Петерса, Стирлинга и Белла. Рассмотрим подробно процесс вычисления полиномов на примере полиномов Белла [287]. Частичные полиномы Белла на бесконечном множестве переменных можно определить следующей производящей функцией [6]:

$$\sum_{n,k\geqslant 0} B_{n,k}(x_1, x_2, \ldots, x_{n-k+1})\frac{t^n}{n!}u^k = \exp\left(\sum_{m\geqslant 1} x_m \frac{t^m}{m!}u\right).$$

Чтобы использовать полиномы Белла для решения практических задач, необходимо получать их коэффициенты при степенях x. Это можно сделать, используя следующее явное выражение для частичных полиномы Белла:

$$B_{n,k}(x_1, x_2, \ldots, x_{n-k+1}) =$$

$$= \sum \frac{n!}{j_1! j_2! \cdots j_{n-k+1}!}\left(\frac{x_1}{1!}\right)^{j_1}\left(\frac{x_2}{2!}\right)^{j_2}\cdots\left(\frac{x_{n-k+1}}{(n-k+1)!}\right)^{j_{n-k+1}}, \qquad (6.1)$$

где суммирование ведется по всем $j_1, j_2, \ldots, j_{n-k+1}$ $(k = \overline{1,n})$, для которых

$$\begin{cases} j_1 + j_2 + \ldots + j_{n-k+1} = k, \\ j_1 + 2j_2 + \ldots + (n-k+1)j_{n-k+1} = n. \end{cases}$$

Также в практических задачах может возникнуть необходимость вычисления суммы частичных полиномов Белла для заданного значения n. Такая сумма называется n-полным полиномом Белла и имеет следующий вид:

$$B_n(x_1, x_2, \ldots, x_n) = \sum_{k=1}^{n} B_{n,k}(x_1, x_2, \ldots, x_{n-k+1}). \tag{6.2}$$

Разложение заданных параметров в многочлен для получения полиномов Белла в соответствии с определениями (6.1) и (6.2) имеет большую вычислительную сложность.

Также для получения полиномов Белла используются алгоритмы, основанные на рекурсивных формулах. В качестве примера можно привести выражения, описанные в [52],

$$kB_{n,k}(x_1, x_2, \ldots) = \sum_{i=1}^{n} \binom{n}{i} x_i B_{n-i,k-1}(x_1, x_2, \ldots),$$

$$B_{n,k}(x_1, x_2, \ldots) = \sum_{i=k-1}^{n-1} \binom{n-1}{i} x_{n-i} B_{i,k-1}(x_1, x_2, \ldots),$$

или выражение, приведенное в [288],

$$B_{n,k}(x_1, x_2, \ldots) = \sum_{i=1}^{n-k+1} \binom{n-1}{i-1} x_i B_{n-i,k-1}(x_1, x_2, \ldots). \tag{6.3}$$

В работе [289] доказывается справедливость следующего выражения для частичных полиномов Белла:

$$B_{n+1,k+1}(x_1, x_2, \ldots, x_{n+1}) = \sum_{i=0}^{n-k} \binom{n}{i} x_{i+1} B_{n-i,k}(x_1, x_2, \ldots, x_{n-i}).$$

Рекурсивные соотношения существуют и для n-полных полиномов Белла, например, приведенное в [290] соотношение имеет следующий вид:

$$B_n(x_1, \ldots, x_n) = \sum_{k=1}^{n} \binom{n-1}{k-1} x_k B_{n-k}(x_1, \ldots, x_{n-k}).$$

Однако способы, основанные на алгоритмах с применением рекурсивных соотношений, также имеют большую вычислительную сложность, оцениваемую как $O(n^2)$ [290]. Таким образом, одной из проблем, связанных с применением полиномов Белла на практике, является сложность вычисления их коэффициентов при степенях x. Поэтому поиск новых вычислительных способов нахождения полиномов Белла является актуальной задачей.

Помимо способов нахождения полиномов Белла через определение и через рекурсивные соотношения, существуют и другие вычислительные способы нахождения полиномов Белла. Одним из таких способов является использование математических пакетов. Функции для нахождения полиномов Белла реализованы в таких системах как Mathematica, Maple, Sage. Также существует пользовательская реализация способа вычисления частичных полиномов Белла на основе выражения (6.3) для MATLAB. В таблице 6.1 приведено описание программных функций, использующихся в этих математических пакетах для вычисления полиномов Белла.

Таблица 6.1 — Перечень функций для вычисления полиномов Белла в математических пакетах

Программа	Функция	Описание функции
Mathematica	BellY	Вычисляет частичный полином Белла для заданных значений n и k и последовательности $\{x_1, \ldots, x_{n-k+1}\}$
Maple	IncompleteBellB	Вычисляет частичный полином Белла для заданных значений n и k и последовательности $\{x_1, \ldots, x_n\}$
	CompleteBellB	Вычисляет n-полные полиномы Белла для заданного значения n и последовательности $\{x_1, \ldots, x_n\}$
Sage	bell_polynomial	Вычисляет частичный полином Белла для заданных значений n и k
MATLAB	IncompleteBellPoly	Вычисляет матрицу частичных полиномов Белла заданных значений n и k и последовательности $\{x_1, \ldots, x_n\}$

Существуют и другие вычислительные способы нахождения полиномов Белла. Например, в работе [291] рассматривается способ построения матрицы частичных полиномов Белла на основе алгоритма аналитического дифференцирования полинома. В данной работе уделяется особое внимание спецификации полинома, которая включает в себя коэффициенты слагаемых полинома, индексы и степени множителей, для хранения полинома в оперативной памяти, обращения к элементу по порядковому номеру и перебора всех элементов.

Применение разработанного метода вычисления коэффициентов степеней производящих функций позволяет автоматизировать процесс вычисления частичных полиномов Белла и n-полных полиномов Белла в виде библиотеки к математическому пакету Mathematica. Рассматривая полиномы Белла, в которых x_i имеет i-ю производную функции $y(x)$, можно получить взаимооднозначную связь частичных полиномов Белла и композит:

$$B_{n,k} = \frac{n!}{k!} Y^{\Delta}(n,k,x), \tag{6.4}$$

где

$$Y^{\Delta}(n,k,x) = \sum_{\pi_k \in C_n} \frac{y^{(\lambda_1)}(x)}{\lambda_1!} \frac{y^{(\lambda_2)}(x)}{\lambda_2!} \cdots \frac{y^{(\lambda_k)}(x)}{\lambda_k!}$$

является композитой обыкновенной производящей функции

$$Y(x,z) = y(x+z) - y(x) = \sum_{n>0} \frac{y^{(n)}(x)}{n!} z^n. \tag{6.5}$$

Для демонстрации корректности работы разработанной библиотеки сравним результат вычислений ее функций с результатом вычисления встроенной функции Mathematica. На рисунке 6.2 показан пример вычисления частичного полинома Белла функцией библиотеки `BellPoly` для $n = 5$ и $k = 2$, где последовательность $(x_1, x_2, \ldots, x_{n-k+1})$ является последовательностью производных функции $y(x) = \mathrm{tg}\,(x)$, а также с помощью встроенной функции Mathematica `BellY`.

В данном примере были использованы следующие функции библиотеки:

– функция `CompositaTan` вычисляет композиту функции $\mathrm{tg}\,(x)$ для задаваемых параметров n и k;

– функция `DeTan` вычисляет композиту вида (6.4) функции $y(x) = \mathrm{tg}\,(x)$ для задаваемых параметров n, k и x;

– функция `BellPoly` вычисляет частичный полином Белла для задаваемой композиты вида $Y^{\Delta}(n,k,x)$ и параметров n, k и x.

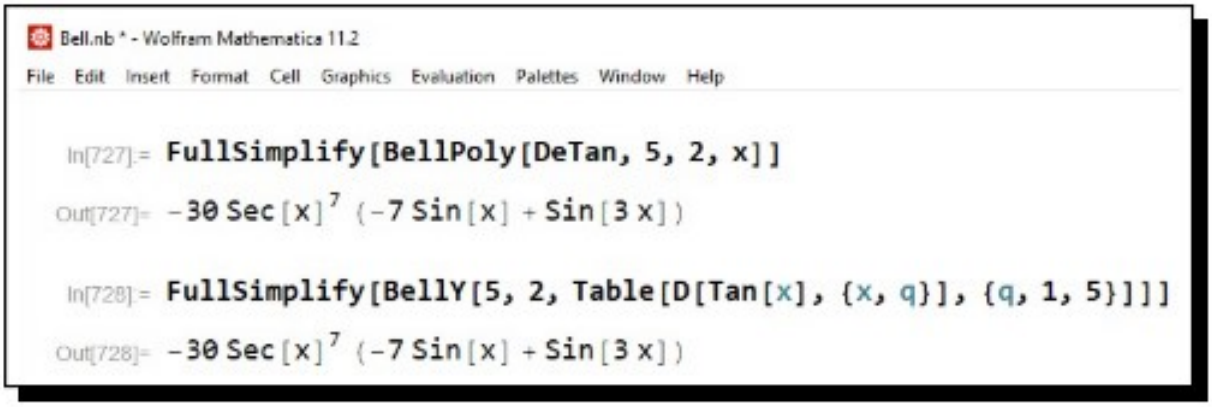


Рисунок 6.2 — Вычисление частичного полинома Белла для функции $y(x) = \mathrm{tg}\,(x)$

Сравнение способа нахождения полиномов Белла, основанного на композитах производящих функций, проводилось со способами, встроенными в Mathematica и Maple. Способы, реализованные в Sage и в [291], не позволяют находить полиномы Белла, в которых x_i является i-й производной функции $y(x)$, так как данные способы не предусматривают возможности вводить последовательность $(x_1, x_2, \ldots, x_{n-k+1})$. Способ, реализованный в MATLAB, основан на простом рекуррентном соотношении и его вычислительная сложность известна.

Выбранные для сравнения математические пакеты Mathematica и Maple являются проприетарным программным обеспечением, поэтому алгоритмы реализованных в данных системах вычислительных способов нахождения полиномов Белла не находятся в свободном доступе. Исходя из этого, сравнение способов вычисления полиномов Белла проводилось на основе эмпирически полученных данных, а в качестве критериев сравнения выступали время и память, затраченные на вычисления.

Для проведения сравнения была составлена тестовая база производящих функций, которая состоит из трех разделов. Каждый из разделов отличается сложностью входящих в него производящих функций. Первый раздел тестовой базы состоит из простейших функций, второй раздел состоит из сложных функций, которые составлены из двух простейших функций и одной операции над ними (сложение, умножение, композиция), третий раздел состоит из сложных функций, которые составлены из трех функций и двух операций над ними. Каждый из разделов содержит по 10 функций, подробный перечень данных функций приведен в таблице 6.2.

Таблица 6.2 — Тестовая база производящих функций

№	Первый раздел	Второй раздел	Третий раздел
1	$\sin(x)$	$\sin\left(\operatorname{tg}\left(x\right)\right)$	$\sin\left(\operatorname{tg}\left(\frac{1}{x}\right)\right)$
2	$\operatorname{tg}(x)$	$\ln\left(x\right)+xe^{x}$	$\operatorname{tg}\left(\ln^{2}\left(x\right)\right)$
3	$\sqrt{x}$	$\cos\left(2x-x^{2}+x^{3}\right)$	$e^{\frac{1-x}{1+x}}+\operatorname{tg}\left(x\right)$
4	xe^{x}	$\sqrt{x}+\frac{1-x}{1+x}$	$\ln\left(x\right)+e^{x}+\frac{x}{\sqrt{1-x^{2}}}$
5	$\frac{1-x}{1+x}$	$\sqrt{\operatorname{tg}\left(x\right)}$	$\sqrt{4x-x^{2}+2x^{3}}+\frac{1-x}{1+x}$
6	$\frac{1}{x}$	$\frac{1}{1-x-x^{2}}+x^{2}$	$\ln\left(\sqrt{\frac{1}{x}}\right)$
7	$\sqrt{1-x^{2}}$	$5x-3x^{2}+\ln\left(x\right)$	$\frac{1-x}{1+x}+\operatorname{tg}^{2}\left(x\right)$
8	$\arccos\left(x\right)$	$\sqrt{4x-x^{2}+2x^{3}}$	$\frac{1}{x+3x^{2}-2x^{3}}+x^{2}$
9	$\sec\left(x\right)$	$e^{x}+x+\frac{1}{x}$	$\sqrt{\ln\left(2x-x^{2}+x^{3}\right)}$
10	$-3x+2x^{2}+5x^{3}$	$\frac{1-\sqrt{x}}{1+\sqrt{x}}$	$\frac{1-\sqrt{x}}{1+\sqrt{x}}+e^{x}$

Для сравнения полученных результатов были подсчитаны средние значения затраченных на вычисления времени и оперативной памяти как по каждому из разделов тестовой базы, так и по всей тестовой базе. В таблице 6.3 приведено среднее значение времени вычислений по каждому разделу тестовой базы и в среднем по всей тестовой базе.

Таблица 6.3 — Среднее значение времени вычислений (в секундах)

Вычислительный способ	Раздел тестовой базы			
	Первый	Второй	Третий	Вся база
Разработанная библиотека	3,82	13,50	106,34	41,22
Mathematica	24,05	24,45	124,92	57,81
Maple	79,18	407,04	1518,83	668,35

Из полученных результатов видно, что с переходом от меньшего раздела тестовой базы к большему увеличивается время вычисления n-полных полиномов Белла всеми способами. Наименьший прирост времени наблюдался у Mathematica — время вычисления полиномов Белла для функций третьего раздела тестовой базы больше времени вычисления для функций первого раздела тестовой базы

в 5,19 раз. Наибольший прирост времени для разработанной библиотеки — увеличение в 27,84 раз. Прирост времени для Maple — увеличение в 19,19 раз. Но в абсолютных показателях среднее значение времени вычисления способом, реализованным в разработанной библиотеке, является меньше других способов по всем разделам тестовой базы.

В таблице 6.4 приведено среднее значение количества затраченной на вычисления оперативной памяти по каждому разделу тестовой базы и в среднем по всей тестовой базе.

Таблица 6.4 — Среднее значение количества затраченной на вычисления оперативной памяти (в мегабайтах)

Вычислительный способ	Раздел тестовой базы			
	Первый	Второй	Третий	Вся база
Разработанная библиотека	5,35	166,33	432,92	201,53
Mathematica	6360,51	29141,23	243032,40	92844,71
Maple	3117,14	16004,49	187527,97	68883,20

Из полученных результатов видно, что с переходом от меньшего раздела тестовой базы к большему увеличивается количество затрачиваемой на вычисления n-полных полиномов Белла оперативной памяти. Наименьший прирост памяти наблюдался у Mathematica — количество затрачиваемой на вычисления полиномов Белла оперативной памяти для функций третьего раздела тестовой базы больше по сравнению с функциями первого раздела тестовой базы в 38,21 раз. Наибольший прирост необходимой на вычисления памяти для разработанной библиотеки — увеличение в 80,92 раз. Прирост необходимой на вычисления памяти для Maple — увеличение в 60,16 раз. Но в абсолютных показателях среднее значение количества затраченной на вычисления оперативной памяти способом, реализованным в разработанной библиотеке, является меньше других способов по всем разделам тестовой базы.

Таким образом, реализованный метод вычисления полиномов Белла на основе применения композит производящих функций тратит меньше ресурсов, чем методы, используемые в математических пакетах Mathematica и Maple. По реализованным способам вычисления других полиномов результаты соизмеримы с результатами вычисления полиномов Белла.

6.3 Программное обеспечение комбинаторной генерации

С целью автоматизации вычислительных процессов в рамках работы разработанных в третьей главе алгоритмов комбинаторной генерации было создано соответствующее программное обеспечение, которое представляет собой совокупность подключаемых библиотек к системе компьютерной алгебры «Maxima». При этом структура каждой отдельно взятой библиотеки является набором модулей, реализующих необходимые расчеты для конкретного комбинаторного множества. Общий вид структуры библиотеки представлен на рисунке 6.3.

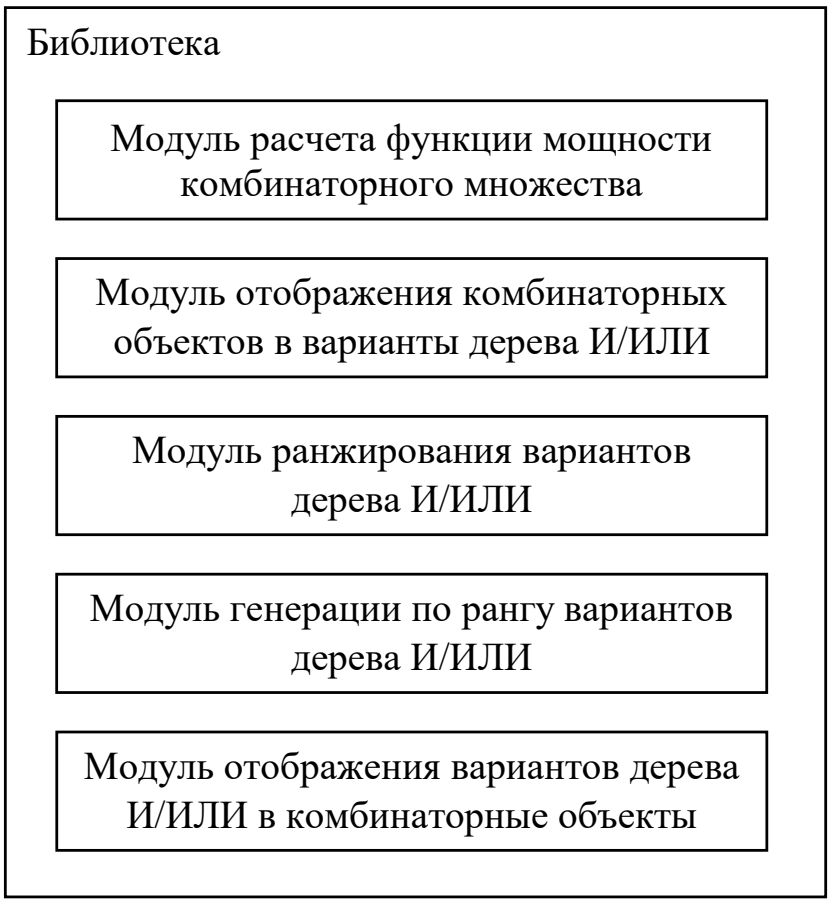


Рисунок 6.3 — Общий вид структуры библиотеки с разработанными алгоритмами комбинаторной генерации

Для работы с такой библиотекой необходимо ее подключить к основному рабочему листу программы «Maxima» воспользовавшись функцией `load()`.

Далее представлено краткое описание предназначения каждого модуля библиотеки:

– модуль расчета функции мощности комбинаторного множества: по заданному набору параметров комбинаторного множества $A_{n,m,\ldots,l}$ выполняется вычисление значения его функции мощности $f(n,m,\ldots,l) = |A_{n,m,\ldots,l}|$;

– модуль отображения комбинаторных объектов в варианты дерева И/ИЛИ: по заданному комбинаторному объекту $a \in A_{n,m,\ldots,l}$ выполняется вычисление соответствующего ему варианта v дерева И/ИЛИ D, то есть реализуется отображение $\texttt{ObjectToVariant}(a, D) : A_{n,m,\ldots,l} \to W(D)$;

– модуль ранжирования вариантов дерева И/ИЛИ: по заданному варианту $v \in W(D)$ дерева И/ИЛИ D выполняется вычисление соответствующего ему ранга r, то есть реализуется отображение $\texttt{RankVariant}(v, D) : W(D) \to \mathbb{N}_{|W(D)|}$;

– модуль генерации по рангу вариантов дерева И/ИЛИ: по заданному рангу $r \in \mathbb{N}_{|W(D)|}$ выполняется вычисление соответствующего ему варианта v дерева И/ИЛИ D, то есть реализуется обратное отображение $\texttt{UnrankVariant}(r, D) : \mathbb{N}_{|W(D)|} \to W(D)$;

– модуль отображения вариантов дерева И/ИЛИ в комбинаторные объекты: по заданному варианту v дерева И/ИЛИ D выполняется вычисление соответствующего ему комбинаторного объекта $a \in A_{n,m,\ldots,l}$, то есть реализуется обратное отображение $\texttt{VariantToObject}(v, D) : W(D) \to A_{n,m,\ldots,l}$.

Созданная совокупность библиотек реализует разработанные в ходе проведения диссертационного исследования и подробно представленные в третьей главе алгоритмы комбинаторной генерации для следующего перечня комбинаторных множеств:

– множество сочетаний из n элементов по m;

– множество самонепересекающихся решеточных путей на плоскости из точки (0,0) в точку (n,n) с шагами вида (0,1) и (1,0) и двумя шагами вида $(-1,0)$;

– множество помеченных путей Дика длины $2n$ с m подъемами на возвратных шагах;

– множество путей Дика длины $2n$ с m пиками, образованных l путями Дика меньшей длины;

– множество последовательностей из не менее t правильных ответов на закрытый тест, который содержит n вопросов с m вариантами ответа на каждый из них;

– множество исходов событий турнира на выбывание при участии в нем 2^{n-1} игроков;

– множество частей круга, полученных при разрезе его поверхности n прямыми линиями таким образом, чтобы количество получаемых частей было максимально возможным;

– множество последовательностей длины n, представляющих собой последовательности правильно вложенных скобок, разряженных нулями;

– множество разбиений множества n элементов на непересекающиеся непустые подмножества.

Рассмотрим пример внутренней структуры библиотеки на конкретном комбинаторном множестве путей Дика длины $2n$ с m пиками, образованных l путями Дика меньшей длины.

На рисунке 6.4 представлено внутреннее содержание модуля расчета функции мощности комбинаторного множества на основе явного выражения (3.36).

```
N(n,m,l):=
if n>=m and m>=l and l>0
   then (l/n)·binomial(n,m−l)·binomial(n,m)
   else
   (
      if n=m and m=l
         then 1
         else 0
   );
```

Рисунок 6.4 — Модуль расчета функции мощности комбинаторного множества

На рисунке 6.5 представлено внутреннее содержание модуля ранжирования вариантов дерева И/ИЛИ, реализованного на основе применения разработанного Алгоритма 11.

На рисунке 6.6 представлено внутреннее содержание модуля генерации по рангу вариантов дерева И/ИЛИ, реализованного на основе применения разработанного Алгоритма 12.

На рисунке 6.7 представлено внутреннее содержание модуля отображения вариантов дерева И/ИЛИ в комбинаторные объекты.

На рисунке 6.8 представлено внутреннее содержание модуля отображения комбинаторных объектов в варианты дерева И/ИЛИ.

```
RankVariant(v,n,m,l):=
block
(
   [r,S0,S1,S2,S3],
   if n=m and m=l then r:0
   else if v[1]=1 then (S0:0, r:S0+RankVariant(rest(v,1),n-1,m-1,l-1))
   else if v[1]=2 then (S1:N(n-1,m-1,l-1), r:S1+RankVariant(rest(v,1),n-1,m-1,l))
   else if v[1]=3 then (S2:N(n-1,m-1,l-1)+N(n-1,m-1,l), r:S2+RankVariant(rest(v,1),n-1,m,l))
   else if v[1]=4 then (S3:N(n-1,m-1,l-1)+N(n-1,m-1,l)+N(n-1,m,l), r:S3+RankVariant(rest(v,1),n-1,m,l+1)),
   return(r)
);
```

Рисунок 6.5 — Модуль ранжирования вариантов дерева И/ИЛИ

```
UnrankVariant(r,n,m,l):=
block
(
   [v,S0,S1,S2,S3,S4],
   if n=m and m=l then v:[]
   else (S0:0, S1:N(n-1,m-1,l-1), if r<S1 then v:append([1],UnrankVariant(r-S0,n-1,m-1,l-1))
   else (S2:S1+N(n-1,m-1,l), if r<S2 then v:append([2],UnrankVariant(r-S1,n-1,m-1,l))
   else (S3:S2+N(n-1,m,l), if r<S3 then v:append([3],UnrankVariant(r-S2,n-1,m,l))
   else (S4:S3+N(n-1,m,l+1), if r<S4 then v:append([4],UnrankVariant(r-S3,n-1,m,l+1))
   )))),
   return(v)
);
```

Рисунок 6.6 — Модуль генерации по рангу вариантов дерева И/ИЛИ

```
VariantToObject(v,n,m,l):=
block
(
   [a],
   if n=m and m=l then a:makelist([0,1],i,1,n) else
   (
      if v[1]=1 then (a:cons([0,1],VariantToObject(rest(v,1),n-1,m-1,l-1)))
      else if v[1]=2 then (a:VariantToObject(rest(v,1),n-1,m-1,l), a[1]:endcons(1,endcons(0,a[1])))
      else if v[1]=3 then (a:VariantToObject(rest(v,1),n-1,m,l), a[1]:endcons(1,cons(0,a[1])))
      else if v[1]=4 then (a:VariantToObject(rest(v,1),n-1,m,l+1), a[2]:append(a[1],endcons(1,cons(0,a[2]))), a:rest(a,1))
   ),
   return(a)
);
```

Рисунок 6.7 — Модуль отображения вариантов дерева И/ИЛИ в комбинаторные объекты

```
ObjectToVariant(a,n,m,l):=
block
(
   [v],
   if n=m and m=l then v:[] else
   (
      if a[1]=[0,1] then (v:cons(1,ObjectToVariant(rest(a,1),n-1,m-1,l-1)))
      else if lastn(a[1],2)=[0,1] then (a[1]:rest(a[1],-2), v:cons(2,ObjectToVariant(a,n-1,m-1,l)))
      else if checka(makelist(a[1][i],i,2,length(a[1])-1)) then (a[1]:makelist(a[1][i],i,2,length(a[1])-1), v:cons(3,ObjectToVariant(a,n-1,m,l)))
      else v:cons(4,ObjectToVariant(append(dividea(a[1]),rest(a,1)),n-1,m,l+1))
   ),
   return(v)
);

checka(a):=
block
(
   [f,s],
   f:true,
   s:0,
   for i:1 thru length(a) do
   (
      if a[i]=0 then s:s+1,
      if a[i]=1 then s:s-1,
      if s<0 then (f:false, return(f))
   ),
   return(f)
);
dividea(a):=
block
(
   [a1,a2,s],
   s:0,
   for i:length(a)-1 thru 1 step -1 do
   (
      if a[i]=0 then s:s-1,
      if a[i]=1 then s:s+1,
      if s<0 then (a1:makelist(a[j],j,1,i-1), a2:makelist(a[j],j,i+1,length(a)-1), return([a1,a2]))
   ),
   return([a1,a2])
);
```

Рисунок 6.8 — Модуль отображения комбинаторных объектов в варианты дерева И/ИЛИ

Для демонстрации работоспособности выполним последовательно следующий перечень преобразований для комбинаторного множества с фиксированными значениями параметров $n = 5$, $m = 3$, $l = 2$:

– для каждого возможного значения ранга r сгенерируем соответствующий ему вариант v дерева И/ИЛИ;

– для каждого варианта v дерева И/ИЛИ выполним восстановление комбинаторного объекта a (в данном случаем каждый подъем представлен значением 0, а каждый спуск — значением 1);

– для каждого комбинаторного объекта a выполним обратное преобразование с целью получения соответствующего ему варианта v дерева И/ИЛИ;

– для каждого варианта v дерева И/ИЛИ вычислим соответствующий ему ранг r.

На рисунке 6.9 представлен полученный результат, где можно увидеть корректную работу восстановления комбинаторных объектов и их последующего кодирования в виде ранга.

```
(%i12) nn:5;
       mm:3;
       ll:2;
       NN:N(nn,mm,ll);
       for r:0 thru NN-1 do
       (
         w:UnrankVariant(r,nn,mm,ll),
         aa:VariantToObject(w,nn,mm,ll),
         ww:ObjectToVariant(copy(aa),nn,mm,ll),
         rr:RankVariant(ww,nn,mm,ll),
         print("r =",r, " => v =", w, " => a =", aa, " => v =", ww, " => r =", rr)
       );
(nn)   5
(mm)   3
(ll)   2
(NN)   20
       r = 0  => v = [1,2,3,3]  => a = [[0,1],[0,0,0,1,1,1,0,1]]  => v = [1,2,3,3]  => r = 0
       r = 1  => v = [1,3,2,3]  => a = [[0,1],[0,0,0,1,1,0,1,1]]  => v = [1,3,2,3]  => r = 1
       r = 2  => v = [1,3,3,2]  => a = [[0,1],[0,0,0,1,0,1,1,1]]  => v = [1,3,3,2]  => r = 2
       r = 3  => v = [1,3,4]  => a = [[0,1],[0,0,1,0,0,1,1,1]]  => v = [1,3,4]  => r = 3
       r = 4  => v = [1,4,1,3]  => a = [[0,1],[0,1,0,0,0,1,1,1]]  => v = [1,4,1,3]  => r = 4
       r = 5  => v = [1,4,3]  => a = [[0,1],[0,0,1,1,0,0,1,1]]  => v = [1,4,3]  => r = 5
       r = 6  => v = [2,1,3,3]  => a = [[0,1,0,1],[0,0,0,1,1,1]]  => v = [2,1,3,3]  => r = 6
       r = 7  => v = [2,3,1,3]  => a = [[0,0,1,1,0,1],[0,0,1,1]]  => v = [2,3,1,3]  => r = 7
       r = 8  => v = [2,3,3]  => a = [[0,0,0,1,1,1,0,1],[0,1]]  => v = [2,3,3]  => r = 8
       r = 9  => v = [3,1,2,3]  => a = [[0,0,1,1],[0,0,1,1,0,1]]  => v = [3,1,2,3]  => r = 9
       r = 10  => v = [3,1,3,2]  => a = [[0,0,1,1],[0,0,1,0,1,1]]  => v = [3,1,3,2]  => r = 10
       r = 11  => v = [3,1,4]  => a = [[0,0,1,1],[0,1,0,0,1,1]]  => v = [3,1,4]  => r = 11
       r = 12  => v = [3,2,1,3]  => a = [[0,0,1,0,1,1],[0,0,1,1]]  => v = [3,2,1,3]  => r = 12
       r = 13  => v = [3,2,3]  => a = [[0,0,0,1,1,0,1,1],[0,1]]  => v = [3,2,3]  => r = 13
       r = 14  => v = [3,3,1,2]  => a = [[0,0,0,1,1,1],[0,1,0,1]]  => v = [3,3,1,2]  => r = 14
       r = 15  => v = [3,3,2]  => a = [[0,0,0,1,0,1,1,1],[0,1]]  => v = [3,3,2]  => r = 15
       r = 16  => v = [3,4]  => a = [[0,0,1,0,0,1,1,1],[0,1]]  => v = [3,4]  => r = 16
       r = 17  => v = [4,1,1,3]  => a = [[0,1,0,0,1,1],[0,0,1,1]]  => v = [4,1,1,3]  => r = 17
       r = 18  => v = [4,1,3]  => a = [[0,1,0,0,0,1,1,1],[0,1]]  => v = [4,1,3]  => r = 18
       r = 19  => v = [4,3]  => a = [[0,0,1,1,0,0,1,1],[0,1]]  => v = [4,3]  => r = 19
(%o12) done
```

Рисунок 6.9 — Пример работы модулей разработанной библиотеки

6.4 Внедрение результатов диссертационной работы

Достоверность полученных результатов диссертационного исследования подтверждается их внедрением в деятельность научно-исследовательского института автоматики и электромеханики «НИИ АЭМ ТУСУР», АО «Информационные спутниковые системы» имени академика М. Ф. Решетнёва», ООО «ПлантаПлюс», ООО «Эль Контент», ФГАОУ ВО НИ ТПУ и ФГБОУ ВО «ТУСУР», что оказало положительный эффект (Приложение Б).

Результаты диссертационной работы были внедрены в деятельность «НИИ АЭМ ТУСУР», который является одним из ведущих НИИ Сибирского региона по разработке, изготовлению и внедрению автоматизированных комплексов для отработки и предстартовых испытаний энергопреобразующей аппаратуры космических аппаратов различного назначения, и в деятельность АО «Информационные спутниковые системы» имени академика М. Ф. Решетнёва», которое занимается проведением научно-исследовательских и опытно-конструкторских работ в области создания космической техники.

Современный космический аппарат представляет собой сложный взаимосвязанный комплекс технических систем различного назначения. Одной из основных систем космического аппарата является система электропитания. Система электропитания в свою очередь представляет собой совокупность источников тока, аппаратуры преобразования энергии и стабилизации выходного напряжения с необходимой автоматикой контроля и управления. Существенное увеличение сроков активного существования космических аппаратов требует проведения более детальных исследований как штатных, так и нештатных режимов для эксплуатации в различных режимах космического пространства. Например, при отработке и проведении предстартовых испытаний в наземных условиях энергопреобразующей аппаратуры космических аппаратов проверяются как отдельные составляющие системы электропитания, так и вся система в целом как в нормальных, так и в аварийных режимах работы. Поскольку полная проверка технических характеристик системы электропитания требует больших технических, временных и экономических ресурсов, то, помимо физического моделирования при наземных испытаниях, используют имитационно-физическое моделирование, при котором отдельные компоненты систем электропитания заменяются эквивалентным специализированным оборудованием (имитаторами) [292].

Имитаторы позволяют моделировать требуемые характеристики устройств при существенно меньших затратах и формировать имитацию всех режимов функционирования. Так, например, согласно техническим требованиям к имитаторам солнечных батарей автоматических космических аппаратов, имитаторы должны воспроизводить все возможные параметры выходного тока солнечных батарей основываясь на многообразии входящих воздействий, которым подвергается батарея, включая орбитальное вращение, осевое вращение, выравнивание осей, затмения, операции начала и завершения жизненного цикла. Система имитации состоит из совокупности таких имитаторов. Поэтому применяется автоматизированная система контроля, что позволяет существенно уменьшить срок отработки и снизить финансовые и технологические затраты, а также повысить качество испытаний при комплексной проверке оборудования космического аппарата перед запуском и на всех этапах проверок и экспериментальных отработок. Автоматизированная система контроля состоит из отдельных устройств, каждое из которых может работать как самостоятельно, так и как единое целое под управлением оператора через персональный компьютер, что обеспечивает реализацию различных рабочих режимов и алгоритмов функционирования системы [292], что наглядно представлено на рисунке 6.10.

Рисунок 6.10 — Автоматизированная система контроля энергопреобразующей аппаратуры

Каждое устройство (имитатор) отвечает за формирование своих выходных характеристик. Поскольку совокупность всевозможных выходных характеристик и режимов работы обширна, появляется задача генерации управляющих сигналов на автоматизированном рабочем месте оператора для формирования заданных выходных характеристик имитаторов.

На основе применения описанного в диссертационной работе подхода разработки алгоритмов комбинаторной генерации, были реализованы алгоритмы генерации последовательности управляющих сигналов, которые использовались для формирования выходных характеристик имитаторов. Программное обеспечение на основе разработанных алгоритмов в аппаратуре комплексного тестирования систем энергообеспечения космических аппаратов для всех шин питания 27В, 40В и 100В позволило снизить время тестирования систем энергообеспечения в среднем на 43%, что существенно уменьшило экономические затраты.

Результаты диссертационной работы были внедрены в деятельность ООО «ПлантаПлюс», которое занимается химическими и биологическими исследованиями и производством микробиологических препаратов для растениеводства. В рамках оперативной деятельности на предприятии был сформирован значительный массив экспериментальных данных. Поэтому модифицированный метод построения алгоритмов комбинаторной генерации был применен для разработки алгоритмов кодирования и декодирования микробиологических препаратов, представленных деревьями И/ИЛИ. Разработанные алгоритмы направлены на улучшение информационной системы хранения и обработки экспериментальных данных, связанных с исследованием и производством микробиологических препаратов для растениеводства. Применение предложенных в диссертационной работе подходов позволило сократить объем базы данных на 9% за счет уменьшения количества дублируемой информации и повысить скорость поиска и обработки данных на 5%.

Результаты диссертационной работы были внедрены в деятельность ООО «Эль Контент», которое занимается разработкой систем обучения, проектирования учебного контента и тестирования остаточных знаний. На основе разработанных алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ для комбинаторного множества последовательностей вариантов ответа на тест с вопросами закрытого типа был сформирован генератор, формирующий выборку ответов для тестирования системы оценивания. А именно для формирования последовательности из не менее t правильных ответов на закрытый

тест, который содержит n вопросов с m вариантами ответа на каждый из них. Применение разработанного генератора позволило уменьшить временные затраты на 50% во время тестирования системы оценивания формата «тест».

Результаты диссертационной работы были внедрены в деятельность ФГБОУ ВО «ТУСУР» в рамках цифровизации образовательной деятельности. В ходе создания автоматизированной системы обучения математическим дисциплинам была разработана программа для создания математических заданий путем генерации внутренних параметров и формированием обратной связи по анализу дерева ответов обучающегося. Данная программа представляет собой программный модуль на языке системы компьютерной алгебры Maxima в плагине STACK системы Moodle. Программа может использоваться для формирования комплекса адаптивных тренажеров с обратной связью при обучении школьной математике и высшей математике для студентов первых курсов. Внедрение предложенного решения позволило сократить временные затраты на создание и проверку контрольных и домашних работ за счет автоматизации данного процесса. Разработанные обучающие тренажеры были внедрены в учебный процесс для студентов первого курса по дисциплине «Математический анализ». В программе реализованы генераторы заданий по следующим разделам математики: графики, экстремумы, уравнения, неравенства, задачи теории вероятностей, логические задачи, теория пределов, дифференцирование и интегрирование. С использованием разработанных обучающих тренажеров было обучено более 500 студентов.

Результаты диссертационной работы были использованы в учебном процессе на факультете вычислительных систем ФГБОУ ВО «ТУСУР» при чтении курса лекций и проведении практических занятий по дисциплинам «Дискретная математика» и «Структуры данных», а также в рамках научно-исследовательской работы студентов для подготовки специалистов по направлениям 09.03.01 «Информатика и вычислительная техника» и 27.03.04 «Управление в технических системах».

Кроме того, результаты диссертационной работы были использованы в учебном процессе Инженерной школы информационных технологий и робототехники ФГАОУ ВО НИ ТПУ при подготовке студентов направлений 09.03.01 «Информатика и вычислительная техника» и 09.03.02 «Информационные системы и технологии». Освоение студентами предложенного алгоритмического и программного обеспечения позволило сформировать навыки составления алгоритмов, разработки программ на алгоритмических языках и тестирования работоспособности программ, построенных на дискретных структурах.

6.5 Выводы по главе

Основным результатом данной главы является разработанное программное обеспечение для вычисления коэффициентов степеней производящих функций и для генерации по рангу элементов комбинаторного множества в виде библиотек к математическим пакетам Maxima и Mathematica.

Разработана библиотека к математическому пакету Mathematica для автоматизации работы с производящими функциями и их композитами. Разработанная библиотека содержит: 150 программных функций для вычисления композит производящих функций полиномиального, рационального, тригонометрического, гиперболического, логарифмического, экспоненциального, и иррационального видов; 26 программных функций для вычисления суммы, умножения, композиции, взаимных и обратных производящих функций, а также вывода найденных коэффициентов и композит производящих функций в формате таблиц и математической выражений. Применение разработанной библиотеки позволяет решать с меньшими затратами следующие задачи: находить явные выражения композиции производящих функций, строить алгоритмы вычисления матричных представлений обратных и взаимных производящих функций, находить решения функциональных уравнений на основе уравнения Лагранжа, строить критерии простоты числа. Сравнение способов вычисления коэффициентов производящих функций для одномерного случая показало, что методы, разработанные и реализованные в данной диссертации, сопоставимы с другими методами по критерию затраченного на вычисления времени, но предполагают меньшее количество затрачиваемой оперативной памяти. Для производящих функций двух и трех переменных предлагаемые методы показывают значительно лучшие результаты по сравнению с аналогами.

Разработана библиотека к математическому пакету Mathematica для вычисления выражений полиномов. Разработанная библиотека содержит: 23 программные функции для вычисления полиномов Абеля, Бернулли, Бесселя, Чебышева, Эйлера, Эйлера-Фробениуса, Гегенбауэра, Эрмита, Гумберта, Якоби, Лагерра, Лежандра, Лерча, Малера, Мейкснера, Мотта, Наруми, Петерса, Стирлинга и Белла. Сравнение предлагаемых методов проведено на примере вычисления полиномов Белла. В качестве аналогов были взяты встроенные функции вычисления полиномов Белла в Mathematica и Maple. Результаты сравнения показали преиму-

щество предлагаемого способа над существующими по критерию затраченного на вычисления времени и оперативной памяти.

Разработано программное обеспечение для ранжирования и генерации по рангу элементов комбинаторных множеств, которое автоматизирует процессы вычислений в рамках разработанных в третьей главе алгоритмов комбинаторной генерации.

Результаты данной главы опубликованы в следующих публикациях [165; 293–299].

Заключение

В диссертационной работе решена крупная научная задача развития методов преобразования информации в данные и знания, применяющих аппарат производящих функций многих переменных.

Основные результаты диссертационной работы:

1. Изложены методы получения коэффициентов степеней производящих функций для случая одномерных, двумерных, трехмерных производящих функций и n-мерных рациональных производящих функций. Рассмотрены правила преобразования коэффициентов степеней взаимных, обратных и композиции производящих функций многих переменных, что позволяет получить их явные представления. За счет разработанного математического исчисления над коэффициентами степеней производящих функций представлен комплексный метод формирования информационных объектов, состоящий из совокупности методов и правил оперирования производящими функциями одной, двух и трех переменных, а также n-мерными рациональными производящими функциями. В качестве приложения разработанных подходов найдены явные выражения для ряда известных последовательностей онлайн-энциклопедии целочисленных последовательностей и ряда информационных объектов. Рассмотрен вопрос применения предложенного комплексного метода для формирования и описания информационных объектов на следующих примерах: специальные числа и полиномы, числовые треугольники и решеточные пути. Полученные результаты дополняют имеющиеся методы в теории производящих функций и теории полиномов на предмет эффективного определения явных представлений для соответствующих информационных объектов, а также позволяют находить связанные с ними разнообразные свойства;

2. Представлен модифицированный метод построения алгоритмов комбинаторной генерации на основе деревьев И/ИЛИ, который отличается от оригинального метода и его модификаций применением разработанного комплексного метода получения явных выражений коэффициентов производящих функций многих переменных для нахождения выражения функции мощности комбинаторного множества, в том числе определяемого несколькими параметрами. Также предложенный метод отличается применением методов приближенных вычислений и двоичного поиска для поиска выбранного сына ИЛИ-узла, что позволяет снижать вычислительную сложность алгоритмов генерации по рангу;

3. Разработаны алгоритмы комбинаторной генерации для информационных объектов, представленных комбинаторными множествами: множество сочетаний из n по m в колексикографическом порядке; множество самонепересекающихся решеточных путей на плоскости; множество помеченных путей Дика длины $2n$ с m подъемами на возвратных шагах; множество путей Дика с пиками; множество последовательностей вариантов ответа на тест с вопросами закрытого типа; множество исходов турнира на выбывание; множество частей круга, полученных при его разрезе прямыми линиями; множество правильных скобочных последовательностей, разряженных нулями; множество разбиений множества;

4. Создана база знаний производящих функций двух переменных, основанная на фреймовой модели. Данная база знаний реализована в виде автоматизированной электронной энциклопедии числовых пирамид, преимуществом которой является автоматизированный процесс поиска хранящихся записей. Полученная база знаний, содержащая 1502 производящие функции и их коэффициенты, является мощным инструментом для тестирования модулей программных систем компьютерной алгебры, выполняющих преобразования производящих функций и их коэффициентов;

5. Представлен метод построения критериев простоты числа на основе методов оперирования коэффициентами степеней производящих функций. С применением разработанного метода получены критерии простоты числа. Также на основе предложенного метода разработано специализированное программное обеспечение — генератор критериев простоты числа, и программа для анализа сгенерированных критериев. Благодаря полученным результатам определены теоретические основы для дальнейшего построения новых инструментальных средств, основанных на алгоритмах проверки чисел на простоту;

6. На основе разработанных методов и алгоритмов создано программное обеспечение для вычисления коэффициентов степеней производящих функций и для генерации по рангу элементов комбинаторного множества в виде библиотек к математическим пакетам Maxima и Mathematica. Применение разработанного программного обеспечения позволяет решать с меньшими затратами следующие задачи: находить явные выражения композиции производящих функций, строить алгоритмы вычисления матричных представлений обратных и взаимных производящих функций, находить решения функциональных уравнений на основе уравнения Лагранжа, строить критерии простоты числа, получать алгоритмы ранжирования и генерации по рангу элементов комбинаторных множеств.

Список литературы

1. *Стенли Р.* Перечислительная комбинаторика. — М.: Мир, 1990. — 440 с.
2. *Graham R. L., Knuth D. E., Patashnik O.* Concrete mathematics: A foundation for computer science. — Second edition. — USA: Addison-Wesley Publishing Company, 1994. — 657 p.
3. *Ландо С. А.* Лекции о производящих функциях. — М.: МЦНМО, 2007. — 144 с.
4. *Кнут Д. Э.* Искусство программирования. Том 1: Основные алгоритмы. — М.: Вильямс, 2002. — 720 с.
5. *Эндрюс Г.* Теория разбиений. — М.: Наука, 1982. — 256 с.
6. *Риордан Д.* Введение в комбинаторный анализ. — М.: Издательство иностранной литературы, 1963. — 288 с.
7. *Виленкин Н. Я.* Комбинаторика. — М.: Наука, 1969. — 328 с.
8. *Рыбников К. А.* Введение в комбинаторный анализ. — М.: Издательство МГУ, 1985. — 308 с.
9. *Сачков В. Н.* Введение в комбинаторные методы дискретной математики. — М.: МЦНМО, 2004. — 424 с.
10. *Егорычев Г. П.* Интегральное представление и вычисление комбинаторных сумм. — Новосибирск: Наука, Сибирское отделение, 1977. — 285 с.
11. *Кузьмин О. В.* Обобщенные пирамиды Паскаля и их приложения. — Новосибирск: Наука, Сибирская издательская фирма РАН, 2000. — 294 с.
12. *Wilf H. S.* Generatingfunctionology. — Second edition. — USA: Academic Press, 1994. — 228 p.
13. *Воронин Сергей Михайлович.* Аналитические свойства производящих функций Дирихле арифметических объектов : дис. ... д-ра физ.-мат. наук. — Москва, 1977. — 90 с.

14. *Сеге Г.* Ортогональные многочлены. — М.: Физматгиз, 1962. — 500 с.

15. *Bateman H., Erdelyi A.* Higher transcendental functions. Volume I. — USA: McGraw-Hill Book Company, 1953. — 302 p.

16. *Bateman H., Erdelyi A.* Higher transcendental functions. Volume II. — USA: McGraw-Hill Book Company, 1953. — 396 p.

17. *Bateman H., Erdelyi A.* Higher transcendental functions. Volume III. — USA: McGraw-Hill Book Company, 1955. — 292 p.

18. *Boas R. P., Buck R. C.* Polynomial expansions of analytic functions. — Second edition. — Germany: Springer, 1964. — 85 p.

19. *Геронимус Я. Л.* Теория ортогональных многочленов: Обзор достижений отечественной математики. — М.-Л.: ГИТТЛ, 1950. — 164 с.

20. *Лебедев Н. Н.* Специальные функции и их приложения. — М.: ГИТТЛ, 1953. — 380 с.

21. *Суетин П. К.* Классические ортогональные многочлены. — М.: Наука, 1979. — 418 с.

22. *Прасолов В. В.* Многочлены. — М.: МЦНМО, 2014. — 336 с.

23. *McBride E. B.* Obtaining generating functions. — Germany: Springer, 1971. — 112 p.

24. *Srivastava H. M., Manocha H. L.* A treatise on generating functions: Mathematics and its applications. — UK: Ellis Horwood, 1984. — 572 p.

25. *Rassias T. M., Srivastava H. M., Yanushauskas A.* Topics in polynomials of one and several variables and their applications. — Singapore: World Scientific, 1993. — 648 p.

26. *Srivastava H. M.* Some generalizations and basic (or q-) extensions of the Bernoulli, Euler and Genocchi polynomials // *Applied Mathematics and Information Sciences.* — 2011. — Vol. 5, no. 3. — P. 390–444.

27. *Srivastava H. M., Choi J.* Zeta and q-zeta functions and associated series and integrals. — USA: Elsevier, 2012. — 674 p.

28. *Kim T.* q-generalized Euler numbers and polynomials // *Russian Journal of Mathematical Physics*. — 2006. — Vol. 13, no. 3. — P. 293–298.

29. *Bayad A., Kim T.* Identities for the Bernoulli, the Euler and the Genocchi numbers and polynomials // *Advanced Studies in Contemporary Mathematics (Kyungshang)*. — 2010. — Vol. 20, no. 2. — P. 247–253.

30. Some theorems on Bernoulli and Euler numbers / K.-W. Hwang, D. V. Dolgy, D. S. Kim et al. // *Ars Combinatoria*. — 2013. — Vol. 109. — P. 285–297.

31. A new approach to Bell and poly-Bell numbers and polynomials / T. Kim, D. S. Kim, D. V. Dolgy et al. // *AIMS Mathematics*. — 2022. — Vol. 7, no. 3. — P. 4004–4016.

32. *Ozden H., Simsek Y., Srivastava H. M.* A unified presentation of the generating functions of the generalized Bernoulli, Euler and Genocchi polynomials // *Computers and Mathematics with Applications*. — 2010. — Vol. 60, no. 10. — P. 2779–2787.

33. *Acikgoz M., Simsek Y.* A new generating function of (q-) Bernstein-type polynomials and their interpolation function // *Abstract and Applied Analysis*. — 2010. — Vol. 2010. — Article 769095.

34. *Simsek Y.* Complete sum of products of (h,q)-extension of Euler polynomials and numbers // *Journal of Difference Equations and Applications*. — 2010. — Vol. 16, no. 11. — P. 1331–1348.

35. *Dere R., Simsek Y.* Applications of umbral algebra to some special polynomials // *Advanced Studies in Contemporary Mathematics (Kyungshang)*. — 2012. — Vol. 22, no. 3. — P. 433–438.

36. *Simsek Y.* Generating functions for generalized Stirling type numbers, array type polynomials, Eulerian type polynomials and their applications // *Fixed Point Theory and Applications*. — 2013. — Vol. 2013. — Article 87.

37. *Srivastava H. M., Kurt B., Simsek Y.* Some families of Genocchi type polynomials and their interpolation functions // *Integral Transforms and Special Functions*. — 2012. — Vol. 23, no. 12. — P. 919–938.

38. *Qi F., Guo B.-N.* Several explicit and recursive formulas for generalized Motzkin numbers // *AIMS Mathematics*. — 2020. — Vol. 5, no. 2. — P. 1333–1345.

39. Some properties of central Delannoy numbers / F. Qi, V. Cernanova, X.-T. Shi, B.-N. Guo // *Journal of Computational and Applied Mathematics*. — 2018. — Vol. 328. — P. 101–115.

40. *Boyadzhiev K. N.* Derivative polynomials for Tanh, Tan, Sech and Sec in explicit form // *Fibonacci Quarterly*. — 2007. — Vol. 45, no. 4. — P. 291–303.

41. *Cenkci M.* An explicit formula for generalized potential polynomials and its applications // *Discrete Mathematics*. — 2009. — Vol. 309, no. 6. — P. 1498–1510.

42. *Srivastava H. M., Todorov P. G.* An explicit formula for the generalized Bernoulli polynomials // *Journal of Mathematical Analysis and Applications*. — 1988. — Vol. 130, no. 2. — P. 509–513.

43. *Liu G.-D., Srivastava H. M.* Explicit formulas for the Norlund polynomials $B_n^{(x)}$ and $b_n^{(x)}$ // *Computers and Mathematics with Applications*. — 2006. — Vol. 51, no. 9–10. — P. 1377–1384.

44. *Chen G., Chen L.* Some identities involving the Fubini polynomials and Euler polynomials // *Mathematics*. — 2018. — Vol. 6, no. 12. — Article 300.

45. *Ma Y., Zhang W.* Some identities involving Fibonacci polynomials and Fibonacci numbers // *Mathematics*. — 2018. — Vol. 6, no. 12. — Article 334.

46. *Shen S., Chen L.* Some types of identities involving the Legendre polynomials // *Mathematics*. — 2019. — Vol. 7, no. 2. — Article 114.

47. *Rainville E. D.* Special functions. — USA: The Macmillan Company, 1960. — 365 p.

48. *Manocha H. L.* Generating functions for Jacobi and Laguerre polynomials // *Proceedings of the American Mathematical Society*. — 1969. — Vol. 23, no. 3. — P. 590–595.

49. *Srivastava H. M.* Some generalizations of Carlitz's theorem // *Pacific Journal of Mathematics*. — 1979. — Vol. 85, no. 2. — P. 471–477.

50. *Kilbas A. A., Srivastava H. M., Trujillo J. J.* Theory and applications of fractional differential equations. — USA: Elsevier, 2006. — 540 p.

51. *Guinand A. P.* The umbral method: A survey of elementary mnemonic and manipulative uses // *The American Mathematical Monthly*. — 1979. — Vol. 86, no. 3. — P. 187–195.

52. *Roman S.* The umbral calculus. — USA: Academic Press, 1984. — 193 p.

53. *Rota G.-C., Kahaner D., Odlyzko A.* On the foundations of combinatorial theory VII. Finite operator calculus // *Journal of Mathematical Analysis and Applications*. — 1973. — Vol. 42, no. 3. — P. 684–760.

54. *Roman S., Rota G.-C.* The umbral calculus // *Advances in Mathematics*. — 1978. — Vol. 27, no. 2. — P. 95–188.

55. *Faa di Bruno F.* Sullo sviluppo delle funzioni // *Annali di Scienze Matematiche et Fisiche di Tortoloni*. — 1855. — Vol. 6. — P. 479–480.

56. *Faa di Bruno F.* Note sur une nouvelle formule de calcul differentiel // *The Quarterly Journal of Pure and Applied Mathematics*. — 1857. — Vol. 1. — P. 359–360.

57. *Comtet L.* Advanced combinatorics: The art of finite and infinite expansions. — Netherlands: D. Reidel Publishing Company, 1974. — 354 p.

58. *Стенли Р.* Перечислительная комбинаторика: Деревья, производящие функции и симметрические функции. — М.: Мир, 2009. — 767 с.

59. *Gessel I. M.* A combinatorial proof of the multivariable lagrange inversion formula // *Journal of Combinatorial Theory, Series A*. — 1987. — Vol. 45, no. 2. — P. 178–195.

60. *Gessel I. M.* Lagrange inversion // *Journal of Combinatorial Theory, Series A*. — 2016. — Vol. 144. — P. 212–249.

61. *Labelle G.* Une nouvelle demonstration combinatoire des formules d'inversion de Lagrange // *Advances in Mathematics*. — 1981. — Vol. 42, no. 3. — P. 217–247.

62. *Gessel I. M., Labelle G.* Lagrange inversion for species // *Journal of Combinatorial Theory, Series A*. — 1995. — Vol. 72, no. 1. — P. 95–117.

63. *Bergeron F., Labelle G., Leroux P.* Combinatorial species and tree-like structures. — UK: Cambridge University Press, 1998. — 480 p.

64. *Krattenthaler C.* Operator methods and Lagrange inversion: a unified approach to Lagrange formulas // *Transactions of the American Mathematical Society.* — 1988. — Vol. 305, no. 2. — P. 431–465.

65. *Merlini D., Sprugnoli R., Verri M. C.* Lagrange inversion: When and how // *Acta Applicandae Mathematicae.* — 2006. — Vol. 94, no. 3. — P. 233–249.

66. *Stanton D.* Recent results for the q-Lagrange inversion formula // Ramanujan Revisited: Proceedings of the Centenary Conference. — USA: 1988. — P. 525–536.

67. *Drmota M.* A bivariate asymptotic expansion of coefficients of powers of generating functions // *European Journal of Combinatorics.* — 1994. — Vol. 15, no. 2. — P. 139–152.

68. *Кручинин В. В., Кручинин Д. В.* Степени производящих функций и их применение. — Томск: Издательство ТУСУР, 2013. — 236 с.

69. The Riordan group / L. W. Shapiro, S. Getu, W.-J. Woan, L. Woodson // *Discrete Applied Mathematics.* — 1991. — Vol. 34, no. 1–3. — P. 229–239.

70. *Sprugnoli R.* Riordan arrays and combinatorial sums // *Discrete Mathematics.* — 1994. — Vol. 132, no. 1–3. — P. 267–290.

71. *He T.-X., Sprugnoli R.* Sequence characterization of Riordan arrays // *Discrete Mathematics.* — 2009. — Vol. 309, no. 12. — P. 3962–3974.

72. *Barry P.* Riordan arrays: A primer. — UK: Lulu Press, 2017. — 308 p.

73. *Knuth D. E.* The art of computer programming, Volume 2: Seminumerical algorithms. — USA: Addison-Wesley Professional, 1997. — 784 p.

74. *Du Peng.* The Aztec diamond edge-probability generating function : Master's thesis in mathematics. — USA, 2011. — 41 p.

75. *Feretic S.* The column-convex polyominoes perimeter generating function for everybody // *Croatica Chemica Acta.* — 1996. — Vol. 69, no. 3. — P. 741–756.

76. *Bousquet-Melou M., Rechnitzer A.* The site-perimeter of bargraphs // *Advances in Applied Mathematics.* — 2003. — Vol. 31, no. 1. — P. 86–112.

77. *Blecher A., Brennan C., Knopfmacher A.* Combinatorial parameters in bargraphs // *Quaestiones Mathematicae.* — 2016. — Vol. 39, no. 5. — P. 619–635.

78. Dyck and Motzkin triangles with multiplicities / V. R. Meshkov, A. V. Omelchenko, M. I. Petrov, E. A. Tropp // *Moscow Mathematical Journal.* — 2010. — Vol. 10, no. 3. — P. 611–628.

79. *Gan X.-X., Bugajewski D.* A note on formal power series // *Commentationes Mathematicae Universitatis Carolinae.* — 2010. — Vol. 51, no. 4. — P. 595–604.

80. *Pemantle R., Wilson M. C.* Asymptotics of multivariate sequences I: Smooth points of the singular variety // *Journal of Combinatorial Theory, Series A.* — 2002. — Vol. 97, no. 1. — P. 129–161.

81. *Pemantle R., Wilson M. C.* Asymptotics of multivariate sequences II: Multiple points of the singular variety // *Combinatorics, Probability and Computing.* — 2004. — Vol. 13, no. 4–5. — P. 735–761.

82. *Pemantle R., Wilson M. C.* Twenty combinatorial examples of asymptotics derived from multivariate generating functions // *SIAM Review.* — 2008. — Vol. 50, no. 2. — P. 199–272.

83. Asymptotics of Multivariate Sequences [Электронный ресурс]. — URL: https://www.cs.auckland.ac.nz/~mcw/Research/mvGF/asymultseq/ (дата обращения: 01.04.2022).

84. *Bender E. A., Bruce Richmond L.* Central and local limit theorems applied to asymptotic enumeration II: Multivariate generating functions // *Journal of Combinatorial Theory, Series A.* — 1983. — Vol. 34, no. 3. — P. 255–265.

85. *Raichev A., Wilson M. C.* Asymptotics of coefficients of multivariate generating functions: improvements for smooth points // *Electronic Journal of Combinatorics.* — 2008. — Vol. 15, no. 1. — Article R89.

86. *Шишкина О. А.* Многочлены Бернулли от нескольких переменных и суммирование мономов по целым точкам рационального параллелотопа // *Известия*

Иркутского государственного университета. Серия Математика. — 2016. — Т. 16. — С. 89–101.

87. *Феллер В.* Введение в теорию вероятностей и ее приложения. Том 1. — М.: Мир, 1964. — 500 с.

88. *Dumas P., Thimonier L.* Random palindromes: multivariate generating function and Bernoulli density // *Discrete Mathematics.* — 1995. — Vol. 139, no. 1–3. — P. 143–154.

89. *Kruchinin V. V., Kruchinin D. V.* Composita and its properties // *Journal of Analysis and Number Theory.* — 2014. — Vol. 2, no. 2. — P. 37–44.

90. *Кручинин Дмитрий Владимирович.* Метод получения явных выражений полиномов на основе степеней производящих функций : дис. ... канд. физ.-мат. наук. — Красноярск, 2016. — 97 с.

91. *Gould H. W.* Combinatorial identities: A standardized set of tables listing 500 binomial coefficient summations. — USA: Morgantown Printing and Binding, 1972. — 106 p.

92. *Lehmer D. H.* Interesting series involving the central binomial coefficient // *The American Mathematical Monthly.* — 1985. — Vol. 92, no. 7. — P. 449–457.

93. *Chen H.* Interesting series associated with central binomial coefficients, Catalan numbers and harmonic numbers // *Journal of Integer Sequences.* — 2016. — Vol. 19, no. 1. — Article 16.1.5.

94. *Sloane N. J. A.* The On-Line Encyclopedia of Integer Sequences [Электронный ресурс]. — URL: http://www.oeis.org/ (дата обращения: 01.04.2022).

95. *Суетин П. К.* Ряды по многочленам Фабера. — М.: Наука, 1984. — 336 с.

96. *Curtiss J. H.* Faber polynomials and the Faber series // *The American Mathematical Monthly.* — 1971. — Vol. 78, no. 6. — P. 577—-596.

97. *Propp J.* Generalized domino-shuffling // *Theoretical Computer Science.* — 2003. — Vol. 303, no. 2–3. — P. 267–301.

98. *Simsek Y.* Special numbers and polynomials including their generating functions in umbral analysis methods // *Axioms.* — 2018. — Vol. 7, no. 2. — Article 22.

99. *Simsek Y.* New families of special numbers for computing negative order Euler numbers and related numbers and polynomials // *Applicable Analysis and Discrete Mathematics.* — 2018. — Vol. 12, no. 1. — P. 1–35.

100. *Simsek Y.* Construction method for generating functions of special numbers and polynomials arising from analysis of new operators // *Mathematical Methods in the Applied Sciences.* — 2018. — Vol. 41, no. 16. — P. 6934–6954.

101. *Kim D. S., Kim T.* On degenerate Bell numbers and polynomials // *Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas.* — 2017. — Vol. 111, no. 2. — P. 435–446.

102. *Kim T., Kim D. S., Kwon H.-I.* Some identities of Carlitz degenerate Bernoulli numbers and polynomials // *Iranian Journal of Science and Technology, Transactions A: Science.* — 2017. — Vol. 41, no. 3. — P. 749–753.

103. *Kim T., Ryoo C. S.* Some identities for Euler and Bernoulli polynomials and their zeros // *Axioms.* — 2018. — Vol. 7, no. 3. — Article 56.

104. *Ryoo C. S.* Some identities involving generalized degenerate tangent polynomials arising from differential equations // *Journal of Computational Analysis and Applications.* — 2019. — Vol. 26, no. 6. — P. 975–984.

105. *Ryoo C. S.* Some identities involving modified degenerate tangent numbers and polynomials // *Global Journal of Pure and Applied Mathematics.* — 2016. — Vol. 12, no. 3. — P. 2621–2630.

106. *Kruchinin D. V., Kruchinin V. V.* Application of a composition of generating functions for obtaining explicit formulas of polynomials // *Journal of Mathematical Analysis and Applications.* — 2013. — Vol. 404, no. 1. — P. 161–171.

107. *Carlitz L.* A degenerate staudt-clausen theorem // *Archiv der Mathematik.* — 1956. — Vol. 7, no. 1. — P. 28–33.

108. *Carlitz L.* Degenerate Stirling, Bernoulli and Eulerian numbers // *Utilitas Mathematica.* — 1979. — Vol. 15. — P. 51–88.

109. *Howard F. T.* Explicit formulas for degenerate Bernoulli numbers // *Discrete Mathematics.* — 1996. — Vol. 162, no. 1–3. — P. 175–185.

110. *Kim D. S., Kim T., Dolgy D. V.* A note on degenerate Bernoulli numbers and polynomials associated with p-adic invariant integral on Z_p // *Applied Mathematics and Computation.* — 2015. — Vol. 259. — P. 198–204.

111. *Kim D. S., Kim T.* Some identities of degenerate special polynomials // *Open Mathematics.* — 2015. — Vol. 13, no. 1. — P. 380–389.

112. *Kim D. S., Kim T., Dolgy D. V.* On q-analogs of degenerate Bernoulli polynomials // *Advances in Difference Equations.* — 2015. — Vol. 2015, no. 1. — Article 194.

113. *Kim D. S., Kim T.* Some identities of Korobov-type polynomials associated with p-adic integrals on Z_p // *Advances in Difference Equations.* — 2015. — Vol. 2015, no. 1. — Article 282.

114. *Krall H. L., Frink O.* A new class of ortogonal polynomials: The Bessel polynomials // *Transactions of the American Mathematical Society.* — 1949. — Vol. 65. — P. 100–115.

115. *Carlitz L.* A note on the Bessel polynomials // *Duke Mathematical Journal.* — 1957. — Vol. 24, no. 2. — P. 151–162.

116. *Bailey W. N.* Generalised hypergeometric series. — UK: Cambridge University Press, 1935. — Vol. 1935. — 108 p.

117. *Stein E., Weiss G.* Introduction to Fourier analysis on Euclidean spaces. — USA: Princeton University Press, 1971. — 312 p.

118. *Коробов Н. М.* Специальные полиномы и их приложения // *Математические записки.* — 1996. — Т. 2. — С. 77–89.

119. *Устинов А. В.* Полиномы Коробова и теневой анализ // *Чебышевский сборник.* — 2003. — Т. 4, № 4(8). — С. 137–152.

120. *Young P. T.* Degenerate Bernoulli polynomials, generalized factorial sums, and their applications // *Journal of Number Theory.* — 2008. — Vol. 128, no. 4. — P. 738–758.

121. *Hetyei G.* Meixner polynomials of the second kind and quantum algebras representing su(1,1) // *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences.* — 2010. — Vol. 466, no. 2117. — P. 1409–1428.

122. *Chihara T. S.* An introduction to orthogonal polynomials. — USA: Gordon and Breach, 1978. — 249 p.

123. *Barbero G. J. F., Salas J., Villaseor E. J. S.* Generalized Stirling permutations and forests: Higher-order Eulerian and Ward numbers // *Electronic Journal of Combinatorics*. — 2015. — Vol. 22, no. 3. — Article P3.37.

124. *Gessel I., Stanley R. P.* Stirling polynomials // *Journal of Combinatorial Theory, Series A*. — 1978. — Vol. 24, no. 1. — P. 24–33.

125. *Cangul I. N., Cevik A. S., Simsek Y.* Generalization of q-Apostol-type Eulerian numbers and polynomials, and their interpolation functions // *Advanced Studies in Contemporary Mathematics (Kyungshang)*. — 2015. — Vol. 25, no. 2. — P. 211–220.

126. *Carlitz L.* Some numbers related to the Stirling numbers of the first and second kind // *Publikacije Elektrotehnickog fakulteta. Serija Matematika i fizika*. — 1976. — Vol. 544/576. — P. 49–55.

127. *Boutiche M. A., Rahmani M., Srivastava H. M.* Explicit formulas associated with some families of generalized Bernoulli and Euler polynomials // *Mediterranean Journal of Mathematics*. — 2017. — Vol. 14, no. 2. — Article 89.

128. *Elezovic N.* Generalized Bernoulli polynomials and numbers, revisited // *Mediterranean Journal of Mathematics*. — 2016. — Vol. 13, no. 1. — P. 141–151.

129. *Srivastava H. M., Boutiche M. A., Rahmani M.* Some explicit formulas for the Frobenius-Euler polynomials of higher order // *Applied Mathematics and Information Sciences*. — 2017. — Vol. 11, no. 2. — P. 621–626.

130. *Agarwal A. K.* A note on generalized Sylvester polynomials // *Indian Journal of Pure and Applied Mathematics*. — 1984. — Vol. 15. — P. 431–434.

131. Some identities of Bernoulli, Euler and Abel polynomials arising from umbral calculus / D. S. Kim, T. Kim, S.-H. Lee, S.-H. Rim // *Advances in Difference Equations*. — 2013. — Vol. 2013. — Article 15.

132. *Kang J. Y., Ryoo C. S.* A research on the some properties and distribution of zeros for Stirling polynomials // *Journal of Nonlinear Science and Applications*. — 2016. — Vol. 9, no. 4. — P. 1735–1747.

133. Certain properties of Gegenbauer polynomials via Lie algebra / J. C. Prajapati, J. Choi, K. B. Kachhia, P. Agarwal // *Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas.* — 2017. — Vol. 111, no. 4. — P. 1031–1037.

134. *Kruchinin D. V., Shablya Y. V.* Explicit formulas for Meixner polynomials // *International Journal of Mathematics and Mathematical Sciences.* — 2015. — Vol. 2015. — Article 620569.

135. *Tepper M.* A factorial conjecture // *Mathematics Magazine.* — 1965. — Vol. 38, no. 5. — P. 303–304.

136. *Long C. T.* Proof of Tepper's factorial conjecture // *Mathematics Magazine.* — 1965. — Vol. 38, no. 5. — P. 304–305.

137. *Papp F. J.* Another proof of Tepper's identity // *Mathematics Magazine.* — 1972. — Vol. 45, no. 3. — P. 119–121.

138. *Gould H. W.* Euler's formula for n-th differences of powers // *The American Mathematical Monthly.* — 1978. — Vol. 85, no. 6. — P. 450–467.

139. *Bayat M., Teimoori H.* Pascal k-eliminated functional matrix and its property // *Linear Algebra and its Applications.* — 2000. — Vol. 308, no. 1–3. — P. 65–75.

140. *Bayat M., Teimoori H.* Minimal polynomial of Pascal matrices over the field Z_p // *Discrete Mathematics.* — 2001. — Vol. 232, no. 1–3. — P. 91–94.

141. *Bayat M., Teimoori H.* The linear algebra of the generalized Pascal functional matrix // *Linear Algebra and its Applications.* — 1999. — Vol. 295, no. 1–3. — P. 81–89.

142. *Zhaot X., Wang T.* The algebraic properties of the generalized Pascal functional matrices associated with the exponential families // *Linear Algebra and Its Applications.* — 2000. — Vol. 318, no. 1–3. — P. 45–52.

143. *Arponen T.* Matrix approach to polynomials 2 // *Linear Algebra and its Applications.* — 2005. — Vol. 394, no. 1–3. — P. 257–276.

144. *Yang Y., Micek C.* Generalized Pascal functional matrix and its applications // *Linear Algebra and its Applications.* — 2007. — Vol. 423, no. 2–3. — P. 230–245.

145. Some identities of ordinary and degenerate Bernoulli numbers and polynomials / D. V. Dolgy, D. S. Kim, J. Kwon, T. Kim // *Symmetry*. — 2019. — Vol. 11, no. 7. — Article 847.

146. *Gessel I. M.* Applications of the classical umbral calculus // *Algebra Universalis*. — 2003. — Vol. 49, no. 4. — P. 397–434.

147. *Dilcher K.* Bernoulli and Euler polynomials // NIST Handbook of Mathematical Functions. — UK: Cambridge University Press, 2010. — P. 587–599.

148. *Todorov P. G.* On the theory of the Bernoulli polynomials and numbers // *Journal of Mathematical Analysis and Applications*. — 1984. — Vol. 104, no. 2. — P. 309–350.

149. *Chang C.-H., Ha C.-W.* A multiplication theorem for the Lerch zeta function and explicit representations of the Bernoulli and Euler polynomials // *Journal of Mathematical Analysis and Applications*. — 2006. — Vol. 315, no. 2. — P. 758–767.

150. *Prudnikov A. P., Brychkov Y. A., Marichev O. I.* Integrals and series. Volume 3: More special functions. — USA: Gordon and Breach Science Publishers, 1990. — 23–24 p.

151. *Kim D. S., Kim T., Lee S.-H.* Umbral calculus and the Frobenius-Euler polynomials // *Abstract and Applied Analysis*. — 2013. — Vol. 2013. — Article 871512.

152. Frobenius-Euler polynomials and umbral calculus in the p-adic case / D. S. Kim, T. Kim, S.-H. Lee, S.-H. Rim // *Advances in Difference Equations*. — 2012. — Vol. 2012. — Article 222.

153. *Kruchinin D. V., Kruchinin V. V.* Explicit formula for reciprocal generating function and its application // *Advanced Studies in Contemporary Mathematics (Kyungshang)*. — 2019. — Vol. 29, no. 3. — P. 365–372.

154. A method for obtaining coefficients of compositional inverse generating functions / D. V. Kruchinin, Y. V. Shablya, V. V. Kruchinin, A. A. Shelupanov // International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2015). — AIP Conference Proceedings. — Vol. 1738. — Article 130003. — 2016.

155. *Kruchinin D., Kruchinin V., Shablya Y.* Method for obtaining coefficients of powers of bivariate generating functions // *Mathematics.* — 2021. — Vol. 9, no. 4. — Article 428.

156. *Kruchinin D., Kruchinin V.* A method for obtaining generating functions for central coefficients of triangles // *Journal of Integer Sequences.* — 2012. — Vol. 15, no. 9. — Article 12.9.3.

157. *Kruchinin D. V., Kruchinin V. V.* A generating function for the diagonal $T_{2n,n}$ in triangles // *Journal of Integer Sequences.* — 2015. — Vol. 18, no. 4. — Article 15.4.6.

158. *Kruchinin D. V.* New approach to study numeric triangles // *KnE Engineering.* — 2018. — Vol. 3, no. 5. — P. 290–297.

159. *Kruchinin D. V., Kruchinin V. V.* Explicit formulas for Some generalized polynomials // *Applied Mathematics and Information Sciences.* — 2013. — Vol. 7, no. 5. — P. 2083–2088.

160. *Kruchinin D. V., Kruchinin V. V.* A method for obtaining expressions for polynomials based on a composition of generating functions // International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2012). — AIP Conference Proceedings. — Vol. 1479. — 2012. — P. 383–386.

161. Several formulas for special values of the Bell polynomials of the second kind and applications / F. Qi, X.-T. Shi, F.-F. Liu, D. V. Kruchinin // *Journal of Applied Analysis and Computation.* — 2017. — Vol. 7, no. 3. — P. 857–871.

162. *Kruchinin D. V.* Explicit formulas for Korobov polynomials // *Proceedings of the Jangjeon Mathematical Society.* — 2017. — Vol. 20, no. 1. — P. 43–50.

163. *Kruchinin D. V.* Explicit formula for the generalized Mott polynomials // *Advanced Studies in Contemporary Mathematics (Kyungshang).* — 2014. — Vol. 24, no. 3. — P. 327–332.

164. Explicit formulas for the Eulerian numbers of the second kind / D. V. Kruchinin, V. V. Kruchinin, Y. V. Shablya, A. A. Shelupanov // International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2018). — AIP Conference Proceedings. — Vol. 2116. — Article 100008. — 2019.

165. Метод получения явных выражений полиномов на основе степеней производящих функций и его реализация / Д. В. Кручинин, В. В. Кручинин, А. А. Шелупанов и др. — Томск: В-Спектр, 2017. — 172 с.

166. *Kruchinin D. V., Kruchinin V. V.* About some properties of polynomials defined by generating functions of form $F(t,x)^{\alpha} \cdot G(t,\alpha)^{x}$ // International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2016). — AIP Conference Proceedings. — Vol. 1863. — Article 300015. — 2017.

167. *Kruchinin D., Kruchinin V., Shablya Y.* Obtaining explicit formulas and identities for polynomials defined by generating functions of the form $F(t)^{x} \cdot G(t)^{\alpha}$ // Polynomials – Theory and Application. — UK: IntechOpen, 2019.

168. *Kruchinin D., Kruchinin V., Simsek Y.* Generalized Tepper's identity and its application // *Mathematics*. — 2020. — Vol. 8, no. 2. — Article 243.

169. Explicit formulas for enumeration of lattice paths: Basketball and the kernel method / C. Banderier, C. Krattenthaler, A. Krinik et al. // Lattice Path Combinatorics and Applications. — Springer, 2019. — P. 78–118.

170. *Кручинин В. В.* Методы построения алгоритмов генерации и нумерации комбинаторных объектов на основе деревьев И/ИЛИ. — Томск: В-Спектр, 2007. — 200 с.

171. *Кнут Д. Э.* Искусство программирования. Том 4А: Комбинаторные алгоритмы, часть 1. — М.: Вильямс, 2018. — 960 с.

172. *Ruskey F.* Combinatorial generation. Working version (1j-CSC 425/520) [Электронный ресурс]. — 2003. — URL: https://page.math.tu-berlin.de/~felsner/SemWS17-18/Ruskey-Comb-Gen.pdf (дата обращения: 01.04.2022).

173. *Reingold E. M., Nievergelt J., Deo N.* Combinatorial algorithms: Theory and practice. — USA: Prentice-Hall, 1977. — 433 p.

174. *Kreher D. L., Stinson D. R.* Combinatorial algorithms: Generation, enumeration, and search. — USA: CRC Press, 1999. — 344 p.

175. ECO: A methodology for the enumeration of combinatorial objects / E. Barcucci, A. Del Lungo, E. Pergola, R. Pinzani // *Journal of Difference Equations and Applications*. — 1999. — Vol. 5, no. 4–5. — P. 435–490.

176. *Barcucci E., Del Lungo A., Pergola E.* Random generation of trees and other combinatorial objects // *Theoretical Computer Science.* — 1999. — Vol. 218, no. 2. — P. 219–232.

177. Exhaustive generation of combinatorial objects by ECO / S. Bacchelli, E. Barcucci, E. Grazzini, E. Pergola // *Acta Informatica.* — 2004. — Vol. 40, no. 8. — P. 585–602.

178. Mixed succession rules: The commutative case / S. Bacchelli, L. Ferrari, R. Pinzani, R. Sprugnoli // *Journal of Combinatorial Theory, Series A.* — 2010. — Vol. 117, no. 5. — P. 568–582.

179. *Del Lungo A., Frosini A., Rinaldi S.* ECO method and the exhaustive generation of convex polyominoes // *Lecture Notes in Computer Science: Discrete Mathematics and Theoretical Computer Science.* — 2003. — Vol. 2731. — P. 129–140.

180. On the generation and enumeration of some classes of convex polyominoes / A. Del Lungo, E. Duchi, A. Frosini, S. Rinaldi // *Electronic Journal of Combinatorics.* — 2004. — Vol. 11, no. 1. — Article R60.

181. *Vajnovszki V.* Generating involutions, derangements, and relatives by ECO // *Discrete Mathematics and Theoretical Computer Science.* — 2010. — Vol. 12, no. 1. — P. 109–122.

182. *Vajnovszki V.* An efficient Gray code algorithm for generating all permutations with a given major index // *Journal of Discrete Algorithms.* — 2014. — Vol. 26. — P. 77–88.

183. *Do P. T., Tran T. T. H., Vajnovszki V.* Exhaustive generation for permutations avoiding (colored) regular sets of patterns // *Discrete Applied Mathematics.* — 2019. — Vol. 268. — P. 44–53.

184. *Flajolet P., Zimmerman P., Cutsem B.* A calculus for the random generation of combinatorial structures // *Theoretical Computer Science.* — 1994. — Vol. 132, no. 1–2. — P. 1–35.

185. *Sedgewick R., Flajolet P.* An introduction to the analysis of algorithms. — Second edition. — USA: Addison-Wesley Professional, 2013. — 592 p.

186. *Martinez C., Molinero X.* A generic approach for the unranking of labeled combinatorial classes // *Random Structures and Algorithms.* — 2001. — Vol. 19, no. 3–4. — P. 472–497.

187. *Martinez C., Molinero X.* Generic algorithms for the generation of combinatorial objects // *Lecture Notes in Computer Science: Mathematical Foundations of Computer Science.* — 2003. — Vol. 2747. — P. 572–581.

188. *Martinez C., Molinero X.* An experimental study of unranking algorithms // *Lecture Notes in Computer Science: Experimental and Efficient Algorithms.* — 2004. — Vol. 3059. — P. 326–340.

189. *Martinez C., Molinero X.* Efficient iteration in admissible combinatorial classes // *Theoretical Computer Science.* — 2005. — Vol. 346, no. 2–3. — P. 388–417.

190. *Molinero X., Vives J.* Unranking algorithms for combinatorial structures // *International Journal of Applied Mathematics and Informatics.* — 2015. — Vol. 9. — P. 110–115.

191. *Ryabko B. Y.* Fast enumeration of combinatorial objects // *Discrete Mathematics and Applications.* — 1998. — Vol. 8, no. 2. — P. 163–182.

192. *Medvedeva Y. S., Ryabko B. Y.* Fast enumeration algorithm for words with given constraints on run lengths of ones // *Problems of Information Transmission.* — 2010. — Vol. 46, no. 4. — P. 390–399.

193. *Medvedeva Y.* Fast enumeration of words generated by Dyck grammars // *Mathematical Notes.* — 2014. — Vol. 96, no. 1–2. — P. 68–83.

194. *Кручинин Владимир Викторович.* Методы, алгоритмы и программное обеспечение комбинаторной генерации : дис. ... д-ра техн. наук. — Томск, 2010. — 393 с.

195. *Шабля Юрий Васильевич.* Алгоритмическое обеспечение комбинаторной генерации на основе применения теории производящих функций : дис. ... канд. техн. наук. — Томск, 2019. — 123 с.

196. *Shablya Y., Kruchinin D., Kruchinin V.* Application of the method of compositae in combinatorial generation // Mediterranean International Conference of Pure

and Applied Mathematics and Related Areas (MICOPAM 2019). — 2019. — P. 91–94.

197. *Hartman P., Sawada J.* Ranking and unranking fixed-density necklaces and Lyndon words // *Theoretical Computer Science.* — 2019. — Vol. 791. — P. 36–47.

198. Amortized efficiency of generation, ranking and unranking left-child sequences in lexicographic order / K.-J. Pai, J.-M. Chang, R.-Y. Wu, S.-C. Chang // *Discrete Applied Mathematics.* — 2019. — Vol. 268. — P. 223–236.

199. *Amani M., Nowzari-Dalini A.* Efficient generation, ranking, and unranking of (k,m)-ary trees in B-order // *Bulletin of the Iranian Mathematical Society.* — 2019. — Vol. 45, no. 4. — P. 1145–1158.

200. *Akl S. G.* A comparison of combination generation methods // *ACM Transactions on Mathematical Software (TOMS).* — 1981. — Vol. 7, no. 1. — P. 42–45.

201. *Mifsud C. J.* Algorithm 154: Combination in lexicographical order // *Communications of the ACM.* — 1963. — Vol. 6, no. 3. — P. 103.

202. *Chan B., Akl S. G.* Generating combinations in parallel // *BIT.* — 1986. — Vol. 26, no. 1. — P. 1–6.

203. *Chen G. H., Chern M.-S.* Parallel generation of permutations and combinations // *BIT.* — 1986. — Vol. 26, no. 3. — P. 277–283.

204. *Akl S. G., Gries D., Stojmenovic I.* An optimal parallel algorithm for generating combinations // *Information Processing Letters.* — 1989. — Vol. 33, no. 3. — P. 135–139.

205. *Lin C.-J.* A parallel algorithm for generating combinations // *Computers and Mathematics with Applications.* — 1989. — Vol. 17, no. 12. — P. 1523–1533.

206. *Tsay J. C., Lin C. J.* A systolic design for generating combinations in lexicographic order // *Parallel Computing.* — 1990. — Vol. 13, no. 1. — P. 119–125.

207. *Stojmenovic I.* A simple systolic algorithm for generating combinations in lexicographic order // *Computers and Mathematics with Applications.* — 1992. — Vol. 24, no. 4. — P. 61–64.

208. *Elhage H, Stojmenovic I.* Systolic generation of combinations from arbitrary elements // *Parallel Processing Letters.* — 1992. — Vol. 2, no. 2–3. — P. 241–248.

209. *Kapralski A.* New methods for the generation of permutations, combinations, and other combinatorial objects in parallel // *Journal of Parallel and Distributed Computing.* — 1993. — Vol. 17, no. 4. — P. 315–326.

210. *Xu C.-W., Ma X, Shiue W.-K.* A new parallel combination generator // International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 1996). — 1996. — P. 25–28.

211. *Kokosinski Z.* On parallel generation of combinations in associative processor architectures // IASTED International Conference on Parallel and Distributed Systems (Euro-PDS 1997). — 1997. — P. 283–289.

212. *Itai A.* Generating permutations and combinations in lexicographical order // *Journal of the Brazilian Computer Society.* — 2001. — Vol. 7, no. 3. — P. 65–68.

213. *Eades P., McKay B.* An algorithm for generating subsets of fixed size with a strong minimal change property // *Information Processing Letters.* — 1984. — Vol. 19, no. 3. — P. 131–133.

214. *Xiang L., Ushijima K.* On $O(1)$ time algorithms for combinatorial generation // *Computer Journal.* — 2001. — Vol. 44, no. 4. — P. 292–302.

215. *Torres-Jimenez J., Izquierdo-Marquez I.* A low spatial complexity algorithm to generate combinations with the strong minimal change property // *Discrete Mathematics, Algorithms and Applications.* — 2019. — Vol. 11, no. 5. — Article 1950060.

216. *Ruskey F., Williams A.* The coolest way to generate combinations // *Discrete Mathematics.* — 2009. — Vol. 309, no. 17. — P. 5305–5320.

217. *Knott G. D.* A numbering system for combinations // *Communications of the ACM.* — 1974. — Vol. 17, no. 1. — P. 45–46.

218. *Er M. C.* Lexicographic ordering, ranking and unranking of combinations // *International Journal of Computer Mathematics.* — 1985. — Vol. 17, no. 3–4. — P. 277–283.

219. *Kokosinski Z.* Algorithms for unranking combinations and their applications // IASTED/ISMM International Conference on Parallel and Distributed Computing and Systems (PDPCS 1995). — 1995. — P. 216–224.

220. *Kokosinski Z.* Unranking combinations in parallel // International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 1996). — 1996. — P. 79–82.

221. *Shablya Y., Kruchinin D.* Euler–Catalan's number triangle and its application // *Symmetry*. — 2020. — Vol. 12, no. 4. — Article 600.

222. *Genitrini A., Pepin M.* Lexicographic unranking of combinations revisited // *Algorithms*. — 2021. — Vol. 14, no. 3. — Article 97.

223. *Shimizu T., Fukunaga T., Nagamochi H.* Unranking of small combinations from large sets // *Journal of Discrete Algorithms*. — 2014. — Vol. 29. — P. 8–20.

224. *Parque V., Miyashita T.* Towards the succinct representation of m out of n // *Lecture Notes in Computer Science: Internet and Distributed Computing Systems*. — 2018. — Vol. 11226. — P. 16–26.

225. *Parque V., Miyashita T.* Unranking combinations using gradient-based optimization // International Conference on Tools with Artificial Intelligence (ICTAI 2018). — 2018. — P. 579–586.

226. *Roberts A. W., Varberg D. E.* Convex functions. — USA: Academic Press, 1973. — 300 p.

227. *Arfken G. B., Weber H. J., Harris F. E.* Mathematical methods for physicists: A comprehensive guide. — USA: Academic Press, 2012. — 1220 p.

228. *Gao S., Chen K.-H.* Tackling sequences from prudent self-avoiding walks // International Conference on Foundations of ComputerScience (FCS 2014). — 2014.

229. *Deutsch E.* Dyck path enumeration // *Discrete Mathematics*. — 1999. — Vol. 204, no. 1–3. — P. 167–202.

230. *Petersen T. K.* Eulerian numbers. — USA: Birkhauser, 2015. — 474 p.

231. *Connor Desai S., Reimers S.* Comparing the use of open and closed questions for web-based measures of the continued-influence effect // *Behavior Research Methods*. — 2019. — Vol. 51, no. 3. — P. 1426–1440.

232. Open-ended vs. close-ended questions in web questionnaires / U. Reja, K. L. Manfreda, V. Hlebec, V. Vehovar // *Developments in Applied Statistics*. — 2003. — Vol. 19. — P. 159–177.

233. *Banks R. B.* Slicing pizzas, racing turtles and further adventures in applied mathematics. — USA: Princeton University Press, 1999. — 290 p.

234. *England M., Bradford R., Davenport J. H.* Cylindrical algebraic decomposition with equational constraints // *Journal of Symbolic Computation*. — 2020. — Vol. 100. — P. 38–71.

235. *Шабля Ю. В., Кручинин Д. В.* Модификация метода построения алгоритмов комбинаторной генерации на основе применения теории производящих функций // *Доклады ТУСУР*. — 2019. — Т. 22, № 3. — С. 55–60.

236. *Shablya Y., Kruchinin D., Kruchinin V.* Method for developing combinatorial generation algorithms based on AND/OR trees and its application // *Mathematics*. — 2020. — Vol. 8, no. 6. — Article 962.

237. Algorithms for ranking and unranking the combinatorial set of closed questionnaire answers / P. P. Shcheglov, G. A. Filippov, Y. V. Shablya, D. V. Kruchinin // International Conference on Prospects of Fundamental Sciences Development. — Journal of Physics: Conference Series. — Vol. 1611. — Article 012069. — 2020.

238. *Kruchinin D., Kruchinin V., Shablya Y.* On some properties of generalized Narayana numbers // *Quaestiones Mathematicae*. — 2021.

239. *Кручинин Д. В.* Модификация метода построения алгоритмов комбинаторной генерации на основе применения производящих функций многих переменных и приближенных вычислений // *Доклады ТУСУР*. — 2022. — Т. 25, № 1. — С. 55–60.

240. Unranking small combinations of a large set in co-lexicographic order / V. Kruchinin, Y. Shablya, D. Kruchinin, V. Rulevskiy // *Algorithms*. — 2022. — Vol. 15, no. 2. — Article 36.

241. World!Of Numbers [Электронный ресурс]. — URL: http://www.worldofnumbers.com/ (дата обращения: 01.04.2022).

242. The Combinatorial Object Server [Электронный ресурс]. — URL: http://combos.org/ (дата обращения: 01.04.2022).

243. *Sloane N. J. A.* The on-line encyclopedia of integer sequences // *Notices of the American Mathematical Society.* — 2018. — Vol. 65, no. 9. — P. 1062–1074.

244. *Kimberling C.* Path-counting and Fibonacci numbers // *Fibonacci Quarterly.* — 2002. — Vol. 40, no. 4. — P. 328–338.

245. *Кручинин Д. В.* База знаний коэффициентов k-степени производящих функций двух переменных // *Доклады ТУСУР.* — 2021. — Т. 24, № 4. — С. 85–89.

246. *Кручинин Д. В.* Методика использования базы знаний производящих функций двух переменных // *Системы анализа и обработки данных.* — 2022. — Т. 85, № 1. — С. 121–139.

247. *Yan S. Y.* Primality testing and integer factorization in public-key cryptography. — USA: Springer, 2009. — 389 p.

248. *Goel N., Gupta I., Dass B. K.* Zero knowledge undeniable signature scheme over semigroup action problem // *Italian Journal of Pure and Applied Mathematics.* — 2017. — Vol. 38. — P. 45–53.

249. *Somsuk K.* The improvement of initial value closer to the target for Fermat's factorization algorithm // *Journal of Discrete Mathematical Sciences and Cryptography.* — 2018. — Vol. 21, no. 7–8. — P. 1573–1580.

250. *Балабанов А. А., Агафонов А. Ф., Рыку В. А.* Алгоритм быстрой генерации ключей в криптографической системе RSA // *Вестник научно-технического развития.* — 2009. — Т. 23, № 7. — С. 11–17.

251. *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии. — М.: МНЦМО, 2003. — 328 с.

252. *Черемушкин А. В.* Лекции по арифметическим алгоритмам в криптографии. — М.: МНЦМО, 2002. — 104 с.

253. *Ribenboim P.* The little book of bigger primes. — Second edition. — USA: Springer, 2004. — Vol. 2004. — 379 p.

254. *Mestrovic R.* A primality criterion based on Lucas' congruence // *International Journal of Number Theory.* — 2016. — Vol. 12, no. 5. — P. 1365–1369.

255. *Agrawal M.* Primality tests based on Fermat's little theorem // *Lecture Notes in Computer Science: Distributed Computing and Networking.* — 2006. — Vol. 4308. — P. 288–293.

256. *Robert A. M.* A course in p-adic analysis. — USA: Springer, 2000. — 454 p.

257. Contrast various tests for primality / V. Kochar, D. P. Goswami, M. Agarwal, S. Nandi // International Conference on Accessibility to Digital World (ICADW 2016). — 2017. — P. 39–44.

258. *Koshy T.* Fibonacci and Lucas numbers with applications. — USA: John Wiley & Sons, 2001. — 648 p.

259. *Kocer E. G., Tuglu N.* The Binet formulas for the Pell and Pell–Lucas p-numbers // *Ars Combinatoria.* — 2007. — Vol. 85. — P. 3–17.

260. Powers in products of terms of Pell's and Pell–Lucas sequences / J. J. Bravo, P. Das, S. Guzman, S. Laishram // *International Journal of Number Theory.* — 2015. — Vol. 11, no. 4. — P. 1259–1274.

261. New families of Jacobsthal and Jacobsthal–Lucas numbers / P. Catarino, P. Vasco, H. Campos et al. // *Algebra and Discrete Mathematics.* — 2015. — Vol. 20, no. 1. — P. 40–54.

262. On a divisibility relation for Lucas sequences / Y. F. Bilu, T. Komatsu, F. Luca et al. // *Journal of Number Theory.* — 2016. — Vol. 163. — P. 1–18.

263. *Li C., Wenpeng Z.* Chebyshev polynomials and their some interesting applications // *Advances in Difference Equations.* — 2017. — Vol. 2017, no. 1. — Article 303.

264. Some identities of Chebyshev polynomials arising from non-linear differential equations / T. Kim, D. S. Kim, J.-J. Seo, D. V. Dolgy // *Journal of Computational Analysis and Applications.* — 2017. — Vol. 23, no. 5. — P. 820–832.

265. *Foata D.* Eulerian polynomials: From Euler's time to the present // The legacy of Alladi Ramakrishnan in the mathematical sciences. — USA: Springer, 2010. — P. 253–273.

266. *Araci S., Sen E., Acikgoz M.* A class of generating functions for a new generalization of Eulerian polynomials with their interpolation functions // *Filomat.* — 2016. — Vol. 30, no. 8. — P. 2269–2275.

267. *Touchard J.* Proprietes arithmetiques de certains nombres recurrents // *Annales de la Societe Scientifique de Bruxelles.* — 1933. — Vol. 53. — P. 21–31.

268. *Klazar M.* Bell numbers, their relatives, and algebraic differential equations // *Journal of Combinatorial Theory, Series A.* — 2003. — Vol. 102, no. 1. — P. 63–87.

269. *Rowland E. S.* A natural prime-generating recurrence // *Journal of Integer Sequences.* — 2008. — Vol. 11, no. 2. — Article 08.2.8. — P. 1–13.

270. *Кручинин Д. В., Кручинин В. В.* Метод построения алгоритмов проверки простоты натуральных чисел для задач защиты информации // *Доклады ТУСУР.* — 2011. — Т. 24, № 2. — С. 247–251.

271. *Кручинин Д. В.* О свойствах коэффициентов суперпозиции некоторых производящих функций // *Прикладная дискретная математика.* — 2012. — Т. 15, № 1. — С. 55–59.

272. *Shablya Y. V., Kruchinin D. V., Shelupanov A. A.* New properties of a composition of ordinary generating functions for primes // *Journal of Discrete Mathematical Sciences and Cryptography.* — 2021. — Vol. 24, no. 4. — P. 917–930.

273. Properties of a composition of exponential and ordinary generating functions / D. V. Kruchinin, Y. V. Shablya, V. V. Kruchinin, A. A. Shelupanov // *Communications in Mathematics and Applications.* — 2018. — Vol. 9, no. 4. — P. 705–711.

274. Integer properties of a composition of exponential generating functions / D. V. Kruchinin, Y. V. Shablya, O. O. Evsutin, A. A. Shelupanov // International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2016). — AIP Conference Proceedings. — Vol. 1863. — Article 300014. — 2017.

275. *Кручинин Д.В.* Метод построения рекуррентных вероятностных генераторов простых чисел // *Доклады ТУСУР*. — 2012. — Vol. 25, no. 2. — P. 131–135.

276. *Кручинин Д. В., Шабля Ю. В.* Программное обеспечение для анализа тестов простоты натурального числа // *Доклады ТУСУР*. — 2014. — Т. 34. — С. 95–99.

277. *Шабля Ю. В., Кручинин Д. В., Шелупанов А. А.* Генератор критериев простоты натурального числа на основе свойств композиции производящих функций // *Доклады ТУСУР*. — 2015. — Т. 38. — С. 97–101.

278. Актуальные направления развития методов и средств защиты информации / А. А. Шелупанов, О. О. Евсютин, А. А. Конев и др. // *Доклады ТУСУР*. — 2017. — Т. 20, № 3. — С. 11–24.

279. Information security methods – Modern research directions / A. Shelupanov, O. Evsyutin, A. Konev et al. // *Symmetry*. — 2019. — Vol. 11, no. 2. — Article 150.

280. *Перминова М. Ю.* Программный модуль получения явных выражений коэффициентов производящих функций, основанных на использовании композиции // *Доклады ТУСУР*. — 2017. — Т. 20, № 1. — С. 65–70.

281. *Petkovsek Marko*. Finding closed-form solutions of difference equations by symbolic methods : PhD thesis in mathematics. — USA, 1991.

282. Wolfram Language & System. Documentation Center [Электронный ресурс]. — URL: https://reference.wolfram.com/language/ (дата обращения: 01.04.2022).

283. Help – Maplesoft [Электронный ресурс]. — URL: https://www.maplesoft.com/support/help/ (дата обращения: 01.04.2022).

284. Maxima Manual [Электронный ресурс]. — URL: https://maxima.sourceforge.io/docs/manual/maxima.pdf (дата обращения: 01.04.2022).

285. Help Center for MATLAB, Simulink and other MathWorks products [Электронный ресурс]. — URL: https://www.mathworks.com/help/ (дата обращения: 01.04.2022).

286. Алгоритмы: Построение и анализ / Т. Х. Кормен, Ч. И. Лейзерсон, Р. Л. Ривест, К. Штайн. — Третье изд. — М.: Вильямс, 2013. — 1328 с.

287. *Bell E. T.* Partition polynomials // *Annals of Mathematics.* — 1927. — Vol. 2, no. 1/4. — P. 38 – 46.

288. *Wheeler F. S.* Bell polynomials // *ACM SIGSAM Bulletin.* — 1987. — Vol. 21, no. 3. — P. 44–53.

289. *Natalini P., Paolo P. E.* Remarks on Bell and higher order Bell polynomials and numbers // *Cogent Mathematics.* — 2016. — Vol. 3, no. 1. — Article 1220670.

290. *Krivoruchenko M. I.* Trace identities for skew-symmetric matrices // *Mathematics and Computer Science.* — 2016. — Vol. 1, no. 2. — P. 21–28.

291. *Мельникова В. А.* Алгоритм аналитического дифференцированиякомбинаторных полиномов разбиений // *Системы. Методы. Технологии.* — 2013. — Т. 319, № 3. — С. 112–116.

292. НИИ АЭМ ТУСУР [Электронный ресурс]. — URL: http://niiaem.tomsk.ru/ (дата обращения: 01.04.2022).

293. A library for calculating polynomials based on compositae of generating functions / D. V. Kruchinin, V. S. Melman, Y. V. Shablya, A. A. Shelupanov // International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2018). — AIP Conference Proceedings. — Vol. 2116. — Article 100007. — 2019.

294. *Перминова М. Ю., Кручинин В. В., Кручинин Д. В.* Алгоритм декомпозиции полиномов, основанный на разбиениях // *Доклады ТУСУР.* — 2015. — Т. 38. — С. 102–107.

295. *Kruchinin D. V.* On solving some functional equations // *Advances in Difference Equations.* — 2015. — Vol. 2015. — Article 17.

296. *Kruchinin D. V., Perminova M. Y.* About solving some functional equations related to the Lagrange inversion theorem // *Montes Taurus Journal of Pure and Applied Mathematics.* — 2021. — Vol. 3, no. 1. — P. 62–69.

297. Сравнительный анализ вычислительных способов нахождения коэффициентов ряда Тейлора в математических пакетах / В. С. Мельман, Ю. В. Шабля,

Д. В. Кручинин, В. В. Кручинин // *Доклады ТУСУР*. — 2017. — Т. 20, № 4. — С. 71–74.

298. Realization of a method for calculating Bell polynomials based on compositae of generating functions / V. S. Melman, Y. V. Shablya, D. V. Kruchinin, A. A. Shelupanov // *Journal of Informatics and Mathematical Sciences*. — 2018. — Vol. 10, no. 4. — P. 659–672.

299. Personalized distance learning using the STACK system / D. Kruchinin, Y. Shablya, A. Shelupanov, V. Melman // International Research Conference "Information technologies in Science, Management, Social sphere and Medicine" (ITSMSSM 2017). — Vol. 72. — 2017. — P. 18–20.

Приложение А

Свидетельства о государственной регистрации программ для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

RU2021669954

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2021669954
Дата регистрации: **06.12.2021**
Номер и дата поступления заявки:
2021669226 29.11.2021
Дата публикации и номер бюллетеня:
06.12.2021 Бюл. № 12

Автор(ы):
Кручинин Дмитрий Владимирович (RU)
Правообладатель(и):
Кручинин Дмитрий Владимирович (RU)

Название программы для ЭВМ:
Программа для генерации и оценки математических заданий

Реферат:
Программа предназначена для создания математических заданий путем генерации внутренних параметров и для формирования обратной связи по анализу ответа обучающегося. Программа представляет собой программный модуль на языке системы компьютерной алгебры Maxima в плагине Stack системы Moodle. Программа может использоваться для формирования комплекса адаптивных тренажеров с обратной связью для обучения школьной математике. В программе реализованы генераторы заданий по следующим разделам математики: графики, экстремумы, уравнения, неравенства, задачи теории вероятностей, логические задачи.

Язык программирования: Maxima, XML

Объем программы для ЭВМ: 3087 КБ

Стр.: 1

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2018611242

Программа «PCG: Primality Criterion Generator» для генерации критериев простоты числа

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Томский государственный университет систем управления и радиоэлектроники» (ТУСУР) (RU)***

Авторы: ***Шабля Юрий Васильевич (RU), Кручинин Дмитрий Владимирович (RU), Мельман Вадим Сергеевич (RU)***

Заявка № **2017662540**
Дата поступления **04 декабря 2017 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***26 января 2018 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2018611241

Программа «PTA: Primality Test Analyser» для анализа тестов простоты числа

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Томский государственный университет систем управления и радиоэлектроники» (ТУСУР) (RU)***

Авторы: ***Шабля Юрий Васильевич (RU), Кручинин Дмитрий Владимирович (RU), Мельман Вадим Сергеевич (RU)***

Заявка № **2017662547**
Дата поступления **04 декабря 2017 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***26 января 2018 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2019660020

Программа для ранжирования и генерации по рангу элементов комбинаторных множеств

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Томский государственный университет систем управления и радиоэлектроники» (ТУСУР) (RU)***

Авторы: ***Шабля Юрий Васильевич (RU),***
Кручинин Дмитрий Владимирович (RU)

Заявка № **2019618795**
Дата поступления **17 июля 2019 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***29 июля 2019 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

Приложение Б

Акты внедрения

УТВЕРЖДАЮ
Директор-Главный конструктор
«НИИ АЭМ ТУСУР»
_____/А.Г. Юдинцев/
«15» 04 2022 г.

АКТ О ВНЕДРЕНИИ
результатов диссертационной работы
Кручинина Дмитрия Владимировича

Настоящий акт подтверждает, что результаты диссертационной работы использованы в проектно-конструкторской деятельности НИИ Автоматики и электромеханики ТУСУР, при разработке программного обеспечения для оборудования контроля и испытаний при экспериментальной наземной отработке литий-ионных аккумуляторных батарей космических аппаратов.

Надежность и эксплуатационные качества разработанного оборудования с программным обеспечением, с использованием алгоритмов комбинаторной генерации, проверены при наземно-технических испытаниях литий-ионных аккумуляторных батарей космических аппаратов. Проверка показала, что разработанное программное обеспечение, полностью соответствует техническому заданию и условиям его работы.

Использование указанных результатов диссертационной работы позволяет повысить качество контрольно-испытательного оборудования аккумуляторных батарей, повысить уровень автоматизации при проведении наземной подготовки бортовых аккумуляторных батарей к штатной эксплуатации.

Результаты внедрялись при выполнении договора
№ 1420187309511010128000871/777-31/17 СЧ ОКР «Разработка зарядно-разрядного программно-аппаратного комплекса (ЗРПАК)».

Заведующий отделом №16 Кремзуков Ю.А.

Акционерное общество
«ИНФОРМАЦИОННЫЕ СПУТНИКОВЫЕ СИСТЕМЫ»
имени академика М.Ф. Решетнёва»

РЕШЕТНЕВ
АО «ИСС»

ул. Ленина, д. 52, г. Железногорск, ЗАТО Железногорск, Красноярский край, Российская Федерация, 662972
Тел. (3919) 76-40-02, 72-24-39, Факс (3919) 72-26-35, 75-61-46, e-mail: office@iss-reshetnev.ru, http: //www.iss-reshetnev.ru
ОГРН 1082452000290, ИНН 2452034898

УТВЕРЖДАЮ
Заместитель генерального директора по науке
К.Г. Охоткин
«14» 04 2022 г

АКТ
об использовании результатов диссертационной работы
Кручинина Дмитрия Владимировича

Настоящий акт подтверждает, что результаты диссертационной работы Д.В. Кручинина использованы в проектно-конструкторской деятельности при разработке программного обеспечения для оборудования контроля и имитации элементов литий-ионных аккумуляторных батарей космических аппаратов.

Надежность и эксплуатационные качества разработанного оборудования с программным обеспечением, с использованием алгоритмов комбинаторной генерации, проверены при наземно-технических испытаниях литий-ионных аккумуляторных батарей космических аппаратов. Проверка показала, что разработанное программное обеспечение, полностью соответствует техническому заданию и условиям его работы.

Использование указанных результатов диссертационной работы позволяет повысить качество контрольно-испытательного и имитационного оборудования литий-ионных аккумуляторных батарей, повысить уровень автоматизации при проведении наземной подготовки бортовых устройств к штатной эксплуатации.

Результаты внедрялись при выполнении договора № 14/БИАБ-200ЛИ/13 СЧ ОКР по теме «Разработка блока имитации элементов литий-ионной аккумуляторной батареи, изготовление и поставка".

Начальник отдела
бортовых систем электропитания Нестеринин М.В.

ПЛАНТА плюс
Биотехнологическое предприятие

Общество с ограниченной ответственностью «ПлантаПлюс»
Фрунзепр., 20, оф. 411, 413, г. Томск, 634029, тел.: (3822) 202-334, http://www.planta-plus.ru
ИНН / КПП: 7017102395 / 701701001, ОГРН: 1047000204096, р/с: 40702810506290004355, к/с: 30101810500000000728,
ПАО «Томскпромстройбанк» г. Томск, БИК: 046902728

УТВЕРЖДАЮ
Директор ООО «ПлантаПлюс»
_____ И. М. Галактионов
«25» марта 2022 г.

АКТ
о внедрении результатов диссертационной работы
Кручинина Дмитрия Владимировича

Комиссия в составе:

председатель комиссии – Кулятов Дмитрий Васильевич, начальник лаборатории промышленной микробиологии ООО «ПлантаПлюс»;

члены комиссии:

- Николаева Дарья Леонидовна, научный сотрудник лаборатории промышленной микробиологии ООО «ПлантаПлюс»;

- Кузьминская Елена Анатольевна, научный сотрудник химической лаборатории ООО «ПлантаПлюс».

составили настоящий акт о том, что результаты диссертационной работы Кручинина Д.В. внедрены в деятельность ООО «Планта-Плюс» в процессе работы над улучшением информационной системы хранения и обработки экспериментальных данных, связанных с исследованием и производством микробиологических препаратов для растениеводства.

Модифицированный метод построения алгоритмов комбинаторной генерации был применен для разработки алгоритмов кодирования и декодирования микробиологических препаратов, представленных деревьями

И/ИЛИ.

Применение предложенных в диссертационной работе Кручинина Д.В. подходов позволило сократить объем базы данных на 9% за счет уменьшения количества дублируемой информации и повысить скорость поиска и обработки данных на 5%.

Председатель комиссии Кулятов Д.В.

Члены комиссии: Николаева Д.Л.

Кузьминская Е.А.

Общество с ограниченной ответственностью
«Эль Контент»

ОГРН 1107017014565 ИНН 7017267534 КПП 701701001
Адрес: г. Томск, ул. Киевская 57-27, Телефон: +7-903-953-5530

АКТ

о внедрении результатов диссертационной работы
Кручинина Дмитрия Владимировича

Настоящий Акт подтверждает, что результаты диссертационной работы внедрены в деятельность ООО «Эль Контент» при решении задач тестирования разработанного программного обеспечения систем обучения.

Программное обеспечение основанное на разработанных алгоритмов ранжирования и генерации по рангу вариантов дерева И/ИЛИ для комбинаторного множества последовательностей вариантов ответа на тест с вопросами закрытого типа было применено для формирования выборки ответов для тестирования системы оценивания.

Разработанное программное обеспечение соответствует заявленным требованиям и позволяет генерировать выборку ответов обучающихся с заданной точностью правильных ответов на задания типа «тест».

Применение разработанного решения позволило уменьшить временные затраты на 50% во время тестирования систем тестового оценивания.

Директор ООО «Эль Контент» А.В. Городович

31 03 2022

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

«УТВЕРЖДАЮ»
И.о. директора Инженерной школы
информационных технологий
и робототехники ТПУ, к.т.н.
_____ А.Ю. Демин
«28» 04 2022 г.

АКТ
использования результатов диссертационной работы
Кручинина Дмитрия Владимировича в образовательном процессе
Инженерной школы информационных технологий и робототехники
«Национального исследовательского Томского политехнического университета»

Настоящий Акт составлен в том, что в образовательный процесс Инженерной школы информационных технологий и робототехники внедрены следующие результаты:

– модифицированный метод комбинаторной генерации, основанный на применении деревьев И/ИЛИ и производящих функций многих переменных;

– алгоритмы комбинаторной генерации для информационных объектов;

– база знаний производящих функций двух переменных и коэффициентов их степеней, состоящая из 1500 фреймов;

– программное обеспечение, представленное в виде библиотек для получения явных выражений коэффициентов производящих функций многих переменных.

Результаты диссертационной работы используются в учебном процессе при подготовке студентов направления подготовки 09.03.01 «Информатика и вычислительная техника» и 09.03.02 «Информационные системы и технологии».

Руководитель отделения информационных
технологий инженерной школы
информационных технологий
и робототехники, к.т.н. В.С. Шерстнев

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

УТВЕРЖДАЮ
Проректор по учебной работе ТУСУР
_____ П.В. Сенченко
«_08_» «_____04_____ 2022 г.

АКТ
внедрения результатов диссертационной работы на соискание степени доктора технических наук в Томском государственном университете систем управления и радиоэлектроники
Кручинина Дмитрия Владимировича

Комиссия в составе:

Председатель комиссии: Черкашин М.В., *канд. техн. наук, доцент, декан факультета вычислительных систем*

члены комиссии:

Шурыгин Ю.А., д-р техн. наук, профессор, зав. каф. КСУП;

Коцубинский В.П. канд. техн. наук, доцент, зам. зав. каф. КСУП

составили настоящий акт о том, что научные результаты диссертационной работы Кручинина Дмитрия Владимировича «Методы, алгоритмы и программное обеспечение на основе производящих функций многих переменных для комплексного исследования информационных объектов», представленной на соискание ученой степени доктора технических наук, внедрены в учебный процесс Томского государственного университета систем управления и радиоэлектроники по направлениям обучения «Информатика и вычислительная техника», «Управление в технических системах».

Разработанные Кручининым Д.В. методы и программное обеспечение использовались в рамках лекционных курсов по дисциплинам «Дискретная математика» и «Структуры данных» и научно-исследовательской работы студентов. В частности, данные результаты позволили сформировать навыки составления алгоритмов, разработки программ на алгоритмических языках и тестирования работоспособности программ, построенных на дискретных структурах.

Также в рамках создания автоматизированной системы обучения математических дисциплинам Кручининым Д.В. была разработана программа для создания математических заданий путем генерации внутренних параметров и для формирования обратной связи по анализу дерева ответов обучающегося. Программа представляет собой программный модуль на языке системы компьютерной алгебры Maxima в плагине Stack системы Moodle. Программа может использоваться для формирования комплекса адаптивных тренажеров с обратной связью для обучения школьной математике и высшей математике для студентов первых курсов. Внедрение предложенного решения позволило сократить временные затраты на создание и проверку контрольных и домашних работ за счет автоматизации данного процесса.

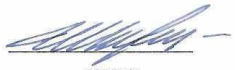
Разработанные решения были апробированы на более 500 студентах первого курса по дисциплине «Математический анализ».

Председатель комиссии:

к.т.н, доцент, декан ФВС  Черкашин М.В.

подпись

Члены комиссии:

д.т.н., проф., зав. каф. КСУП  Шурыгин Ю.А.

подпись

к.т.н., доцент,

зам. зав. каф. КСУП  Коцубинский В.П.

подпись